

The next generation CONTROL®

version 10.6

Analysis Services Integration



Table of Contents

Overview.....	3
AS Models vs. CONTROL Models.....	3
Similarities	4
Differences	4
A few words about DAX.....	5
Control Power Pivot Models.....	6
CONTROL to AS.....	6
What it does and why.....	6
How it works.....	7
Automatic Import.....	8
An Example	9
AS to CONTROL.....	12
What it does and why.....	12
How it works.....	13
A simple example	15
Customizing the import	20
Import Rules.....	20
Using Import Rules to combine AS tables	23
Customizing the AS Column Map.....	25
Customizing the Imported Dimensions	27
Copying AS Models	28
Views.....	29
What is the same?	29
What is different?	30
Use in custom dimensions	31
The Generation Option.....	34



Use in CONTROL Web	36
Leveraging and Extending	36
The variable hierarchy.....	36
Base members	37
Ad hoc members	37
Mappings	38
Meta-data.....	38
On-demand data - Power Pivot to Computational models	38
Dynamic data - Power Pivot to Computational models	41
Dynamic data - Power Pivot to Power Pivot models	44
An Architectural recommendation.....	46
AS Query Data Sources.....	46
Data source properties.....	46
Use in Mappings.....	48
Source Data Views.....	48
ASQuery Flex Function	49
Source data models.....	49



Overview

SQL Server Analysis Services (AS) is at the core of Microsoft's business intelligence strategy. It is one of the key technologies that underpins the wildly popular Power BI visualization products. AS is an extremely powerful, flexible, and robust data management, query, computation, and manipulation platform that is blazingly fast, widely available at low cost, and has a large and growing ecosystem of developers and users across industries and geographies. Microsoft is investing heavily to stay on top of an extremely competitive market.

KCI believes that embracing this technology can bring significant benefits to CONTROL applications in the financial and operational domains it serves. Here are some of the benefits:

- Seamlessly integrating the CONTROL's planning, forecasting, and analytics functions and the visual business intelligence domain of Power BI
- Extending Excel-embedded reporting and modeling capabilities of CONTROL to a broad spectrum of financial and operational functions
- Leveraging innovative features of the Microsoft technology stack such as Artificial Intelligence in CONTROL applications
- Implementing CONTROL applications more quickly and easily by building on models built for AS or Power BI
- Improving the speed and memory footprint of CONTROL reporting applications
- Streamlining the creation of complex calculations by using the powerful computational language of AS (DAX)
- Enabling Excel delivery and distribution of AS data for applications created and maintained outside of CONTROL
- Opening the door to the wide world of Power BI experts

As with any rich new feature set, we expect that our customers and consultants will find all manner of creative ways to apply these capabilities.

AS Models vs. CONTROL Models

To understand both the value and the challenges of using AS with CONTROL, it is essential to understand the similarities and differences between how the two products approach the multi-dimensional application space.

The discussion below relates to the Tabular model, which has become the dominant option in Microsoft's customer base.



Similarities

From 10,000 feet, both CONTROL and AS (and many other multi-dimensional products) look identical. They organize numeric data so that it can be sliced and diced according to key characteristics.

Specifically:

- The primary unit of organization is a “model”
- A model can have one or more “dimensions” which contain the elements related to the model’s keys – e.g., Product, Geography, Time, etc.
- Each dimension groups information related to the key – for example a product may be part of a brand, and a category, have a size, a weight, or a color
- Data can be aggregated or summarized based on the information in a dimension – e.g., departmental data can be summarized by division or business unit
- There is an ordered progression of summarization supported in some or all dimensions – e.g., days to weeks to months to quarters etc.
- The numeric data in the model can be elemental and come from an external source or via direct entry – such as number of units sold and sales price; or be calculated based on other data – such as gross sales (units x price) or profit (sales – cost)
- Each user’s access to the data in a model can be specified through “roles” which limit which members of each dimension can be read or written

Both CONTROL and AS offer multiple interfaces for the user to interact with the data they manage. AS can be presented via Power BI, SSRS (SQL Server Reporting Services), Excel pivot tables, and other third-party tools. CONTROL is primarily utilized from Excel and CONTROL Web.

Differences

The divergence of the CONTROL and AS architectures can largely be traced to the intended purpose of each product.

AS was designed to be an extremely efficient engine for storing and computing data for business intelligence. Once loaded, the data and meta-data in AS is relatively static.

CONTROL is a platform for creating and managing modeling, forecasting, and analytic solutions for finance and operations. The data and meta-data in CONTROL can change from minute to minute. Due to its focus on financial applications, CONTROL must be consistent, auditable, and reconcile exactly to systems of record.

Here are the key differences to keep in mind:

- AS is **read-only**. CONTROL is **read/write**.



- AS models are **segregated**, each isolated in their own database. CONTROL models are contained in a single database and can interact via mappings.
- CONTROL models have a strict “**star-schema**” structure, with all information for a dimension in a single table. AS models may have a “**star schema**” or a “**snowflake schema**”, with information distributed in multiple tables (although the star schema is the recommended organization).
- CONTROL models have a **single fact** table, AS models may have **multiple fact** tables.
- CONTROL’s **dimensions are strongly “typed”** – scenario, organization, variable, and time. AS dimension tables may have a type (Time is common) or maybe completely un-typed.
- The calculation **logic** in CONTROL is specified dimension by dimension, with complex formulas typically associated with the **variable** dimension. AS model calculations can be in calculated columns or tables, but most commonly are defined in **measures**, which can reside in any table and may reference any part of the model.
- CONTROL dimensions are comprised of **levels and attributes** which have strictly specified relationships. AS dimensions have **fields** (columns), and no specified relationships. Level members have ID’s, names, and descriptions.
- A CONTROL model can have **multiple alternate hierarchies** for each dimension. AS models cannot.
- CONTROL dimensions, hierarchies, levels, and attributes are **reusable** across models.
- CONTROL models intrinsically support **commentary** and **dynamic currency translation**, while AS models must be explicitly designed to have those capabilities.
- CONTROL supports **non-uniform levels of detail** in multi-level dimensions. AS does not.

While these differences present some challenges, they also offer opportunities to leverage the strengths of each product to deliver more functional and higher value applications.

The goal of the integration is to give the application designer the flexibility to take advantage of the functionality and performance of AS, without the burden of dealing with the bloody details.

A few words about DAX

DAX is the formula language that is used to define how measures are calculated in AS. It is also a query language that can be used to extract data from AS models.

DAX is an enormously powerful language for performing calculations and data manipulations on multi-dimensional data. These capabilities overlap with intrinsic CONTROL features and functions to a large degree, but also provide easier ways to accomplish some tasks, and can handle requirements that were previously very cumbersome to address.



While DAX is not easy to master, it is a skill that is essential to using Power BI, and will prove enormously valuable in CONTROL applications that leverage AS.

We will be providing examples and patterns in this (10.6) and future releases.

Control Power Pivot Models

Central to all the new capabilities is a new subclass of CONTROL model, the *Power Pivot* model. This new subclass of model has additional properties, particularly the Usage property, which determine how the Power Pivot model relates to AS:

- Control to AS
- AS to Control
- Control to Relational

CONTROL to AS

The ability to export a CONTROL data to use in Power BI was implemented in the 10.2 release and has been enhanced significantly in subsequent releases. The functionality and usage details are described in the “Power BI Integration” document. Please refer to that document if your primary interest is in Power BI.

Here we will focus on the use of CONTROL Power Pivot models that have been exported to AS from CONTROL.

What it does and why

The mechanisms to integrate relationally managed CONTROL data to AS are encapsulated in Power Pivot models which have the Usage property “CONTROL to AS”.

The Power Pivot model has these customizable features:

- A base model and view
- Export properties
 - Anchor dimension
 - Augmentation options
 - Materialization options
- Additional views (which may include source data)
- Calculation groups
- Additional tables (datasources) which may be calculated AS tables
- Customizable definitions of measures and KPI's

These features enable an enormous range of sophistication in the resulting AS model.



In the simplest case, the AS model is a mirror image of a CONTROL model and view. With virtually no knowledge of AS or DAX, you can be creating great visuals in Power BI from CONTROL data exported to AS.

At the other end of the spectrum, you can build composite models which incorporate data from multiple CONTROL models and datasources with different dimensionality and level of detail and incorporates time intelligence and customized calculations across models and dimensions.

The result is a useful model that anyone even remotely familiar with Power BI, Reporting Services, or other tools can use immediately. The meta-data relationships reflect the cleansing, validation, and augmentation performed via CONTROL processes. The data is reconciled and up to date, and shareable by any number of users, with or without CONTROL licenses. When changes take place, it is simple (and scriptable) to refresh the AS model.

How it works

In some respects, the Analysis Services architecture is “simpler” than the CONTROL architecture. Basically, an AS model is a set of tables, and relationships (typically one to many) between those tables. (There are some rules about what is acceptable, but CONTROL applications generally obey those rules.)

Here are the details of how an AS model is built from the CONTROL Power Pivot model and view:

- Each non-anchor dimension-branch of the base view is exported as a table, with the lowest level of the branch as the “primary key” of the table. Both ID’s and member names are included in the table, as are any listing levels or attributes in the branch.
- Depending on the Augmentation property of the Power Pivot model, additional levels and attributes may be included in the table.
- If there is a filter on the dimension-branch, only members included by the filter are included in the table.
- If the view includes multiple scenarios having different hierarchies, the hierarchies are “merged”.
- “Branches” in CONTROL become “Hierarchies” in the AS model.
- A “Fact” table is built from the data in the base view. The fact table does not include aggregated time or organization values.
- Some anchor members become data columns in AS, some become calculated columns, and some become measures.
- If there is an anchor dimension, a column is added to the fact table for each member of that dimension, including any computed or aggregated values.
- If there is a protocol specified, the fact table in AS is partitioned.



- Relationships are created from the dimension tables – the “one” side, to the fact table – the “many” side.
- The process is repeated for additional views with the following caveats:
 - Any dimension-branches that are common with the base view or any preceding view (in terms of filter and branch) are not added.
 - Each additional view has its own fact table in AS
- Any calculation groups are added.
- Any extra tables are added, along with their relationships.
- An “Information” table is added, to facilitate debugging, synchronization, and documentation.

Automatic Import

As you will read later in this document, it’s possible to open CONTROL computational views on AS models. This essentially allows for AS models to be substituted for Control computational models in CONTROL applications. As part of the CONTROL to AS export process, the resulting AS model(s) are automatically conditioned so that they can be substituted for CONTROL computational models.

To achieve this end, there is a process which makes a Power Pivot model “look like” a Computational model. It must have one scenario, time, and variable dimension, and one or more organization dimensions.

This implies the following restrictions on the definition of the Power Pivot model:

- The Power Pivot Anchor must be the variable dimension or unanchored
- If there are multiple views, they must share common scenario and time dimensions
- No view may have a non-anchored custom dimension

This is how the dimensions and hierarchies of the Power Pivot model are defined:

- The non-anchor dimensions and hierarchies of the base model/view are inherited from the base model/view
- Any organization dimensions of the additional model/views which are not in common with the base are inherited from those models/views
- If the variable dimension is the anchor dimension, or if there are multiple variable dimensions amongst all views, a new variable dimension is created
- If the variable dimension is not anchored, a new hierarchy is created with the DAX formula type
- Any non-variable dimension which has a filter other than “All” or the lowest level of its branch different than the root level of the dimension, will have a subset hierarchy created and assigned to that dimension in the model
- Calculation groups are translated to organization dimensions



CONTROL®

- Additional tables which have a relationship to any fact table are added as organization dimensions

An Example

This example exports data from a departmental expense model and incorporates data from a workforce model which tracks resources by employee.

Here are the properties of the CONTROL Power Pivot model:



KKI Properties for Model Department Expense Max Multi

Search

Identification

Name	Department Expense Max Multi
ID	DEPARTMENTEXPENSEMAXMULTI
Class	Model
Subclass	Power Pivot
Category	Development (ID: DEVELOPMENT)
Description	

Definition

Base Model	Expense (ID: EXPENSE)
Base View	Department Expense Max (ID: DEPARTMENTEXPENSEMAX)
Usage	Control to AS

Logging

Accessibility

Miscellaneous

Power Pivot Options

Power Pivot Style	Dimension and Fact Tables (Star Join)
Power Pivot Anchor	Variable
Materialization Beh...	Materialize
Materialization Protoc...	
Table Schema	
Table Name Or Prefix	CAS1
Augment Dimensions	Add all fields to all dimensions
Refresh Behavior	Manual
Export Status	Exported and up to date
Last Export Time	Tuesday, July 06, 2021 5:51:56 PM
AS Server	
AS Database[AS Model]	DEPARTMENTEXPENSEMAXMULTI
Impersonation Mode	Service Account
AS Connection Behavi...	Reconnect using current details
Import Rules Data So...	(None)

Usage

Specifies how this model will be used:

- Control to Relational - data and meta-data will be exported to relational tables and views, for use in other applications.
- Control to AS - data and meta-data will be exported to create an Analysis Services Tabular database and model, which can be used in Power BI, Excel pivot tables, and read-only Control views and mappings
- AS to Control - meta-data in an Analysis Services Tabular model will be imported to Control as dimensions, hierarchies and levels, so that the AS data can be used in Control views and mappings

(Note: Control to AS - Power BI only is a limited case of Control to AS, and is not generally recommended)

OK Cancel

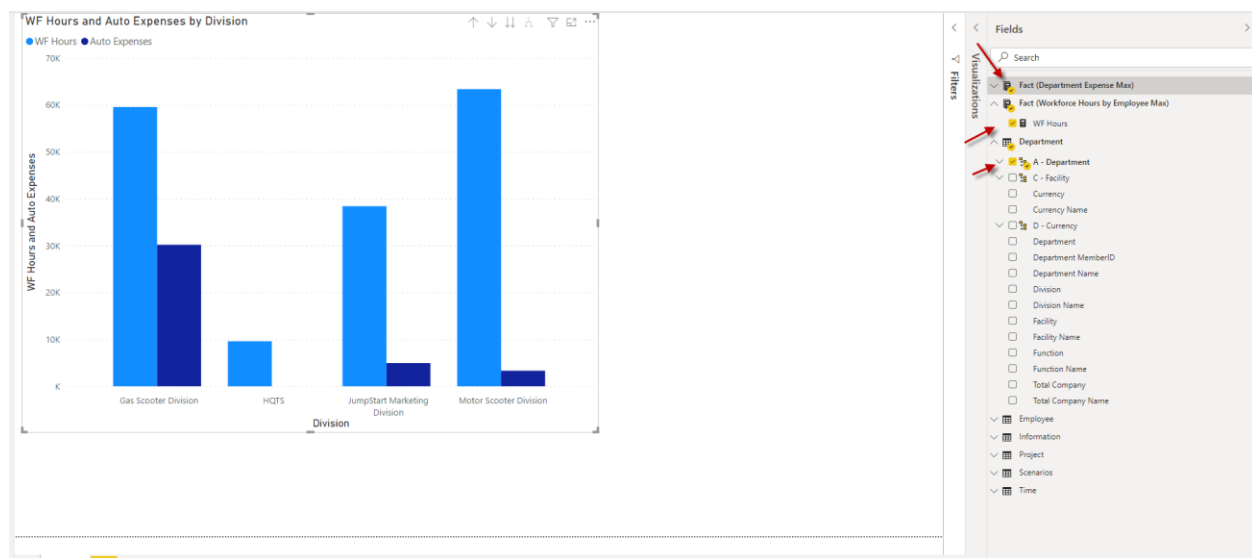
Here is the specification of the view on the workforce model:



CONTROL®

The screenshot shows the CONTROL Model interface. The main workspace is a grid with columns A through K and rows 1 through 35. A large blue 'EDIT BOOK' overlay is centered on the grid. A red arrow points from the 'EDIT BOOK' text to the 'Additional Views' tab in the bottom navigation bar. Another red arrow points from the 'Additional Views' tab to the 'PowerPivot Views for Model - Department Expense Max Multi' pane on the right. This pane contains a 'Selected Content' list with 'Workforce Hours by Employee' selected, and a 'Power Pivot Item Properties' section below it.

Here is how the exported model appears in Power BI.





Note:

- Two fact tables – one from each CONTROL computational model
- Shared dimension tables – Department, Project, Scenario, and Time
- The Employee dimension table from the Workforce model
- The presentation of data from the two models in a single visualization

AS to CONTROL

Wouldn't it be fabulous to have all of CONTROL's reporting, analysis, and modeling capabilities instantly available on any data model built in AS or Power BI? All the flavors of views, drilling flexing, using on the Web, integrating with other CONTROL information? And effortlessly accessed in Excel?

AS to CONTROL usage accomplishes just that!

A technical note:

The same technology is at the heart of AS Tabular, Power BI, and Excel Power Pivot models. It is called the Vertipaq engine. The interface to that engine is what is used to accomplish this integration with CONTROL. As Microsoft removes limitations on this interface, we will provide broader access and improved functionality in this area.

What it does and why

This process makes an AS model look like a CONTROL computational model, so that you can take advantage of all the features and functions of CONTROL, except updating data.

Based on the structure of the AS model, this process can be extremely straightforward or somewhat complicated.

The import process translates the AS model's meta-data into their corresponding CONTROL object model components: dimensions, hierarchies, levels, and attributes. The numeric data is not imported but instead is accessed directly from AS.

The tricky parts are:

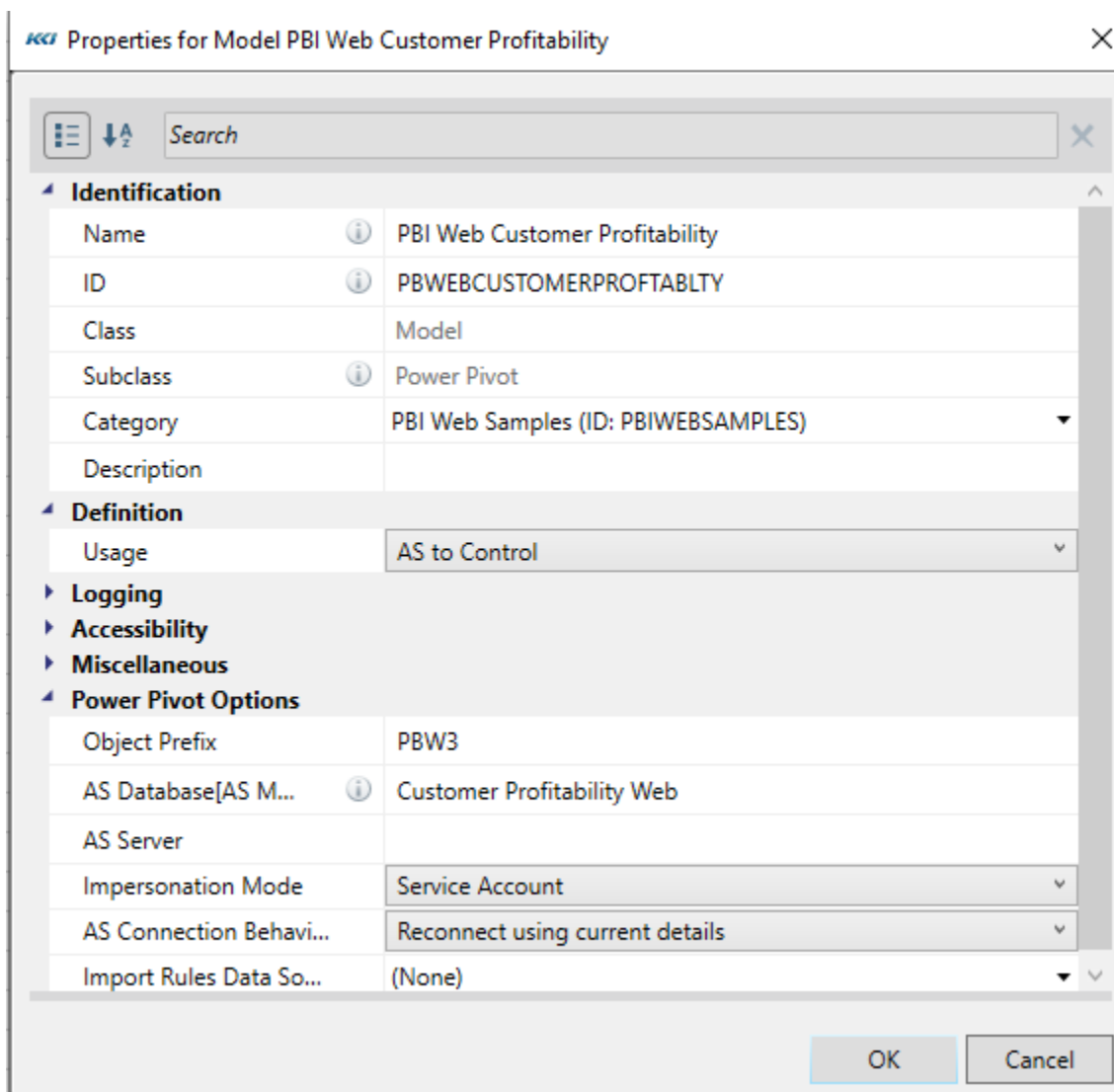
- Determining how to divide up dimensions and where the "fact" tables are.
- How to assign a dimension's subclass.
- Deciding what is a level and what is an attribute.
- For a level, whether a field is an ID or name.
- How to turn table and field names into CONTROL object ID's and names.
- How to determine the relationships among the levels and attributes of a dimension.
- How to decide which columns in the AS tables should be CONTROL variable members.

Because the match between AS and CONTROL is not exact, the default translation process may make incorrect assumptions, so there are several ways to intervene to get the desired result.

The guiding principle of the import is that data presented via CONTROL views or mappings is identical to that presented via Power BI.

How it works

Once you create a Power Pivot model with the usage property set to AS to CONTROL, you need to define the AS Server, AS Database and Object Prefix (which will be defined automatically if you leave it blank):



Properties for Model PBI Web Customer Profitability

Search

Identification

Name	PBI Web Customer Profitability
ID	PBWEBCUSTOMERPROFITBLTY
Class	Model
Subclass	Power Pivot
Category	PBI Web Samples (ID: PBIWEBSAMPLES)
Description	

Definition

Usage	AS to Control
-------	---------------

Logging

Accessibility

Miscellaneous

Power Pivot Options

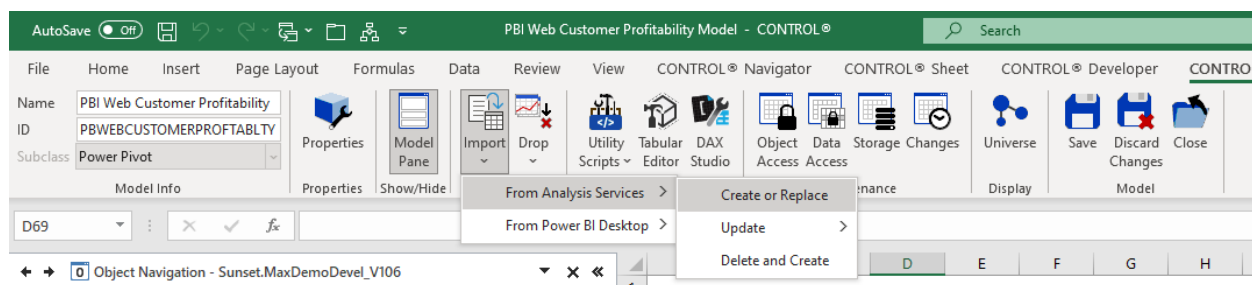
Object Prefix	PBW3
AS Database[AS M...	Customer Profitability Web
AS Server	
Impersonation Mode	Service Account
AS Connection Behavi...	Reconnect using current details
Import Rules Data So...	(None)

OK Cancel

If the AS Server is left blank, it is defined by the replacement value of the &KCI_AS_SServer keyword.



When you open the model's edit book and Import from Analysis Services:



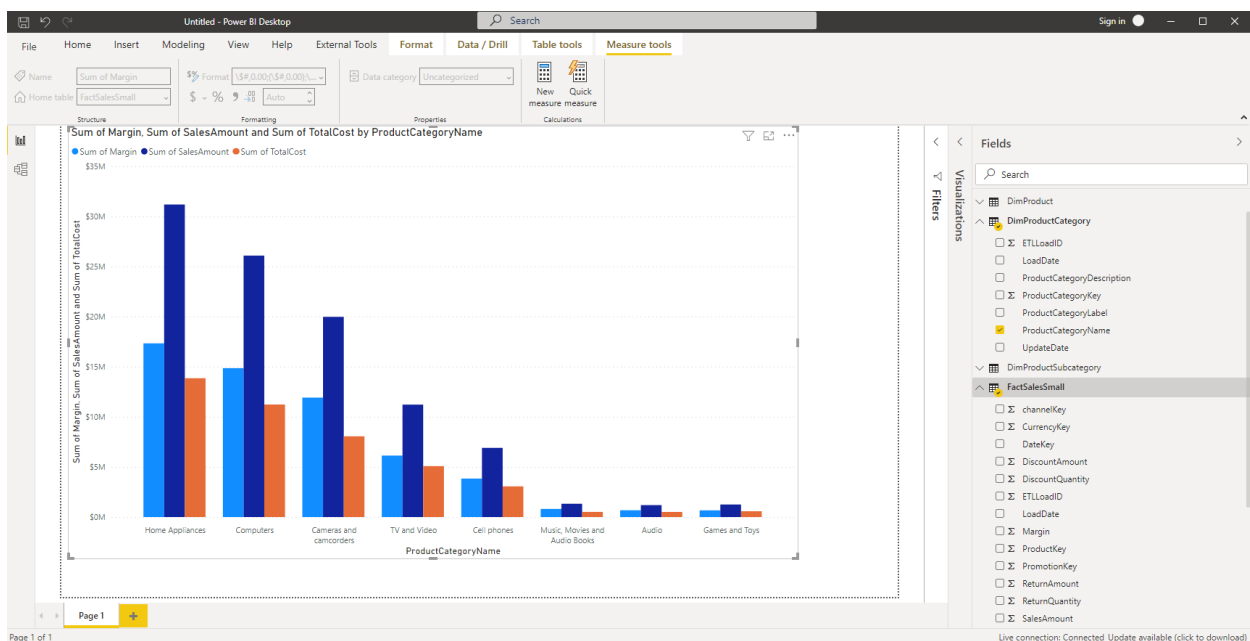
Here is what happens:

- A connection is established to the AS Server and database, and the list of tables in the model, their relationships, and details about each table are retrieved.
- If the model has an Import Rules data source, the rules are retrieved. If none is specified, a default set of rules are defined. (Detailed descriptions of the supported rules follow.)
- A list of dimension tables is defined. Any table which is on the one side of a many to one relationship is considered a dimension table. The import rules can override the default choice for a table.
- Each dimension table is turned into a CONTROL dimension using the following logic:
 - If the table is hidden, it is not imported unless explicitly requested by an import rule
 - The import rules and a heuristic algorithm are used to classify each column as:
 - A level ID
 - A level name
 - An attribute
 - Ignored – hidden columns are ignored unless they are the “root” level – connected as the “to” or “one” side of a relationship
 - In cases where a level name is present and there no ID, an ID is constructed based on the type and size of the column
 - If there is more than one column which is not ignored, and no column that has a single value, a total level is added
 - The reporting relationships between the levels and attributes is inferred from the observed relationships among the columns and a new dimension is defined, along with its levels and attributes
 - The standard hierarchy of the dimension is constructed from the table's contents
 - Levels associated with columns that are sorted (other than alphabetical or numeric order) inherited into the level ordering

- Any hierarchies associated with the table are defined as branches of the dimension
 - The automatically generated branches of the dimension are cleaned up to be more usable in CONTROL
- A list of fact tables is defined. Any table which contains at least one measure is considered a fact table. The import rules can override which tables are considered fact tables.
- For each fact table:
 - A list of variable members is defined including all measures and any visible numeric column with some exceptions – e.g. names like "Key", "ID", "Date", etc.
 - The format, DAX expression, and display folder for grouping of each variable member are defined
- The unified list of variable members is used to create a variable dimension, levels, and a hierarchy whose Formula Type is DAX
- The Power Pivot model's dimension list is constructed from the dimension tables, variable dimension, and default scenario and time dimensions, if those are not included in the AS model

A simple example

The Power BI visualization below is based on an example AS database distributed by Microsoft, called Contoso.





The model has 3 tables related to products – DimProduct, DimProductSubCategory, and DimProductCategory; a Dates table which summarizes daily dates into months and years, and a FactSalesSmall table which has detailed sale, cost, and return data and a few computed measures, including margin.

To use the AS model in CONTROL, we create a Power Pivot model and set the properties as follows:

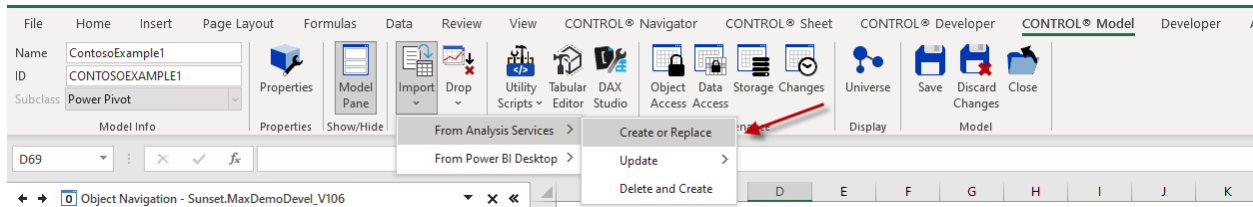
Properties for Model ContosoExample1	
Identification	
Name	ContosoExample1
ID	CONTOSOEXAMPLE1
Class	Model
Subclass	Power Pivot
Category	Development (ID: DEVELOPMENT)
Description	
Definition	
Usage	AS to Control
Logging	
Accessibility	
Miscellaneous	
Power Pivot Options	
Object Prefix	AS1
AS Server	Bells
AS Database[AS M...	ContosoExample1
Impersonation Mode	Service Account
AS Connection Behavi...	Reconnect using current details
Import Rules Data So...	Import AS Contoso (ID: IMPORTASCONTOSO)

The AS Server and AS Database specify which AS model we are using. The Object Prefix is optional and is used to distinguish the CONTROL objects that will be created from other dimensions, hierarchies, levels, and attributes.



CONTROL®

To execute the import, edit the model, select Import, From Analysis Services, and Create or Replace:



The process can take a few minutes. When it is complete, you can see the newly created dimensions in the task pane:

Model - ContosoExample1

Available Dimensions

Used Dimensions

Scenario

Organization

No Scenario

AS1 Dates

AS1 DimProduct

AS1 DimProdu...

AS1 DimProdu...

Variable

Time

AS1 Measures

No Time

Dimension Details

☐ Immediate update

Update Maximize

Note that a CONTROL dimension has been built for every table in the AS model, except for FactSalesSmall.



CONTROL®

Here is an example of how the DimProduct table was turned into a CONTROL dimension and hierarchy:

The screenshot shows the 'AS1 DimProduct' table in a spreadsheet and its representation in the CONTROL interface. The table lists various product models with their AS1 Product IDs and ProductKeys. The CONTROL interface shows a dimension structure with levels: AS1 Status, AS1 Size/Measure, AS1 Size/Range, AS1 DimProduct Total, AS1 Color, AS1 Product, AS1 Size, AS1 Weight/Measure, AS1 Unit/Measure, and AS1 StockType. The hierarchy is visualized as a tree structure.

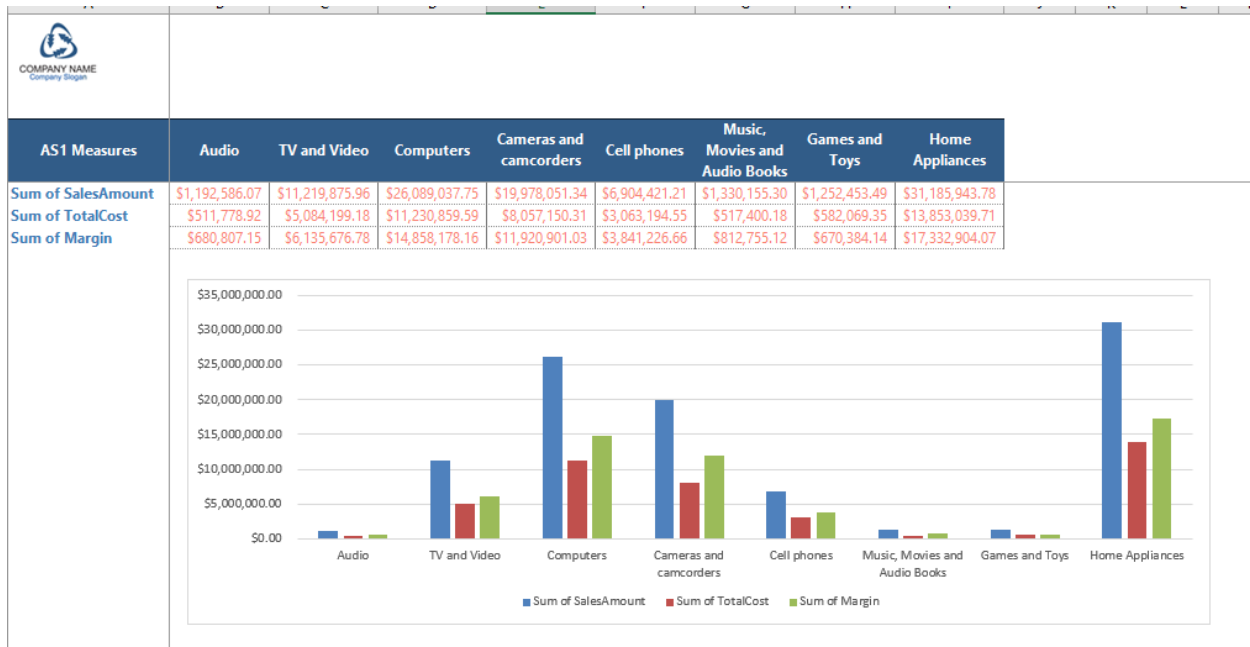
AS1PRODUCT_ID	AS1PRODUCT Name	AS1PRODUCTKEY_ID	AS1PRODUCTKEY
0101001	Contoso 512MB MP3 Player E40 Silver	1	ProductKey 1
0101002	Contoso 512MB MP3 Player E41 Blue	2	ProductKey 2
0101003	Contoso 1GB MP3 Player E100 White	3	ProductKey 3
0101004	Contoso 2GB MP3 Player E200 Silver	4	ProductKey 4
0101005	Contoso 2GB MP3 Player E200 Red	5	ProductKey 5
0101006	Contoso 2GB MP3 Player E200 Black	6	ProductKey 6
0101007	Contoso 2GB MP3 Player E200 Blue	7	ProductKey 7
0101008	Contoso 4GB MP3 Player E400 Silver	8	ProductKey 8
0101009	Contoso 4GB MP3 Player E400 Black	9	ProductKey 9
0101010	Contoso 4GB MP3 Player E400 Green	10	ProductKey 10
0101011	Contoso 4GB MP3 Player E400 Orange	11	ProductKey 11
0101012	Contoso 4GB Flash MP3 Player E401 Blue	12	ProductKey 12
0101013	Contoso 4GB Flash MP3 Player E401 Black	13	ProductKey 13
0101014	Contoso 4GB Flash MP3 Player E401 Silver	14	ProductKey 14
0101015	Contoso 4GB Flash MP3 Player E401 White	15	ProductKey 15
0101016	Contoso 8GB Super-Slim MP3 Video Player M800 White	16	ProductKey 16
0101017	Contoso 8GB Super-Slim MP3 Video Player M800 Red	17	ProductKey 17
0101018	Contoso 8GB Super-Slim MP3 Video Player M800 Green	18	ProductKey 18
0101019	Contoso 8GB Super-Slim MP3 Video Player M800 Pink	19	ProductKey 19
0101020	Contoso 8GB MP3 Player new model M820 Black	20	ProductKey 20
0101021	Contoso 8GB MP3 Player new model M820 Blue	21	ProductKey 21
0101022	Contoso 8GB MP3 Player new model M820 Yellow	22	ProductKey 22
0101023	Contoso 8GB MP3 Player new model M820 White	23	ProductKey 23
0101024	Contoso 16GB Mp3 Player M1600 Blue	24	ProductKey 24
0101025	Contoso 16GB Mp3 Player M1600 Black	25	ProductKey 25
0101026	Contoso 16GB Mp3 Player M1600 Green	26	ProductKey 26
0101027	Contoso 16GB Mp3 Player M1600 White	27	ProductKey 27
0101028	Contoso 16GB Mp3 Player M1600 Red	28	ProductKey 28
0101029	Contoso 32GB Video MP3 Player M3200 White	29	ProductKey 29
0101030	Contoso 32GB Video MP3 Player M3200 Red	30	ProductKey 30
0101031	Contoso 32GB Video MP3 Player M3200 Orange	31	ProductKey 31
0101032	Contoso 32GB Video MP3 Player M3200 Pink	32	ProductKey 32
0101033	Contoso 32GB Video MP3 Player M3200 Black	33	ProductKey 33
0101034	Contoso 4GB Portable MP3 Player M400 Black	34	ProductKey 34
0101035	Contoso 4GB Portable MP3 Player M400 White	35	ProductKey 35
0101036	Contoso 4GB Portable MP3 Player M400 Yellow	36	ProductKey 36
0101037	Contoso 8GB Clock & Radio MP3 Player X800 Silver	37	ProductKey 37
0101038	Contoso 8GB Clock & Radio MP3 Player X800 Black	38	ProductKey 38
0101039	Contoso 8GB Clock & Radio MP3 Player X800 Green	39	ProductKey 39
0101040	Contoso 8GB Clock & Radio MP3 Player X800 Blue	40	ProductKey 40
0101041	Contoso 16GB New Generation MP3 Player M1650 Silver	41	ProductKey 41
0101042	Contoso 16GB New Generation MP3 Player M1650 White	42	ProductKey 42
0101043	Contoso 16GB New Generation MP3 Player M1650 Black	43	ProductKey 43
0101044	Contoso 16GB New Generation MP3 Player M1650 Blue	44	ProductKey 44
0101045	Contoso 16GB New Generation MP3 Player M1650 Pink	45	ProductKey 45

The newly created variable dimension uses the notation "factTable:DAXExpression" in the direct logic defining each variable member:

The screenshot shows the 'AS1 Measures' table in a spreadsheet and its representation in the CONTROL interface. The table lists various measures with their AS1 Measure IDs, Names, and Direct Logic. The CONTROL interface shows a dimension structure with levels: AS1 Measure Summary and AS1 Measure Summary. The hierarchy is visualized as a tree structure.

AS1MEASURESUMMARY_ID	AS1MEASURESUMMARY_ID	Name	Direct Logic
ASMODELMEASURES	SUMOFSALESAMOUNT	Sum of SalesAmount	FactSalesSmall[Sum of SalesAmount]
ASMODELMEASURES	SUMOFTOTALCOST	Sum of TotalCost	FactSalesSmall[Sum of TotalCost]
ASMODELMEASURES	SUMOFMARGIN	Sum of Margin	FactSalesSmall[Sum of Margin]
ASMODELMEASURES	UNITCOST	UnitCost	FactSalesSmallSUM('FactSalesSmall' (UnitCost))
ASMODELMEASURES	UNITPRICE	UnitPrice	FactSalesSmallSUM('FactSalesSmall' (UnitPrice))
ASMODELMEASURES	SALESQUANTITY	SalesQuantity	FactSalesSmallSUM('FactSalesSmall' (SalesQuantity))
ASMODELMEASURES	RETURNQUANTITY	ReturnQuantity	FactSalesSmallSUM('FactSalesSmall' (ReturnQuantity))
ASMODELMEASURES	RETURNAMOUNT	ReturnAmount	FactSalesSmallSUM('FactSalesSmall' (ReturnAmount))
ASMODELMEASURES	DISCOUNTQUANTITY	DiscountQuantity	FactSalesSmallSUM('FactSalesSmall' (DiscountQuantity))
ASMODELMEASURES	DISCOUNAMOUNT	DiscountAmount	FactSalesSmallSUM('FactSalesSmall' (DiscountAmount))
ASMODELMEASURES	TOTALCOST	TotalCost	FactSalesSmallSUM('FactSalesSmall' (TotalCost))
ASMODELMEASURES	SALESAMOUNT	SalesAmount	FactSalesSmallSUM('FactSalesSmall' (SalesAmount))
ASMODELMEASURES	MARGIN	Margin	FactSalesSmallSUM('FactSalesSmall' (Margin))
ASMODELMEASURES		AS Model Measures	

Here is how a CONTROL view on the model might look:



Customizing the import

If the import process succeeds, you will have a usable model in CONTROL, on which you can define views and mappings, and the data will have complete fidelity to the AS model.

However, the imported CONTROL model may not be ideal for any of the following reasons:

- There may be multiple tables that represent a single dimension (e.g., DimProduct, DimProductSubcategory, and DimProductCategory in the example above)
- Fields in a table get defined as part of a level instead of an attribute or vice versa.
- The subclass of a dimension is different from what you would expect (e.g., Dates looks like a time dimension – not an organization dimension)
- The reporting relationships among levels and attributes is not to your liking.
- Some fields that are not useful included in a dimension, or useful tables or fields that are hidden in AS would be useful in CONTROL.

Import Rules

Each Power Pivot model can have an optional data source which contains a set of rules that override the import decision making process. The rules are applied in sequence, and can be specific to a column, a table, or all tables.

Here is the import rule data source for the previous example:

RuleNumber	ASColumnName	ControlSpecification	Tables	Comment
1	Date	Date IDandName		
2	*Date	*Date Attribute		
3	*LoadID	*LoadID Attribute		
4	StyleName	StyleName Attribute		
5	Size	Size Attribute		
6	*ID	* ID		
7	*Label	* ID		
8	*Number	* ID		
9	*Name	* Name		
10	*Description	*Description Attribute		
11	*Key	*Key IDandName		
12	*URL	DontImport		

The columns have the following meanings:

- Rule number is used to order the table. Rules are applied in order, and for a column, if an applicable rule is found, subsequent rules are ignored.
- ASColumnName is the specific name of a column in an AS table, or a wildcard (*) prefix that will match any column whose name ends with the trailing characters.
- ControlSpecification is the explicit text of the rule, or a wildcard (*) prefix that dictates whether the column will be treated as a level ID, a level name, or an attribute.
- Tables is the name of the table that the rule applies to. If this column is blank, the rule is used on all tables.
- Comment is used to document the intent of the rule

These rules apply to an entire AS table – the ASColumnName column should be left blank:

ControlSpecification	Import Behavior
DontImport	The table will be ignored and not imported, regardless of whether it is hidden or visible
DoImport	The table will be imported, regardless of whether it is hidden or visible
ImportFact	The table will be imported as a fact table
ImportDimension	The table will be imported as a dimension table
ImportBoth	The table will be imported as both a dimension and a fact table. Note that it must contain at least one column that has unique values, or the dimension import will fail
ImportDimensionRelated	The table and all tables with a one-to-many relationship to this table will be merged into a single calculated table which will then be imported as a dimension table.

	<p>The merged table will have the name of the specified table with a suffix of "Complete" – for example "DimProductComplete". All the tables involved in the relationships are automatically ignored.</p> <p>Any import rule column references which apply to any of the constituent tables automatically apply to the merged table. This option makes it simple to have a snowflake schema turned into a star schema for more understandable use in CONTROL.</p>
ImportBothRelated	Imports the table as a dimension using ImportDimensionRelated and as a fact table

These rules apply to individual columns on either a specified table or all tables:

ControlSpecification	Import Behavior
Xxxx ID *Xxxx ID	<p>Column Xxxx will be treated as a level ID of a new level named Xxxx, with an ID of prefixXxxxx</p> <p>Similarly, *Xxxx ID will treat any column where the last characters of the column name match Xxxx will be treated as a level ID with a level name that matches the column name</p>
Xxxx Name *Xxxx Name	Column Xxxx will be treated as a level name of a new level named Xxxx, with an ID of prefixXxxxx
Xxxx IDandName *Xxxx IDandName	<p>Column Xxxx will be treated as both the name and ID of a level. The structure of the ID and name depends on the datatype of the column.</p> <p>DateTime columns will have ID YYYYMMDD if all the column values have the same time of day, or YYYYMMDDhhmmss if the times are different.</p> <p>Numeric columns will have member names be the column name followed by the numeric value – e.g., Product 1234</p> <p>Character columns will have the ID derived from the column content and limited to 25 characters.</p>
Xxxx Attribute *Xxxx Attribute	Column Xxxx will be treated as an attribute of a new attribute named Xxxx, with an ID of prefixXxxxx
DontImport	The column will be ignored
ImportBoth	The table will be imported as both a dimension and a fact table. Note that it must contain at least one column that has unique values, or the dimension import will fail
RootLevel	The column contains the ID or name of the root level of the dimension



The editable data source for the import rules will support expanding the list of rules to suit useful customization options as our experience with using AS models grows.

Using Import Rules to combine AS tables

In the previous example, there were three tables related to products and each table became a separate dimension. However, if you view the table relationships, each product has a single subcategory, and each subcategory has a single category. In CONTROL, the natural way to represent this is as a single dimension.

We created a second Power Pivot model and added the ImportDimensionRelated rule for the DimProduct table:

DatasourceColumns				
RuleNumber	ASColumnName	ControlSpecification	Tables	Comment
1	Date	Date IDandName		
2	*Date	*Date Attribute		
3	*LoadID	*LoadID Attribute		
4	StyleName	StyleName Attribute		
5	Size	Size Attribute		
6	*ID	* ID		
7	*Label	* ID		
8	*Number	* ID		
9	*Name	* Name		
10	*Description	*Description Attribute		
11	*Key	*Key IDandName		
12	*URL	DontImport		
13		ImportDimensionRelated	DimProduct	Imports all related tables

When the import is run, the resulting model has only one dimension related to product:



CONTROL®

Model - ContosoExample1 Copy

Available Dimensions

Used Dimensions

Scenario

Organization

No Scenario

AS14 Dates

AS14 DimProdu...

Variable

Time

AS14 Measures

No Time

And the dimension contains the subcategory and category information:

Sample Data In Use

Updating: Not Updateable filter: Level ID AS14PRODUCT

AS14PRODUCT_ID	AS14PRODUCT_Name	AS14PRODUCTKEY_ID	AS14PRODUCTKEY
0101001	Contoso 32GB MP3 Player E11 Silver	P1	ProductKey 1
0101002	Contoso 32GB MP3 Player E11 Blue	P2	ProductKey 2
0101003	Contoso 16 MP3 Player E100 White	P3	ProductKey 3
0101004	Contoso 2G MP3 Player E200 Silver	P4	ProductKey 4
0101005	Contoso 2G MP3 Player E200 Red	P5	ProductKey 5
0101006	Contoso 2G MP3 Player E200 Black	P6	ProductKey 6
0101007	Contoso 2G MP3 Player E200 Blue	P7	ProductKey 7
0101008	Contoso 4G MP3 Player E400 Silver	P8	ProductKey 8
0101009	Contoso 4G MP3 Player E400 Black	P9	ProductKey 9
0101010	Contoso 4G MP3 Player E400 Green	P10	ProductKey 10
0101011	Contoso 4G MP3 Player E400 Orange	P11	ProductKey 11
0101012	Contoso 4GB Flash MP3 Player E401 Blue	P12	ProductKey 12
0101013	Contoso 4GB Flash MP3 Player E401 Black	P13	ProductKey 13
0101014	Contoso 4GB Flash MP3 Player E401 Silver	P14	ProductKey 14
0101015	Contoso 4GB Flash MP3 Player E401 White	P15	ProductKey 15
0101016	Contoso 8GB Super-Slim MP3/Video Player M800 White	P16	ProductKey 16
0101017	Contoso 8GB Super-Slim MP3/Video Player M800 Red	P17	ProductKey 17
0101018	Contoso 8GB Super-Slim MP3/Video Player M800 Green	P18	ProductKey 18
0101019	Contoso 8GB Super-Slim MP3/Video Player M800 Pink	P19	ProductKey 19
0101020	Contoso 8GB MP3 Player new model M820 Black	P20	ProductKey 20
0101021	Contoso 8GB MP3 Player new model M820 Blue	P21	ProductKey 21
0101022	Contoso 8GB MP3 Player new model M820 Yellow	P22	ProductKey 22
0101023	Contoso 8GB MP3 Player new model M820 White	P23	ProductKey 23
0101024	Contoso 16GB MP3 Player M1600 Blue	P24	ProductKey 24
0101025	Contoso 16GB MP3 Player M1600 Black	P25	ProductKey 25
0101026	Contoso 16GB MP3 Player M1600 Green	P26	ProductKey 26
0101027	Contoso 16GB MP3 Player M1600 White	P27	ProductKey 27
0101028	Contoso 16GB MP3 Player M1600 Red	P28	ProductKey 28
0101029	Contoso 32GB Video MP3 Player M3200 White	P29	ProductKey 29
0101030	Contoso 32GB Video MP3 Player M3200 Red	P30	ProductKey 30
0101031	Contoso 32GB Video MP3 Player M3200 Orange	P31	ProductKey 31
0101032	Contoso 32GB Video MP3 Player M3200 Pink	P32	ProductKey 32
0101033	Contoso 4GB Portable MP3 Player M4300 Black	P33	ProductKey 33
0101034	Contoso 4GB Portable MP3 Player M4300 White	P34	ProductKey 34
0101035	Contoso 4GB Portable MP3 Player M4300 Silver	P35	ProductKey 35
0101036	Contoso 8GB Clock & Radio MP3 Player M830 Yellow	P36	ProductKey 36
0101037	Contoso 8GB Clock & Radio MP3 Player M830 Silver	P37	ProductKey 37
0101038	Contoso 8GB Clock & Radio MP3 Player M830 Black	P38	ProductKey 38
0101039	Contoso 8GB Clock & Radio MP3 Player M830 Green	P39	ProductKey 39
0101040	Contoso 8GB Clock & Radio MP3 Player M830 Blue	P40	ProductKey 40
0101041	Contoso 16GB New Generation MP3 Player M1630 Silver	P41	ProductKey 41
0101042	Contoso 16GB New Generation MP3 Player M1630 White	P42	ProductKey 42
0101043	Contoso 16GB New Generation MP3 Player M1630 Black	P43	ProductKey 43
0101044	Contoso 16GB New Generation MP3 Player M1630 Blue	P44	ProductKey 44
0101045	Contoso 16GB New Generation MP3 Player M1630 Pink	P45	ProductKey 45
0101046	Contoso 32GB New Generation MP3 Player M3230 Silver	P46	ProductKey 46

Dimension - AS14 DimProductComplete

Levels | Branches | Hierarchy

Levels and Attributes

Dimension Structure

AS14 Style

AS14 Color

AS14 WeightMeasure

AS14 UnitMeasure

AS14 StockType

AS14 Manufacturer

AS14 ProductCategoryKey

AS14 ProductSubcategoryKey

AS14 Brand

AS14 Product

Level Details

Immediate update

Update

Maximize



CONTROL®

Customizing the AS Column Map

In the import process, the relationships between CONTROL objects and AS Tables is defined based on the import rules and saved in the model's AS Column Map, which is included in its edit book.

The map for the previous example is shown in this object view:

A	B	C	D	E	F	G	H	I	J
	ContosoExample1 Copy Prototype AS Column Map View Updating: Clear and Insert Filter:								
	SubClass	Dimension	LevelID	LevelName	LevelOrAttribute	ASTable	ASIDColumn	ASNameColumn	
Organization	AS14 DimProductComplete	AS14PRODUCT	AS14 Product	Level	DimProductComplete	[ProductLabel]	[ProductName]		
Organization	AS14 DimProductComplete	AS14PRODUCTKEY	AS14 ProductKey	Level	DimProductComplete	[ProductKey]	[ProductKey] PrefixedAndFormatted		
Organization	AS14 DimProductComplete	AS14PRODUCTDESCRIPTION	AS14 ProductDescription	Attribute	DimProductComplete	[ProductDescription]			
Organization	AS14 DimProductComplete	AS14CLASS	AS14 Class	Level	DimProductComplete	[ClassID]	[ClassName]		
Organization	AS14 DimProductComplete	AS14STYLE	AS14 Style	Level	DimProductComplete	[StyleID] NameToID			
Organization	AS14 DimProductComplete	AS14STYLENAME	AS14 StyleName	Attribute	DimProductComplete	[StyleName]			
Organization	AS14 DimProductComplete	AS14COLOR	AS14 Color	Level	DimProductComplete	[ColorID]	[ColorName]		
Organization	AS14 DimProductComplete	AS14SIZEUNITMEASURE	AS14 SizeUnitMeasure	Level	DimProductComplete	[SizeUnitMeasureID] NameToID			
Organization	AS14 DimProductComplete	AS14STOPSALEDATE	AS14 StopSaleDate	Attribute	DimProductComplete	[StopSaleDate] DateTimeTranslation			
Organization	AS14 DimProductComplete	AS14SIZE	AS14 Size	Attribute	DimProductComplete	[Size]			
Organization	AS14 DimProductComplete	AS14WEIGHTUNITMEASURE	AS14 WeightUnitMeasure	Level	DimProductComplete	[WeightUnitMeasureID] NameToID			
Organization	AS14 DimProductComplete	AS14UNITOFMEASURE	AS14 UnitOfMeasure	Level	DimProductComplete	[UnitOfMeasureID]	[UnitOfMeasureName]		
Organization	AS14 DimProductComplete	AS14STOCKTYPE	AS14 StockType	Level	DimProductComplete	[StockTypeID]	[StockTypeName]		
Organization	AS14 DimProductComplete	AS14AVAILABLEFORSALEDATE	AS14 AvailableForSaleDate	Attribute	DimProductComplete	[AvailableForSaleDate] DateTimeTranslation			
Organization	AS14 DimProductComplete	AS14ETLLOADID	AS14 ETLLoadID	Attribute	DimProductComplete	[ETLLoadID]			
Organization	AS14 DimProductComplete	AS14LOADDATE	AS14 LoadDate	Attribute	DimProductComplete	[LoadDate] DateTimeTranslation			
Organization	AS14 DimProductComplete	AS14UPDATEDATE	AS14 UpdateDate	Attribute	DimProductComplete	[UpdateDate] DateTimeTranslation			
Organization	AS14 DimProductComplete	AS14WEIGHT	AS14 Weight	Attribute	DimProductComplete	[Weight]			
Organization	AS14 DimProductComplete	AS14UNITCOST	AS14 UnitCost	Attribute	DimProductComplete	[UnitCost]			
Organization	AS14 DimProductComplete	AS14UNITPRICE	AS14 UnitPrice	Attribute	DimProductComplete	[UnitPrice]			
Organization	AS14 DimProductComplete	AS14PRODUCTSUBCATEGORYKEY	AS14 ProductSubcategoryKey	Level	DimProductComplete	[ProductSubcategoryKey]	[ProductSubcategoryKey] PrefixedAndFormatted		
Organization	AS14 DimProductComplete	AS14BRAND	AS14 Brand	Level	DimProductComplete		[BrandName]		
Organization	AS14 DimProductComplete	AS14MANUFACTURER	AS14 Manufacturer	Level	DimProductComplete	[Manufacturer] NameToID	[Manufacturer]		
Organization	AS14 DimProductComplete	AS14PRODUCTSUBCATEGORY	AS14 ProductSubcategory	Level	DimProductComplete	[ProductSubcategoryLabel]	[ProductSubcategoryName]		
Organization	AS14 DimProductComplete	AS14PRODUCTSUBCATEGORYDESCRIPTION	AS14 ProductSubcategoryDescription	Attribute	DimProductComplete	[ProductSubcategoryDescription]			
Organization	AS14 DimProductComplete	AS14PRODUCTCATEGORYKEY	AS14 ProductCategoryKey	Level	DimProductComplete	[ProductCategoryKey]	[ProductCategoryKey] PrefixedAndFormatted		
Organization	AS14 DimProductComplete	AS14PRODUCTCATEGORY	AS14 ProductCategory	Level	DimProductComplete	[ProductCategoryLabel]	[ProductCategoryName]		
Organization	AS14 DimProductComplete	AS14PRODUCTCATEGORYDESCRIPTION	AS14 ProductCategoryDescription	Attribute	DimProductComplete	[ProductCategoryDescription]			
Organization	AS14 DimProductComplete	AS14SIZERANGE	AS14 SizeRange	Level	DimProductComplete	[SizeRange] NameToID	[SizeRange]		
Organization	AS14 DimProductComplete	AS14DIMPRODUCTCOMPLTTOTAL	AS14 DimProductComplete Total	Level	DimProductComplete	***AddedTotal***			
Organization	AS14 DimProductComplete	AS14STATUS	AS14 Status	Level	DimProductComplete	[Status] NameToID	[Status]		
Scenario	No Scenario	NOSCENARIO	No Scenario	Level					
Time	AS14 Dates	AS14DATE	AS14 Date	Level	Dates	[Date] DateTimeTranslation	[Date] DateTimeTruncation		
Time	AS14 Dates	AS14MONTH	AS14 Month	Level	Dates	[MonthNumber]	[Month]		
Time	AS14 Dates	AS14YEAR	AS14 Year	Level	Dates	[Year]	[Year] PrefixedAndFormatted		
Variable	AS14 Measures	AS14DATESTOTAL	AS14 Dates Total	Level	Dates	***AddedTotal***			
Variable	AS14 Measures	AS14MEASUREDETAIL	AS14 Measure Detail	Level					
Variable	AS14 Measures	AS14MEASURESUMMARY	AS14 Measure Summary	Level					

The map contains a row for every level and attribute of the dimensions of the model, both imported dimensions and default dimensions (such as Scenario) required to make the model usable.

The content of each column of this view is described below:

Column	Meaning
SubClass	The subclass of the dimension and level
Dimension	The name of the CONTROL dimension
LevelID	ID of the CONTROL level or attribute
LevelName	Name of the CONTROL level or attribute
LevelOrAttribute	Object class for this row
ASTable	The table in the AS model that the information for this level or attribute is derived from.
	Different levels of the same CONTROL dimension
	Different levels of the same CONTROL dimension <u>may</u> be derived from different related AS tables

ASIDColumn	<p>The column of the AS table that the ID of the level or value of the attribute is derived from.</p> <p>The format of this column is: [ASColumnName] optionalModifier</p> <p>The optionalModifier is present when the contents of the column in the AS table is transformed in any way when imported into CONTROL. If the optionalModifier is missing, there is no transformation.</p> <p>If the string "****AddedTotal****" is present, it means that the level does not correspond to a column in the AS table but has been added to provide a total of all members of this level.</p> <p>If this column is blank, and there is a non-blank value of the ASNameColumn, the ID is created from the name using standard meta-data conversion rules.</p>
ASNameColumn	<p>The column of the AS table that the name of the level is derived from. The format of the content is identical to the ASIDColumn.</p> <p>This cell will be blank for attributes.</p>

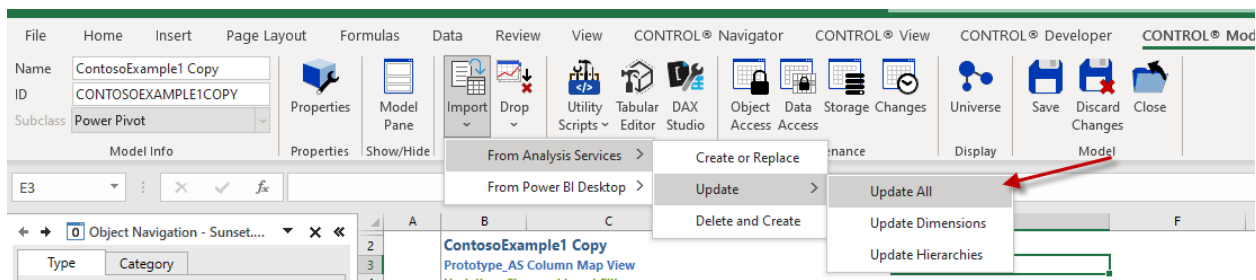
The supported values for the optional modifier are:

Modifier	Meaning
PrefixedAndFormatted	Used to create a readable name. For example, for a numeric column named Product with values 1, 2, 3 etc. the names would be "Product 1", "Product 2", etc.
DateTimeTranslation	Turns an AS date format column into an ID of the form YYYYMMDDhhmmss
DateTranslation	Turns an AS date format column into an ID of the form YYYYMMDD
DateTruncation	Turns an AS date format column into a name by dropping the time in the standard format
NameToID	Uses standard CONTROL logic to turn a string into an ID by removing special characters, making uppercase, and limiting to 25 characters
Truncation250	Converts to an attribute value, limiting strings to 250 characters
LeftOfSpace	Converts to an ID by returning values to the left of the first blank
RightOfSpace	Converts to a name by returning values to the right of the first blank

The AS Column Map is editable, if you want to modify and refine the choices that CONTROL has made in importing the meta-data:

- You may only add, delete, or modify rows associated with dimensions that are dedicated to this Power Pivot model
- All levels which are referenced must exist
- Any level or attribute may only be referenced once in a dimension
- There may be a maximum of one Added Total level
- The subclass of a dimension must be unique
- You may change the subclass of a dimension or level only if the dimension is dedicated to the model and the level is dedicated to the dimension
- You may change a level to an attribute or vice versa so long as it is dedicated
- You may change the AS Table, ASID Column and AS Name Column if the reference is valid and uses a supported modifier

If you choose to modify the AS Column Map, you will need to save your changes, then update the dimensions and hierarchies of the Power Pivot model. To do this, on the model ribbon select Import -> From Analysis Services -> Update -> Update All.



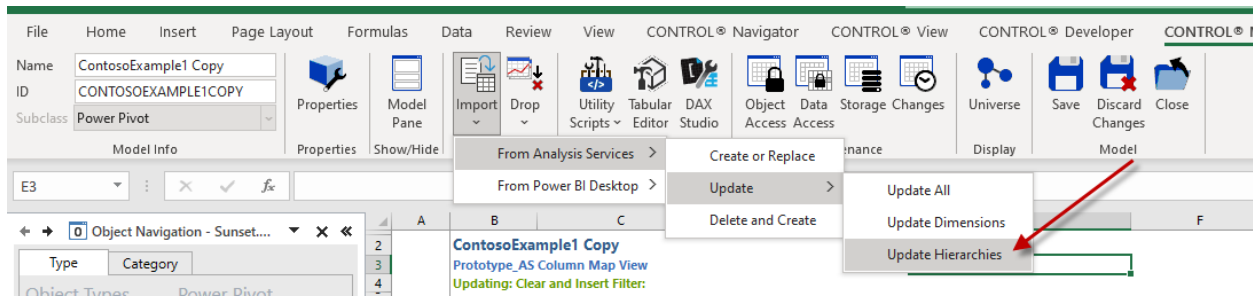
If you want to preview the changes to the dimension structure, choose Update Dimensions.

Customizing the Imported Dimensions

You can also refine the result of the import process by updating the dimension structures directly. Once the dimensions have been created, when you close and re-open the Power Pivot model's edit book, there will be a tab for each imported dimension.

You can perform limited types of reorganizations, including removing a level or attribute and changing reporting relationships. You may not add new levels unrelated to the underlying AS model.

Once you have made and saved your changes, you need to update the hierarchies. To do this, on the model ribbon select Import -> From Analysis Services -> Update -> Update Hierarchies.



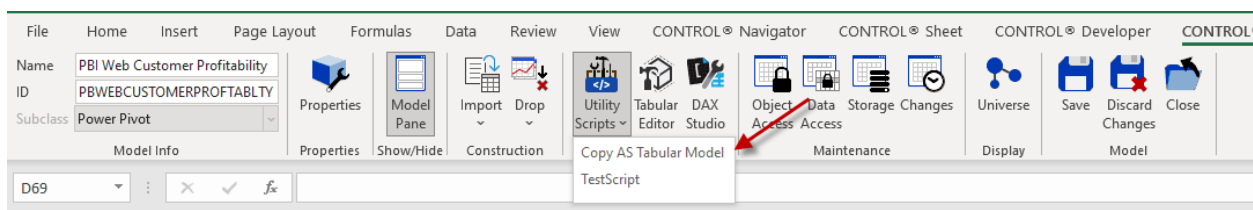
Copying AS Models

We have been using the term “AS Model” to refer to a tabular model running on a central Analysis Services instance. The heart of Analysis Services is the VertiPaq engine, which is an in-memory columnar database that is also used by Power BI on the desktop and in the cloud, and by Power Pivot in Excel.

For a CONTROL Power Pivot model to be usable by multiple users concurrently, it should be running on an Analysis Services instance, either on-premises or in the cloud.

So, there may be cases where you need an AS model that is in a Power BI file or on a different server, or you may want to make a copy of a model so you can add tables or make changes without impacting the original model.

To facilitate copying both the meta-data and data, there is a program script that should be defined to be a utility script:



The script allows you to select the source and target for the copy:

KKI Copy Tabular Model

This utility will make a copy of an existing Microsoft tabular model or update a previously created copy. The tabular model to be copied can exist in Analysis Services or in Power BI Desktop. The target of the copy always exists in Analysis Services.

Specify Source

- ☒ From existing Control Power Pivot model:
[Dropdown]
- ☐ Specified Analysis Services database:
AS Server: [Iron\Tabular]
AS Database: [Dropdown]
- ☐ Power BI Desktop:
Choose Active Instance... [Text Box]

Specify Target

- ☒ To existing Control Power Pivot model
[Dropdown]
- ☐ Specified Analysis Services database
AS Server: [Iron\Tabular]
AS Database: [Dropdown]

Updating Behavior

- ☒ Create
- ☐ Create or Replace
- ☐ Delete and Create

OK Cancel

Note:

1. If the source is a Power BI Desktop file, you must open the file in Power BI first
2. To access a Power BI premium file in the cloud, the source should be Specified Analysis Services database and the AS Server should look like:

powerbi://api.powerbi.com/v1.0/kcicorp.com/PG-Test
3. When the AS model is copied, the contents of the tables in the source files is written to staging relational tables in the CONTROL database. The data sources for the target AS model then point to those staging tables, not to the original sources of the data.
4. There is a confirmation prompt when you select Create or Replace and Delete and Create because the copy will overwrite the target model, so use those options carefully.

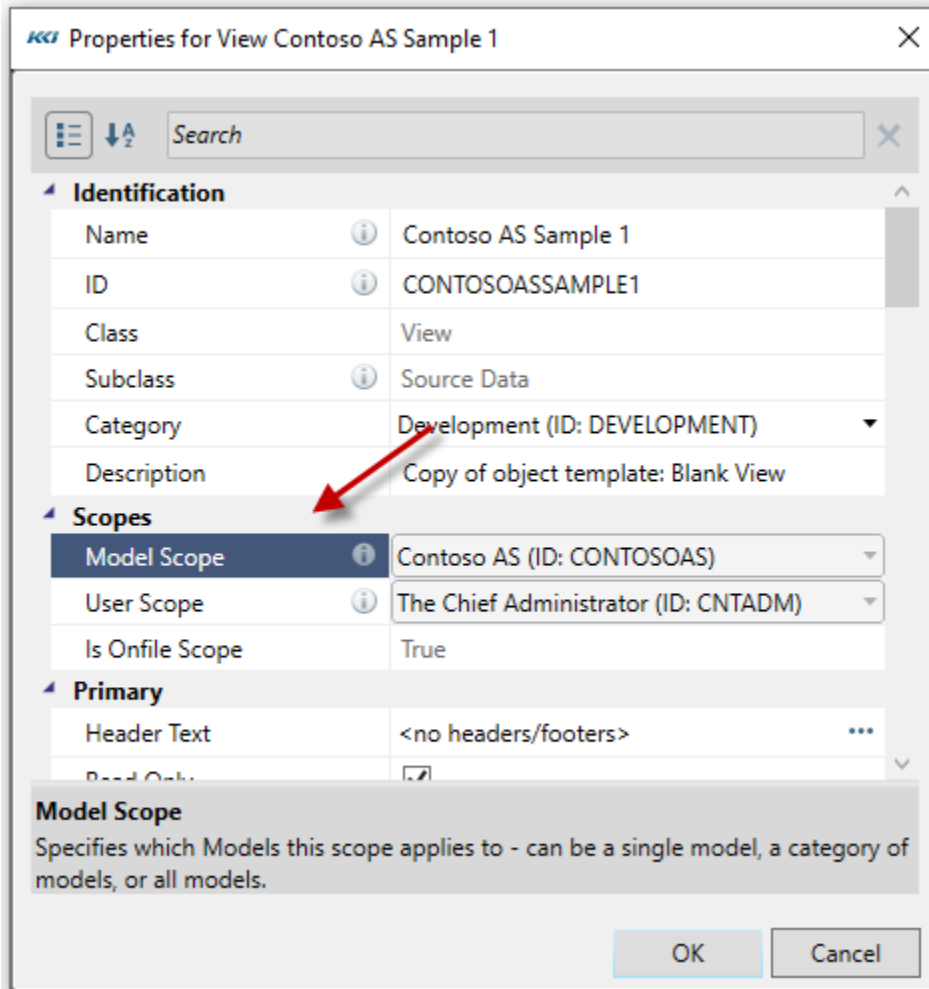
Views

Once you have exported a Control to AS Power Pivot model, or imported and AS to Control model, you are ready to leverage their content in the rich world of CONTROL views.

What is the same?

To an end-user, there is little perceivable difference between using a Power Pivot model and a computational model in a view.

Simply specify a Power Pivot model as the Model Scope property of the view and define the view dimensions, filters, branches, and options as usual:



The screenshot shows the 'Properties for View Contoso AS Sample 1' dialog box. The 'Scopes' section is expanded, and the 'Model Scope' dropdown is selected, showing 'Contoso AS (ID: CONTOSOAS)'. A red arrow points to the 'Model Scope' dropdown. The 'Identification' section shows the view name 'Contoso AS Sample 1' and its ID 'CONTOSOASSAMPLE1'. The 'Primary' section shows the header text '<no headers/footers>'. The 'Model Scope' section at the bottom explains that it specifies which models the scope applies to.

Identification	
Name	Contoso AS Sample 1
ID	CONTOSOASSAMPLE1
Class	View
Subclass	Source Data
Category	Development (ID: DEVELOPMENT)
Description	Copy of object template: Blank View

Scopes	
Model Scope	Contoso AS (ID: CONTOSOAS)
User Scope	The Chief Administrator (ID: CNTADM)
Is Onfile Scope	True

Primary	
Header Text	<no headers/footers>
Read Only	<input checked="" type="checkbox"/>

Model Scope
Specifies which Models this scope applies to - can be a single model, a category of models, or all models.

OK Cancel

You can use all the view styles, navigation mechanisms, template features, and access in flex views.

What is different?

There are some subtle but important differences to be aware of:

- Views on Power Pivot models are always read-only. You can however add comments, but the comments are saved in the CONTROL relational database – not in Analysis Services.
- All calculations are performed on the AS Server according to the definition of the measures and calculated columns. The aggregation and recalculation logic may therefore differ from what you are accustomed to seeing in models defined in CONTROL.



- If the AS Model has calculation groups, and a dimension that is associated with a calculation group is included in the view – there appear to be cases where the query fails due to limitations of the calculation group. In those cases, the values for the failed queries are set to 0.
- In the case of a model with multiple fact tables, there may be dimensions which have no relationship to one or more of the fact tables. Since view data must be assigned to a specific member of all the dimensions in the view, data for missing dimensions is assigned to the most summary member of that dimension.

COMPANY NAME Company Slogan							
Employee	Project Only	Office Expense	Supplies Expense-Office	Telephone Expense	Communications and IT	Rent or Lease Expense	Total Operating Expenses
Employee 1	Sales Office in Equador						50
	No Project						1,660
Employee 2	Engineering Plan A						400
	Engineering Plan B						440
	No Project						920
Employee 3	Engineering Plan B						1,920
Employee 4	No Project						2,160
Employee 5	No Project						1,920
Employee 6	No Project						1,920
Employee 7	No Project						1,920
Department 101	Engineering Plan A						400
	Engineering Plan B						2,360
	Sales Office in Equador						50
	No Project						10,500
Total Company JUMPSTART	Engineering Plan A	2,737	49	623	1,020	6,122	44,908
	Engineering Plan B						78,131
	Sales Office in Equador						3,195
	Test2 of pattern						66,084
	No Project	83,000	5,299	927	927		555,187

In the example above, variables such as Telephone Expense are not related to the Employee dimension, so are assigned to the Total Company member, whereas WF Hours are associated with individual employees.

Use in custom dimensions

In general, you can use Power Pivot models in views with custom dimensions exactly like you would use a Computational model, and you can combine both types of models too. Keep in mind that the data is read-only and is computed according to the logic in the AS model.


However, formulas for custom members are specified using CONTROL logic and are computed after data is queried from AS.



For example, if we wanted to compare the expenditure of resources (money and time) on projects related to product development vs. all other projects, we could create a custom dimension on model above, using base dimensions project and variable (measures):



CONTROL®

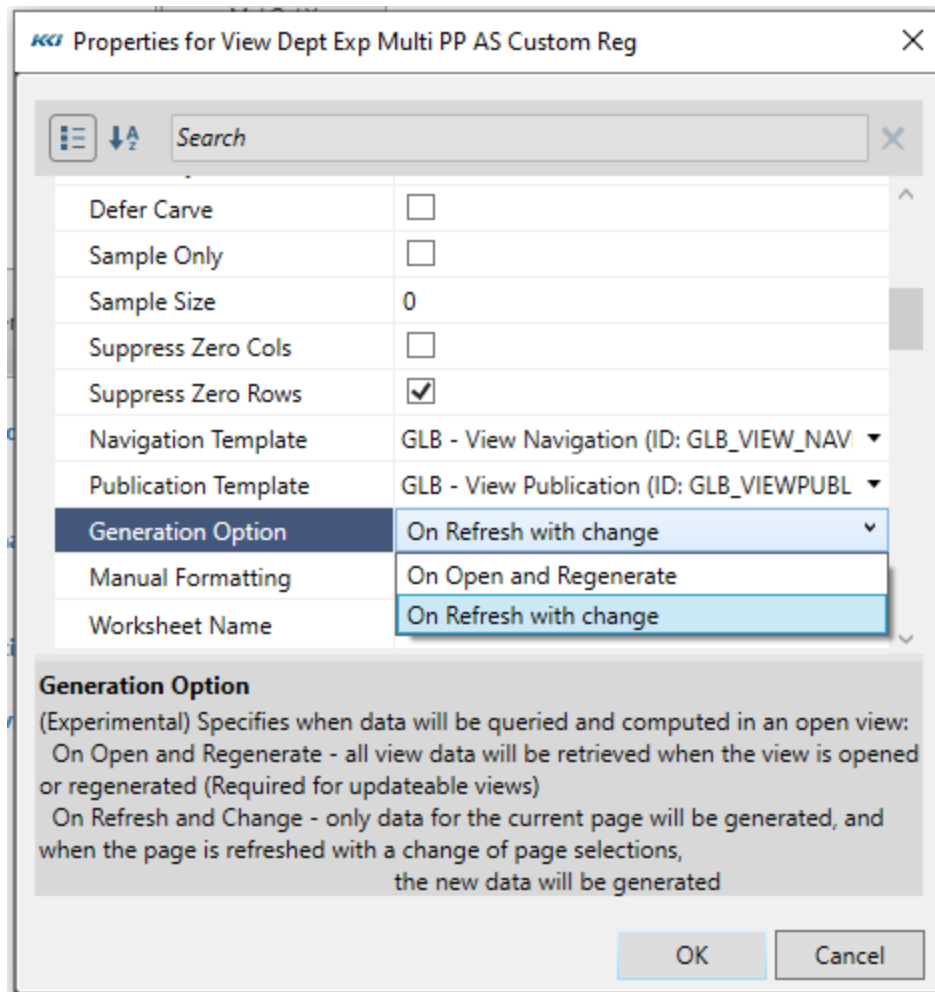
 COMPANY NAME Company Slogan												
Department	1Q09						2Q09					
	Direct Dollars	Direct Hours	Indirect Dollars	Indirect Hours	Direct/Indirect \$ %	Direct/Indirect Hrs %	Direct Dollars	Direct Hours	Indirect Dollars	Indirect Hours	Direct/Indirect \$ %	Direct/Indirect Hrs %
LA Manufacturing			212,633	2,420					197,572	3,730		
LA Operations			56,316	960					62,892	1,680		
LA Quality Control	178		15,798	320	1.13%		206		18,007	720	1.15%	
LA Inventory Control			4,185,344	320					3,189,809	560		
LA Transportation			19,209	320					21,976	560		
LA Accounting			50,783	640					51,848	1,120		
LA IT			36,221	960					41,546	1,680		
LA HR			39,130	640					45,914	1,120		
LA Sales			416,075	2,240					1,282,547	3,920		
LA Parts Sales & Service			42,845	640					50,029	1,120		
LA Admin			42,436	640					49,765	1,120		
LA Finance			542	-					503	-		
LA R&D			29,548	-					270	-		
Los Angeles Plant	178		5,146,878	10,100	0.00%		206		5,012,679	17,330	0.00%	
Gas Scooter Division	178		5,146,878	10,100	0.00%		206		5,012,679	17,330	0.00%	
Internet Sales - LA			651	-					215	-		

The Generation Option

In all releases prior to 10.6, when a view is opened, CONTROL retrieves and computes all data specified by its filter and branch definitions. For data stored in CONTROL's relational tables this was determined to be the most efficient process.

Based on the fundamentally different performance profile of AS queries, we have introduced the option of only retrieving the data on the visible page of the view. As you navigate the view – changing pages or rotating dimensions, the data for the new page is retrieved.

The choice is a scoped option of the view and defaults to the pre-10.6 behavior:



The advantages of "On Refresh with Change" are:

- The time to open the view is reduced
- The memory consumed by the CONTROL engine process may be significantly smaller, and this might be important if you have a lot of users running on a single, memory constrained server

The new option is available for all computational views, irrespective of model type, with the following restrictions:

- The new option is not supported for flex views. (Flex views are not constrained to present data from the "current" page.)
- The view must have the Read Only option selected.
- Defer carve does not work with this option, so if Defer Carve is TRUE, the view is set to Sample Only, and you must unselect Sample Only on the ribbon to generate the view.



Use in CONTROL Web

Computational views on Power Pivot models are usable on CONTROL Web without restriction, other than the usual object access, scope, and privacy considerations.

For Computational views on Power Pivot models to function correctly, your administrator needs to install the required Analysis Services components on the CONTROL Web server so that the CONTROL Web server can communicate with any Analysis Server instances referenced by the views that are exposed in the application menu. Please refer to the CONTROL Setup Guide or the CONTROL readme file for instructions on installing the required Analysis Services components.

Leveraging and Extending

The variable hierarchy

One of the most powerful features of Power Pivot models is the ability to add new calculations, including complex cross-dimensional formulas at any time.

The Formula Type for the variable hierarchy of your Power Pivot model is "DAX", and the formulas are stored in the Direct Logic field in the variable hierarchy.

You can add new members to the variable hierarchy using the hierarchy object view or the hierarchy task pane, remembering to use DAX syntax to specify the formula.

AS11 Measures (Hierarchy)				
Dynamic Rollups View				
Updating: Clear and Insert Rollups, Merge Members Filter: ALL				
AS11MEASURES_COAPARENT_ID	AS11MEASURES_COADETAIL_ID	Name	Direct Logic	
ASMODELMEASURES	SUMOFSALESAMOUNT	Sum of SalesAmount	FactSalesSmall:[Sum of SalesAmount]	
ASMODELMEASURES	SUMOFTOTALCOST	Sum of TotalCost	FactSalesSmall:[Sum of TotalCost]	
ASMODELMEASURES	SUMOFMARGIN	Sum of Margin	FactSalesSmall:[Sum of Margin]	
ASMODELMEASURES	UNITCOST	UnitCost	FactSalesSmall:SUM('FactSalesSmall'[UnitCost])	
ASMODELMEASURES	UNITPRICE	UnitPrice	FactSalesSmall:SUM('FactSalesSmall'[UnitPrice])	
ASMODELMEASURES	SALESQUANTITY	SalesQuantity	FactSalesSmall:SUM('FactSalesSmall'[SalesQuantity])	
ASMODELMEASURES	RETURNQUANTITY	ReturnQuantity	FactSalesSmall:SUM('FactSalesSmall'[ReturnQuantity])	
ASMODELMEASURES	RETURNAMOUNT	ReturnAmount	FactSalesSmall:SUM('FactSalesSmall'[ReturnAmount])	
ASMODELMEASURES	DISCOUNTQUANTITY	DiscountQuantity	FactSalesSmall:SUM('FactSalesSmall'[DiscountQuantity])	
ASMODELMEASURES	DISCOUNTAMOUNT	DiscountAmount	FactSalesSmall:SUM('FactSalesSmall'[DiscountAmount])	
ASMODELMEASURES	TOTALCOST	TotalCost	FactSalesSmall:SUM('FactSalesSmall'[TotalCost])	
ASMODELMEASURES	SALESAMOUNT	SalesAmount	FactSalesSmall:SUM('FactSalesSmall'[SalesAmount])	
ASMODELMEASURES	MARGIN	Margin	FactSalesSmall:SUM('FactSalesSmall'[Margin])	
ASMODELMEASURES		AS Model Measures		
ADDEDMEASURES	TESTNEW	Margin Pct	FactSalesSmall:DIVIDE(SUM('FactSalesSmall'[Margin]),SUM('FactSalesSmall'[SalesAmount]))	
ADDEDMEASURES	TESTNEW2	Number of Sales	FactSalesSmall:COUNTRROWS('FactSalesSmall')	
ADDEDMEASURES	TESTNEW4	Total Number of Sales	COUNTRROWS(ALL('FactSalesSmall'))	
ADDEDMEASURES	TESTNEW3	Average Sale	FactSalesSmall:DIVIDE(SUM('FactSalesSmall'[SalesAmount]),COUNTRROWS('FactSalesSmall'))	
ADDEDMEASURES		Added Measures		

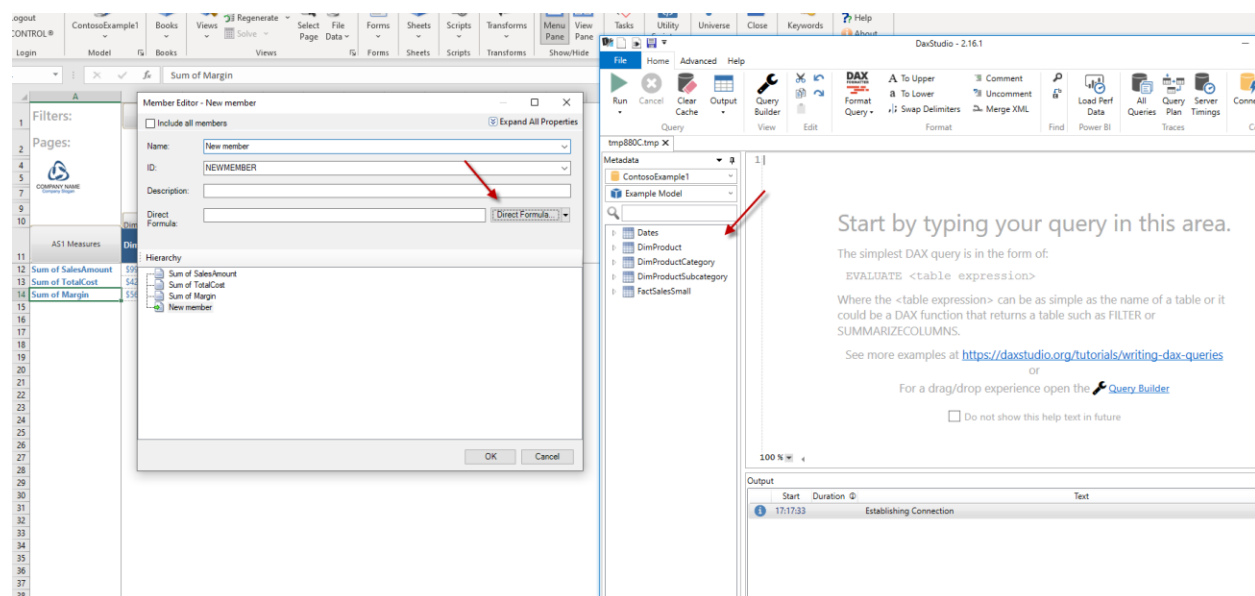
Note:

- The DAX formula should be preceded with the name of the fact table to be queried, followed by a colon. This prefix can be omitted if there is only one fact table associated with the model.

- The entire Direct Logic formula should be preceded by a single quote to prevent CONTROL from attempting to interpret it as a CONTROL formula.
- We recommend that you distinguish between measures that are imported from the AS model be distinguished from those added in the hierarchy, such as by a different parent member (Added Measures).

Base members

You can also add new calculated members from within a view by using the "Create base member.." option on the right click menu when you have selected a cell in the variable dimension.



Clicking on the Direct Formula button will launch the DAX Studio editor, with the current model selected, to make it easier to construct DAX expressions.

Note that you can only reference columns and measures that are part of the AS model, so unlike CONTROL formulas, you cannot refer to a measure you have added in CONTROL.

Ad hoc members

Similarly, from within a view you can create ad hoc member, also specified by a DAX formula. Ad hoc members are stored in an affiliated custom dimension. An ad hoc member may be private to a specific user or group, but in this context, process no differently from base members.



Mappings

A significant benefit of connecting to an AS model is to integrate its data and meta-data with your existing CONTROL applications and to leverage CONTROL features and functions with your externally managed models.

To achieve this integration, mappings have been extended to handle several likely scenarios.

Meta-data

Since the import process creates dimensions, hierarchies, levels, and attributes that are undifferentiated from the objects that are built manually or via external mappings, there are minimal constraints on how you use these objects.

You can:

- Make copies.
- Re-use the dimensions and hierarchies in new models and use the levels and attributes in other dimensions.
- Copy and modify the hierarchies or create subset hierarchies.
- Map from the levels, attributes, and hierarchies to other CONTROL objects or datasources. You can map to these objects, but the changes may not be reflected in views because those changes are not propagated to the underlying AS model.

The meta-data objects created by the import process are dedicated to the Power Pivot model. This allows them to be dropped and recreated easily if you want to iteratively refine the import rules and be dropped when the model is deleted. If you choose to make any object reusable, you will need to delete those objects manually.

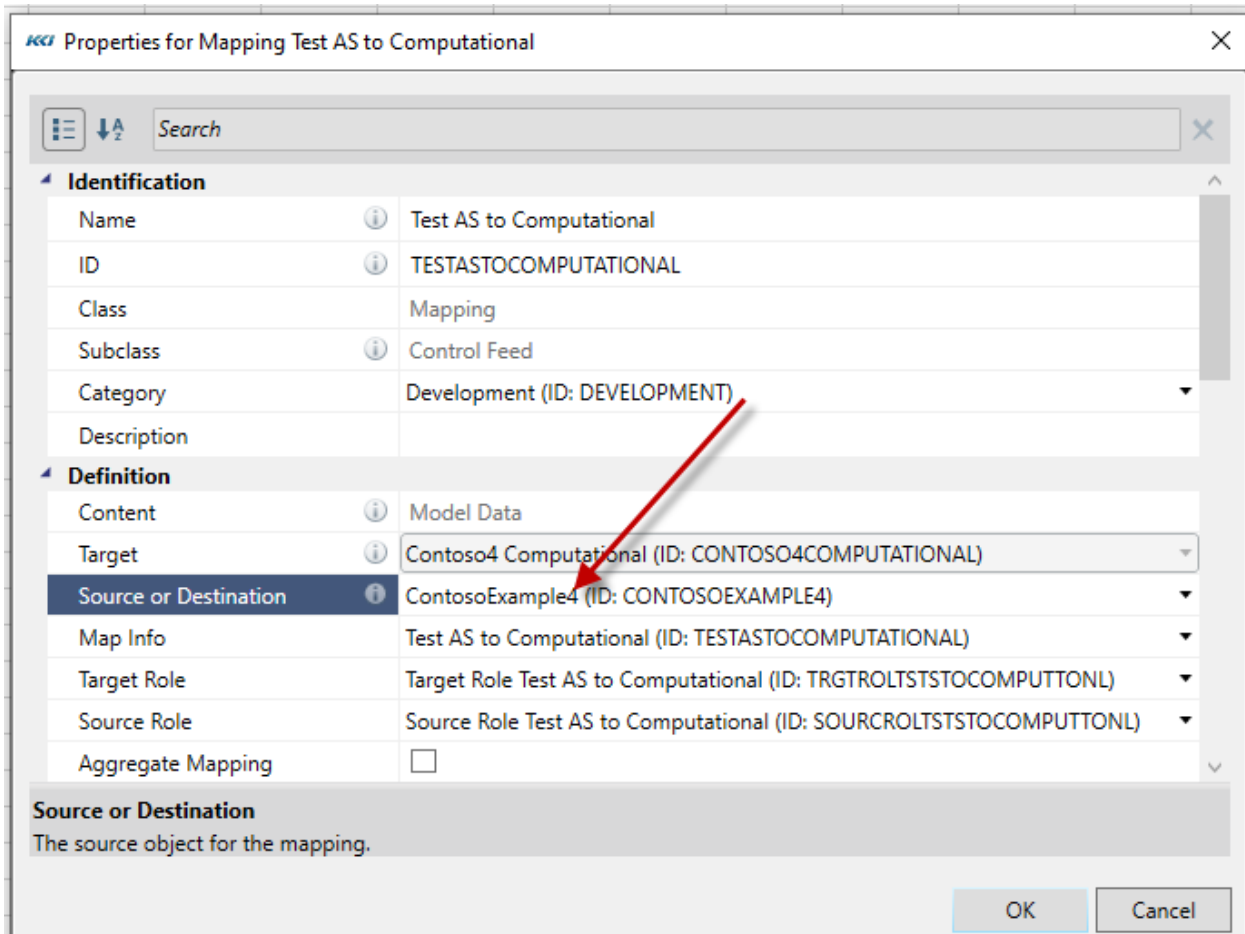
On-demand data - Power Pivot to Computational models

The import process re-creates an AS model's meta-data as CONTROL objects but does not duplicate the data in the fact tables. That data is queried dynamically (using DAX) directly from AS.

You may want to be able to use the data in a computational model because:

- You would like to update or manipulate it using transforms for "what-if" analyses
- You would like to enhance it with data from other sources.
- There are CONTROL calculations you would like to add
- You want to add new levels or attributes so you can slice and dice differently
- You need the data in relational tables for other purposes

The process of creating and executing the mapping only differ in that a Power Pivot model is selected as the source, rather than a computational model:



Properties for Mapping Test AS to Computational

Search

Identification

Name	Test AS to Computational
ID	TESTASTOCOMPUTATIONAL
Class	Mapping
Subclass	Control Feed
Category	Development (ID: DEVELOPMENT)
Description	

Definition

Content	Model Data
Target	Contoso4 Computational (ID: CONTOSO4COMPUTATIONAL)
Source or Destination	ContosoExample4 (ID: CONTOSOEXAMPLE4)
Map Info	Test AS to Computational (ID: TESTASTOCOMPUTATIONAL)
Target Role	Target Role Test AS to Computational (ID: TRGTROLTSTSTOCOMPUTONL)
Source Role	Source Role Test AS to Computational (ID: SOURCROLTSTSTOCOMPUTONL)
Aggregate Mapping	<input type="checkbox"/>

Source or Destination
The source object for the mapping.

OK Cancel

The map associations are initialized using the same dimension and level matching rules as between computational models. In this example, the computational model (target) was defined with the same dimensions as the Power Pivot model (source).

D Mapping - Test AS to Computational
X »

▼ Choose Aspect Types

To map aspects, drag aspect from the tree to the worksheet.

Aspects	Source
<ul style="list-style-type: none"> Contoso4 Computational <ul style="list-style-type: none"> No Scenario (Scenario Dimension) <div>No Scenario (Scenario Level)No Scenario (Scenario Level)</div> AS11 DimProduct (Organization Dimension) <ul style="list-style-type: none"> AS11 ProductSubcategory (Organization Level) AS11 ProductCategoryKey (Organization Level) AS11 ProductCategory (Organization Level) AS11 Manufacturer (Organization Level) AS11 Brand (Organization Level) AS11 Status (Organization Level) AS11 ProductSubcategoryKey (Organization Level) AS11 StockType (Organization Level) AS11 UnitOfMeasure (Organization Level) AS11 WeightUnitMeasure (Organization Level) AS11 DimProduct Total (Organization Level) AS11 ProductURL (Organization Level) AS11 ImageURL (Organization Level) AS11 SizeRange (Organization Level) AS11 SizeUnitMeasure (Organization Level) AS11 Color (Organization Level) AS11 Style (Organization Level) AS11 Class (Organization Level) AS11 ProductKey (Organization Level) 	
AS11 Product (Organization Level)	AS11 Product (Organization Level)
AS11 Measures (Variable Dimension)	AS11 Measures (Variable Dimension)
<ul style="list-style-type: none"> AS11 Dates (Time Dimension) <ul style="list-style-type: none"> AS11 Dates Total (Time Level) AS11 Year (Time Level) AS11 Month (Time Level) 	
AS11 Date (Time Level)	AS11 Date (Time Level)

Aspect Properties
⌆

☐ Immediate update

Update
Maximize



It is critical to point out that the variable hierarchy used in the target model should have its Formula Type property set to CONTROL, as all calculations, including organization and time logic need to be specified using CONTROL syntax.

Dynamic data - Power Pivot to Computational models

In general, a dynamic mapping from a Power Pivot model works exactly like an on-demand mapping, executed when a view is generated. That means that the data is mapped from the AS model at the level of the associations for each dimension, and then CONTROL formulas and hierarchies are used to compute other calculated variables and aggregations.

There is one important exception to this behavior when an entire scenario of the target model is mapped. In this case, ALL values are computed according to the logic in the AS model and then mapped to the target. This allows you to leverage performance or computational characteristics of AS in scenarios that may be static or slowly changing.

Here is an example of the Actual scenario being mapped from an AS model – the associations:

D Mapping - Test AS to Comp Dynamic X »

▼ Choose Aspect Types

To map aspects, drag aspect from the tree to the worksheet.

Aspects	Source
<ul style="list-style-type: none"> ▲ Contoso4 Comp Dynamic <ul style="list-style-type: none"> ▲ Scenarios (Scenario Dimension) <ul style="list-style-type: none"> ▶ Home Currency (Scenario Level) ▲ Scenarios (Scenario Level) <ul style="list-style-type: none"> ▲ Scenarios Members 	
Actual	NOSCENARIO: None (Member)
ANNUALBUD: Annual Budget	
CURR_FCST: Current Forecast	
CURR_BUD: Current Budget	
ACTBWBUD: Actual B/(W) Budget	
FCSTBWBUD: Fcst B/(W) Budget	
FCST_200802: February 2008 Forecast	
FCST_200803: March 2008 Forecast	
FCST_200804: April 2008 Forecast	
FCST_200805: May 2008 Forecast	
BUD_2007: 2007 Budget	
BUD_2008: 2008 Budget	
BUD_2009: 2009 Budget	
BUD_2010: 2010 Budget	
What If 1	
What If 2	
Comments	
STEP: Long Range Plan (STEP)	
VARTOSTEP: Budget Var to STEP	
▶ AS11 DimProduct (Organization Dimension)	AS11 DimProduct (Organization Dimension)
▶ AS11 Measures (Variable Dimension)	AS11 Measures (Variable Dimension)
▶ AS11 Dates (Time Dimension)	AS11 Dates (Time Dimension)

Aspect Properties ⤴



The target data access role:



Formula Bar

D Role - Target Role Test AS to Comp Dynamic ✕ »

Privileges Usage

Dimension Selection ☐ ☐ ☐ Filters

Scenario	Organization
 Scenarios	 AS11 DimProduct

Variable	Time
 AS11 Measures	 AS11 Dates

Dimension	Read Filters	Write Filters
Scenarios	No filter	=SCENARIOS ACTUAL

And a view on the model:



	A	B	C	D	E	F	G	H	I	J	K	L
1	Filters:											
2	Pages:	AS11 Class (Total DimProdu.)										
4												
5												
7												
9												
10		AS11 Date	Scenarios									
11	AS11 Measures	Jan 2007	Feb 2007	Mar 2007	Apr 2007	May 2007	Jun 2					
12		Actual	Current Budget	Actual	Current Budget	Actual	Current Budget	Actual	Current Budget	Actual	Current Budget	Actual
13	Sum of SalesAmount	\$2,219,245.93		\$2,385,321.00		\$2,323,917.02		\$3,044,274.29		\$3,444,693.63		\$2,969,649.52
14	Sum of TotalCost	\$964,780.55		\$1,039,009.00		\$1,031,028.00		\$1,333,673.21		\$1,479,194.70		\$1,277,945.15
15	Sum of Margin	\$1,254,465.38		\$1,346,312.00		\$1,292,889.02		\$1,710,601.08		\$1,965,498.93		\$1,691,704.37
16	UnitCost	\$93,710.70		\$104,292.10		\$114,850.03		\$130,923.86		\$142,540.20		\$124,401.52
17	UnitPrice	\$215,834.50		\$235,941.28		\$257,239.70		\$292,866.88		\$326,038.76		\$284,259.18
18	SalesQuantity	10,263	-	8,797	-	9,761	-	12,284	-	12,492	-	12,346
19	ReturnQuantity	131	-	125	-	131	-	174	-	171	-	160
20	ReturnAmount	\$34,646.18		\$31,992.89		\$34,791.79		\$46,188.50		\$53,802.14		\$43,215.52
21	DiscountQuantity	2,147	-	2,025	-	2,228	-	307	-	372	-	350
22	DiscountAmount	\$47,344.26		\$41,664.73		\$47,076.11		\$15,720.54		\$8,310.09		\$9,467.18
23	TotalCost	\$964,780.55		\$1,039,009.00		\$1,031,028.00		\$1,333,673.21		\$1,479,194.70		\$1,277,945.15
24	SalesAmount	\$2,219,245.93		\$2,385,321.00		\$2,323,917.02		\$3,044,274.29		\$3,444,693.63		\$2,969,649.52
25	Margin	\$1,254,465.38		\$1,346,312.00		\$1,292,889.02		\$1,710,601.08		\$1,965,498.93		\$1,691,704.37
26	-AS Model Measures	-	-	-	-	-	-	-	-	-	-	-
27	Margin Pct	56.5%		56.4%		55.6%		56.2%		57.1%		57.0%
28	Number of Sales	899		831		948		1,118		1,104		1,071
29	Total Number of Sales	27,279		27,279		27,279		27,279		27,279		27,279
30	Average Sale	\$2,468.57		\$2,870.42		\$2,451.39		\$2,722.96		\$3,120.19		\$2,772.78
31	-Added Measures	-	-	-	-	-	-	-	-	-	-	-
32												
33												
34												

Dynamic data - Power Pivot to Power Pivot models

Suppose you want to manage all data for a model in AS, but want to use scenario, time, and variable hierarchies that are built and maintained for your other CONTROL applications. You may have filters, branches, and custom dimensions that your user community is familiar and comfortable with.

To satisfy this requirement, you can build a Power Pivot model (AS to Control), and rather than importing the meta-data, simply assign the dimensions and hierarchies you want to use.

In this example, we will use our common scenario and time dimensions, and the imported organization and variable dimension.

We then create a dynamic mapping from the imported model. Here are the associations:

D Mapping - ContosoDynamic

Choose Aspect Types

To map aspects, drag aspect from the tree to the worksheet.

Aspects	Source
<div> <div>ContosoDynamic</div> <div> <div>Scenarios (Scenario Dimension)</div> <div> <div>Home Currency (Scenario Level)</div> <div> <div>Scenarios (Scenario Level)</div> <div> <div>Scenarios Members</div> <div>Actual</div> <div>NOSCENARIO: None (Member)</div> <div>ANNUALBUD: Annual Budget</div> <div>CURR_FCST: Current Forecast</div> <div>CURR_BUD: Current Budget</div> <div>ACTBWBUD: Actual B/(W) Budget</div> <div>FCSTBWBUD: Fcst B/(W) Budget</div> <div>FCST_200802: February 2008 Forecast</div> <div>FCST_200803: March 2008 Forecast</div> <div>FCST_200804: April 2008 Forecast</div> <div>FCST_200805: May 2008 Forecast</div> <div>BUD_2007: 2007 Budget</div> <div>BUD_2008: 2008 Budget</div> <div>BUD_2009: 2009 Budget</div> <div>BUD_2010: 2010 Budget</div> <div>What If 1</div> <div>What If 2</div> <div>Comments</div> <div>STEP: Long Range Plan (STEP)</div> <div>VARTOSTEP: Budget Var to STEP</div> </div> </div> </div> </div> </div>	

AS11 DimProduct (Organization Dimension)

AS11 DimProduct (Organization Dimension)

AS11 Measures (Variable Dimension)

AS11 Measures (Variable Dimension)

Time (Time Dimension)

Years (Time Level)

AS11 Year (Time Level)

Quarters (Time Level)

Months (Time Level)

AS11 Month (Time Level)

Aspect Properties


☐ Immediate update

Update

Maximize

Here is the resulting view:



Filters:	Scenarios (Actual)																	
Pages:	AS11 Class (Total DimProdu.)																	
 COMPANY NAME Company Slogan																		
M / Q / Y																		
AS11 Measures		Jan 2008	Feb 2008	Mar 2008	-1Q08	Apr 2008	May 2008	Jun 2008	-2Q08	Jul 2008	Aug 2008	Sep 2008	-3Q08	Oct 2008	Nov 2008	Dec 2008	-4Q08	Year 2008
Sum of SalesAmount		\$2,200,426.51	\$2,375,629.06	\$2,380,814.63		\$2,882,308.23	\$2,576,127.41	\$3,083,561.05		\$3,109,553.72	\$2,617,262.96	\$3,096,309.98		\$2,649,897.48	\$3,185,936.77	\$3,187,378.11		\$33,345,205.91
Sum of TotalCost		\$976,024.90	\$1,036,405.80	\$1,032,579.18		\$1,222,191.67	\$1,132,427.42	\$1,300,833.17		\$1,344,066.13	\$1,148,443.18	\$1,329,328.12		\$1,126,717.60	\$1,392,540.63	\$1,349,580.95		\$14,391,138.75
Sum of Margin		\$1,224,401.61	\$1,339,223.26	\$1,348,235.45		\$1,660,116.56	\$1,443,699.99	\$1,782,727.88		\$1,765,487.59	\$1,468,819.78	\$1,766,981.86		\$1,523,179.88	\$1,793,396.14	\$1,837,797.16		\$18,954,067.16
UnitCost		\$91,115.76	\$86,683.06	\$98,069.03		\$106,674.91	\$99,801.27	\$110,032.45		\$98,030.53	\$99,781.65	\$107,025.98		\$100,633.66	\$102,310.67	\$97,319.10		\$1,197,478.07
UnitPrice		\$207,431.56	\$201,788.51	\$228,099.67		\$253,402.07	\$224,865.11	\$259,553.99		\$226,293.92	\$229,141.05	\$249,433.01		\$236,249.64	\$240,710.59	\$231,630.97		\$2,788,600.08
SalesQuantity		9,654	10,332	9,597	-	10,385	14,808	10,251	-	11,943	10,898	10,872	-	9,758	12,390	12,999	-	133,887
ReturnQuantity		91	90	89	-	92	129	106	-	93	99	105	-	106	100	103	-	1,203
ReturnAmount		\$26,929.80	\$27,058.99	\$28,285.59	-	\$31,454.92	\$40,834.48	\$31,385.56	-	\$25,733.51	\$28,233.55	\$31,636.09	-	\$32,279.27	\$30,107.92	\$44,375.22	-	\$378,314.90
DiscountQuantity		1,456	1,541	1,578	-	469	375	421	-	902	1,201	1,312	-	299	1,671	1,695	-	12,920
DiscountAmount		\$43,792.54	\$45,968.70	\$48,305.77	-	\$35,887.48	\$10,661.65	\$13,014.61	-	\$34,320.94	\$37,035.61	\$41,718.87	-	\$18,707.90	\$100,566.92	\$100,070.17	-	\$530,051.16
TotalCost		\$976,024.90	\$1,036,405.80	\$1,032,579.18	-	\$1,222,191.67	\$1,132,427.42	\$1,300,833.17	-	\$1,344,066.13	\$1,148,443.18	\$1,329,328.12	-	\$1,126,717.60	\$1,392,540.63	\$1,349,580.95	-	\$14,391,138.75
SalesAmount		\$2,200,426.51	\$2,375,629.06	\$2,380,814.63	-	\$2,882,308.23	\$2,576,127.41	\$3,083,561.05	-	\$3,109,553.72	\$2,617,262.96	\$3,096,309.98	-	\$2,649,897.48	\$3,185,936.77	\$3,187,378.11	-	\$33,345,205.91
Margin		\$1,224,401.61	\$1,339,223.26	\$1,348,235.45	-	\$1,660,116.56	\$1,443,699.99	\$1,782,727.88	-	\$1,765,487.59	\$1,468,819.78	\$1,766,981.86	-	\$1,523,179.88	\$1,793,396.14	\$1,837,797.16	-	\$18,954,067.16
-AS Model Measures																		
Margin Pct		55.6%	56.4%	56.6%	-	57.6%	56.0%	57.8%	-	56.8%	56.1%	57.1%	-	57.5%	56.3%	57.7%	-	56.8%
Number of Sales		670	649	672	-	748	748	724	-	671	701	745	-	707	716	704	-	8,455
Total Number of Sales		27,279	27,279	27,279	-	27,279	27,279	27,279	-	27,279	27,279	27,279	-	27,279	27,279	27,279	-	27,279
Average Sale		\$3,284.22	\$3,660.45	\$3,542.88	-	\$3,853.35	\$3,444.02	\$4,259.06	-	\$4,634.21	\$3,733.61	\$4,156.12	-	\$3,748.09	\$4,449.63	\$4,527.53	-	\$3,943.84
-Added Measures		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

This example points to something to be mindful of - our AS model does not have a column defined in its Dates table for Quarter. Since all computations and aggregations are performed in AS, and there is no definition of how to compute a quarterly total, the quarter columns are blank. (You can fix this problem by putting an appropriate DAX expression for quarters in the source model's AS Column Map.)

An Architectural recommendation

The possibilities for how to incorporate interesting AS data into your CONTROL world are numerous and go far beyond the simple examples mentioned above.

We strongly suggest that you maintain a "pure" imported model that you can reconcile with data you see in Power BI, SSRS, and Excel pivot tables. Use mappings of the various flavors to other models – computational and Power Pivot – to achieve your reporting and analytic ends.

Since there is no data imported to a Power Pivot model, they do not use much storage or impact performance.

You can then use views, drill to source, and other CONTROL features when needed to reconcile and troubleshoot your applications.

AS Query Data Sources

The tables in an AS model can be queried in much the same way relational tables can be queried using SQL. A big difference is that the query language (DAX) is far more capable of performing useful cross-dimensional calculations and naturally follows table relationships.

Data source properties

To define an AS Query datasource, there are just a few properties that differ from other external sources:

KKI Properties for Datasource AS Tabular Tester

Search

Identification

Name	AS Tabular Tester
ID	ASTABULARTESTER
Class	Datasource
Subclass	AS Query
Category	Development (ID: DEVELOPMENT)
Description	

Definition

Data Base	Bells.Covid 19 Full
Data Table	CNTADM.DSrcE_ASTABULARTESTER
AS Query	EVALUATE 'Covid 19 Geography' ...
Table Creation	Create On Demand

Content

Has Access Roles	<input type="checkbox"/>
Has Attribute Values	<input checked="" type="checkbox"/>
Has Codes	<input type="checkbox"/>
Has Hierarchy Rollups	<input checked="" type="checkbox"/>
Has Level Members	<input checked="" type="checkbox"/>
Has Model Data	<input checked="" type="checkbox"/>
Has User Info	<input type="checkbox"/>
Analysis Services Usage	Not used for AS

Data Base

For relational sources: Enter a database connection string here. This is the same string you would enter in the Database field of the Login dialog. You can leave this text-box empty if the datasource is a table in the current CONTROL database.

For Files, enter the completely qualified file name.

For Excel datasources, enter the workbook. Note that .xls files currently must be re-saved as .xlsx to be used as a data source.

For AS Query datasources, enter Catalog or Server.Catalog where the Catalog equates to the data base for AS Tabular.

If the Server is not supplied, the replacement value of the &KKI_ASSEServer keyword is used to identify the server.

OK Cancel

Property	Meaning
Subclass	AS Query
Data Base	The name of the AS Server followed by a period and the name of the AS database. You can omit the server to use the default AS server
AS Query	The DAX expression that will return a table with the desired results. DAX queries generally begin with "EVALUATE ..." but may have variable definition statements prior to the EVALUATE expression. Pressing the "..." symbol on the right will launch DAX Studio in the context of the selected data base.



Table Creation	Generally, set to Do not create, unless you want the results of your query to be copied into a relational table for other purposes, such as a data mapping.
Analysis Services Usage	The setting of this property determines which column properties will be visible by default in the data source columns object view. Select Not used in AS unless you will be using the datasource in a CONTROL to AS Power Pivot model.

Use in Mappings

Once you have created an AS query datasource, it can be used as the source for the following types of mappings:

- Meta-data mappings to:
 - Levels
 - Hierarchies
 - Codes
- Data mappings to computational models. Note the data source must have the table creation property set to Create on demand or Create dynamically
- External to external mappings

Source Data Views

The AS query data source can be opened from the user navigation pane to see the contents of the query:

Column1	ProductKey	ProductLabel	ProductName	ProductDescription
1	0101001	Contoso 512MB MP3 Player E51 Silver	512MB USB driver plays MP3 and WMA	
2	0101002	Contoso 512MB MP3 Player E51 Blue	512MB USB driver plays MP3 and WMA	
3	0101003	Contoso 1GB MP3 Player E100 White	1GB flash memory and USB driver plays MP3 and WMA	
4	0101004	Contoso 2GB MP3 Player E200 Silver	2GB flash memory, LCD display, plays MP3 and WMA	
5	0101005	Contoso 2GB MP3 Player E200 Red	2GB flash memory, LCD display, plays MP3 and WMA	
6	0101006	Contoso 2GB MP3 Player E200 Black	2GB flash memory, LCD display, plays MP3 and WMA	
7	0101007	Contoso 2GB MP3 Player E200 Blue	2GB flash memory, LCD display, plays MP3 and WMA	
8	0101008	Contoso 4GB MP3 Player E400 Silver	4GB flash memory and FM Radio, LCD Display with 7-Color Backlight, plays MP3 and WMA	
9	0101009	Contoso 4GB MP3 Player E400 Black	4GB flash memory and FM Radio, LCD Display with 7-Color Backlight, plays MP3 and WMA	
10	0101010	Contoso 4GB MP3 Player E400 Green	4GB flash memory and FM Radio, LCD Display with 7-Color Backlight, plays MP3 and WMA	
11	0101011	Contoso 4GB MP3 Player E400 Orange	4GB flash memory and FM Radio, LCD Display with 7-Color Backlight, plays MP3 and WMA	
12	0101012	Contoso 4GB Flash MP3 Player E401 Blue	1.8" color LCD, play MP3, WMA and Video MTV, and share JPG	
13	0101013	Contoso 4GB Flash MP3 Player E401 Black	1.8" color LCD, play MP3, WMA and Video MTV, and share JPG	
14	0101014	Contoso 4GB Flash MP3 Player E401 Silver	1.8" color LCD, play MP3, WMA and Video MTV, and share JPG	
15	0101015	Contoso 4GB Flash MP3 Player E401 White	1.8" color LCD, play MP3, WMA and Video MTV, and share JPG	
16	0101016	Contoso 8GB Super-Slim MP3/Video Player M800 White	2" color LCD, Touchpad, Plays music, video, photos and text	
17	0101017	Contoso 8GB Super-Slim MP3/Video Player M800 Red	2" color LCD, Touchpad, Plays music, video, photos and text	
18	0101018	Contoso 8GB Super-Slim MP3/Video Player M800 Green	2" color LCD, Touchpad, Plays music, video, photos and text	
19	0101019	Contoso 8GB Super-Slim MP3/Video Player M800 Pink	2" color LCD, Touchpad, Plays music, video, photos and text	
20	0101020	Contoso 8GB MP3 Player new model M820 Black	2" LCD with blue-white LED, 320x240-pixel, plays music, video, photos and text, display JPEG, BMP, GIF, TIFF and PNG	
21	0101021	Contoso 8GB MP3 Player new model M820 Blue	2" LCD with blue-white LED, 320x240-pixel, plays music, video, photos and text, display JPEG, BMP, GIF, TIFF and PNG	
22	0101022	Contoso 8GB MP3 Player new model M820 Yellow	2" LCD with blue-white LED, 320x240-pixel, plays music, video, photos and text, display JPEG, BMP, GIF, TIFF and PNG	
23	0101023	Contoso 8GB MP3 Player new model M820 White	2" LCD with blue-white LED, 320x240-pixel, plays music, video, photos and text, display JPEG, BMP, GIF, TIFF and PNG	
24	0101024	Contoso 16GB Mp5 Player M1600 Blue	3" 16:9 TFT Touch screen, 16GB flash memory, plays AVI/RM/RMVB/FLV	
25	0101025	Contoso 16GB Mp5 Player M1600 Black	3" 16:9 TFT Touch screen, 16GB flash memory, plays AVI/RM/RMVB/FLV	
26	0101026	Contoso 16GB Mp5 Player M1600 Green	3" 16:9 TFT Touch screen, 16GB flash memory, plays AVI/RM/RMVB/FLV	

The logic for ordering the rows of the query result mirrors that used for relational sources:

- If the DAX expression contains an "ORDER BY" clause, it determines the sequence of rows



- If there is no "ORDER BY" in the DAX expression and one or more columns has a Sort Type of Ascending or Descending, then the result is sequenced according to the specified sorts
- If neither of the above conditions are true, then the result is sequenced by the key columns, if any
- If none of these conditions applies, the result appears in whatever sequence is returned by AS

ASQuery Flex Function

It is simple to incorporate meta-data and data in flex views, even if you have not created a Power Pivot model, an ASQuery data source, or any views.

The ASQuery flex function requires 4 arguments:

- ASSource specifies the server and database, and the server can be omitted if the database is on the default AS Server
- Query Expression is a DAX expression that is valid for the database. It may also be a valid MDX expression.
- Row is the cardinal row number of the data set. Row 1 designates the column headings.
- Column is a specification (either index or name) of the column

The screenshot displays the ASQuery Flex Function interface. On the left, a DAX query is entered in the 'Query Expression' field. The query is as follows:

```
1 // DAX Query
2 SELECT NON EMPTY {
3     // DAX Query
4     EVALUATE (
5         FILTER(
6             Time,
7             Time[Quarters Name] = "Q1 2010"
8         )
9     )
10 }
11
12 // DAX Query
13 EVALUATE (
14     FILTER(
15         Time,
16         Time[Quarters Name] = "Q1 2010"
17     )
18 )
19
20 // DAX Query
21 EVALUATE (
22     FILTER(
23         Time,
24         Time[Quarters Name] = "Q1 2010"
25     )
26 )
27
28 // DAX Query
29 EVALUATE (
30     FILTER(
31         Time,
32         Time[Quarters Name] = "Q1 2010"
33     )
34 )
35
36 // DAX Query
37 EVALUATE (
38     FILTER(
39         Time,
40         Time[Quarters Name] = "Q1 2010"
41     )
42 )
43
44 // DAX Query
45 EVALUATE (
46     FILTER(
47         Time,
48         Time[Quarters Name] = "Q1 2010"
49     )
50 )
51
52 // DAX Query
53 EVALUATE (
54     FILTER(
55         Time,
56         Time[Quarters Name] = "Q1 2010"
57     )
58 )
59
60 // DAX Query
61 EVALUATE (
62     FILTER(
63         Time,
64         Time[Quarters Name] = "Q1 2010"
65     )
66 )
67
68 // DAX Query
69 EVALUATE (
70     FILTER(
71         Time,
72         Time[Quarters Name] = "Q1 2010"
73     )
74 )
75
76 // DAX Query
77 EVALUATE (
78     FILTER(
79         Time,
80         Time[Quarters Name] = "Q1 2010"
81     )
82 )
83
84 // DAX Query
85 EVALUATE (
86     FILTER(
87         Time,
88         Time[Quarters Name] = "Q1 2010"
89     )
90 )
91
92 // DAX Query
93 EVALUATE (
94     FILTER(
95         Time,
96         Time[Quarters Name] = "Q1 2010"
97     )
98 )
99
100 // DAX Query
101 EVALUATE (
102     FILTER(
103         Time,
104         Time[Quarters Name] = "Q1 2010"
105     )
106 )
107
108 // DAX Query
109 EVALUATE (
110     FILTER(
111         Time,
112         Time[Quarters Name] = "Q1 2010"
113     )
114 )
115
116 // DAX Query
117 EVALUATE (
118     FILTER(
119         Time,
120         Time[Quarters Name] = "Q1 2010"
121     )
122 )
123
124 // DAX Query
125 EVALUATE (
126     FILTER(
127         Time,
128         Time[Quarters Name] = "Q1 2010"
129     )
130 )
131
132 // DAX Query
133 EVALUATE (
134     FILTER(
135         Time,
136         Time[Quarters Name] = "Q1 2010"
137     )
138 )
139
140 // DAX Query
141 EVALUATE (
142     FILTER(
143         Time,
144         Time[Quarters Name] = "Q1 2010"
145     )
146 )
147
148 // DAX Query
149 EVALUATE (
150     FILTER(
151         Time,
152         Time[Quarters Name] = "Q1 2010"
153     )
154 )
155
156 // DAX Query
157 EVALUATE (
158     FILTER(
159         Time,
160         Time[Quarters Name] = "Q1 2010"
161     )
162 )
163
164 // DAX Query
165 EVALUATE (
166     FILTER(
167         Time,
168         Time[Quarters Name] = "Q1 2010"
169     )
170 )
171
172 // DAX Query
173 EVALUATE (
174     FILTER(
175         Time,
176         Time[Quarters Name] = "Q1 2010"
177     )
178 )
179
180 // DAX Query
181 EVALUATE (
182     FILTER(
183         Time,
184         Time[Quarters Name] = "Q1 2010"
185     )
186 )
187
188 // DAX Query
189 EVALUATE (
190     FILTER(
191         Time,
192         Time[Quarters Name] = "Q1 2010"
193     )
194 )
195
196 // DAX Query
197 EVALUATE (
198     FILTER(
199         Time,
200         Time[Quarters Name] = "Q1 2010"
201     )
202 )
203
204 // DAX Query
205 EVALUATE (
206     FILTER(
207         Time,
208         Time[Quarters Name] = "Q1 2010"
209     )
210 )
211
212 // DAX Query
213 EVALUATE (
214     FILTER(
215         Time,
216         Time[Quarters Name] = "Q1 2010"
217     )
218 )
219
220 // DAX Query
221 EVALUATE (
222     FILTER(
223         Time,
224         Time[Quarters Name] = "Q1 2010"
225     )
226 )
227
228 // DAX Query
229 EVALUATE (
230     FILTER(
231         Time,
232         Time[Quarters Name] = "Q1 2010"
233     )
234 )
235
236 // DAX Query
237 EVALUATE (
238     FILTER(
239         Time,
240         Time[Quarters Name] = "Q1 2010"
241     )
242 )
243
244 // DAX Query
245 EVALUATE (
246     FILTER(
247         Time,
248         Time[Quarters Name] = "Q1 2010"
249     )
250 )
251
252 // DAX Query
253 EVALUATE (
254     FILTER(
255         Time,
256         Time[Quarters Name] = "Q1 2010"
257     )
258 )
259
260 // DAX Query
261 EVALUATE (
262     FILTER(
263         Time,
264         Time[Quarters Name] = "Q1 2010"
265     )
266 )
267
268 // DAX Query
269 EVALUATE (
270     FILTER(
271         Time,
272         Time[Quarters Name] = "Q1 2010"
273     )
274 )
275
276 // DAX Query
277 EVALUATE (
278     FILTER(
279         Time,
280         Time[Quarters Name] = "Q1 2010"
281     )
282 )
283
284 // DAX Query
285 EVALUATE (
286     FILTER(
287         Time,
288         Time[Quarters Name] = "Q1 2010"
289     )
290 )
291
292 // DAX Query
293 EVALUATE (
294     FILTER(
295         Time,
296         Time[Quarters Name] = "Q1 2010"
297     )
298 )
299
300 // DAX Query
301 EVALUATE (
302     FILTER(
303         Time,
304         Time[Quarters Name] = "Q1 2010"
305     )
306 )
307
308 // DAX Query
309 EVALUATE (
310     FILTER(
311         Time,
312         Time[Quarters Name] = "Q1 2010"
313     )
314 )
315
316 // DAX Query
317 EVALUATE (
318     FILTER(
319         Time,
320         Time[Quarters Name] = "Q1 2010"
321     )
322 )
323
324 // DAX Query
325 EVALUATE (
326     FILTER(
327         Time,
328         Time[Quarters Name] = "Q1 2010"
329     )
330 )
331
332 // DAX Query
333 EVALUATE (
334     FILTER(
335         Time,
336         Time[Quarters Name] = "Q1 2010"
337     )
338 )
339
340 // DAX Query
341 EVALUATE (
342     FILTER(
343         Time,
344         Time[Quarters Name] = "Q1 2010"
345     )
346 )
347
348 // DAX Query
349 EVALUATE (
350     FILTER(
351         Time,
352         Time[Quarters Name] = "Q1 2010"
353     )
354 )
355
356 // DAX Query
357 EVALUATE (
358     FILTER(
359         Time,
360         Time[Quarters Name] = "Q1 2010"
361     )
362 )
363
364 // DAX Query
365 EVALUATE (
366     FILTER(
367         Time,
368         Time[Quarters Name] = "Q1 2010"
369     )
370 )
371
372 // DAX Query
373 EVALUATE (
374     FILTER(
375         Time,
376         Time[Quarters Name] = "Q1 2010"
377     )
378 )
379
380 // DAX Query
381 EVALUATE (
382     FILTER(
383         Time,
384         Time[Quarters Name] = "Q1 2010"
385     )
386 )
387
388 // DAX Query
389 EVALUATE (
390     FILTER(
391         Time,
392         Time[Quarters Name] = "Q1 2010"
393     )
394 )
395
396 // DAX Query
397 EVALUATE (
398     FILTER(
399         Time,
400         Time[Quarters Name] = "Q1 2010"
401     )
402 )
403
404 // DAX Query
405 EVALUATE (
406     FILTER(
407         Time,
408         Time[Quarters Name] = "Q1 2010"
409     )
410 )
411
412 // DAX Query
413 EVALUATE (
414     FILTER(
415         Time,
416         Time[Quarters Name] = "Q1 2010"
417     )
418 )
419
420 // DAX Query
421 EVALUATE (
422     FILTER(
423         Time,
424         Time[Quarters Name] = "Q1 2010"
425     )
426 )
427
428 // DAX Query
429 EVALUATE (
430     FILTER(
431         Time,
432         Time[Quarters Name] = "Q1 2010"
433     )
434 )
435
436 // DAX Query
437 EVALUATE (
438     FILTER(
439         Time,
440         Time[Quarters Name] = "Q1 2010"
441     )
442 )
443
444 // DAX Query
445 EVALUATE (
446     FILTER(
447         Time,
448         Time[Quarters Name] = "Q1 2010"
449     )
450 )
451
452 // DAX Query
453 EVALUATE (
454     FILTER(
455         Time,
456         Time[Quarters Name] = "Q1 2010"
457     )
458 )
459
460 // DAX Query
461 EVALUATE (
462     FILTER(
463         Time,
464         Time[Quarters Name] = "Q1 2010"
465     )
466 )
467
468 // DAX Query
469 EVALUATE (
470     FILTER(
471         Time,
472         Time[Quarters Name] = "Q1 2010"
473     )
474 )
475
476 // DAX Query
477 EVALUATE (
478     FILTER(
479         Time,
480         Time[Quarters Name] = "Q1 2010"
481     )
482 )
483
484 // DAX Query
485 EVALUATE (
486     FILTER(
487         Time,
488         Time[Quarters Name] = "Q1 2010"
489     )
490 )
491
492 // DAX Query
493 EVALUATE (
494     FILTER(
495         Time,
496         Time[Quarters Name] = "Q1 2010"
497     )
498 )
499
500 // DAX Query
501 EVALUATE (
502     FILTER(
503         Time,
504         Time[Quarters Name] = "Q1 2010"
505     )
506 )
507
508 // DAX Query
509 EVALUATE (
510     FILTER(
511         Time,
512         Time[Quarters Name] = "Q1 2010"
513     )
514 )
515
516 // DAX Query
517 EVALUATE (
518     FILTER(
519         Time,
520         Time[Quarters Name] = "Q1 2010"
521     )
522 )
523
524 // DAX Query
525 EVALUATE (
526     FILTER(
527         Time,
528         Time[Quarters Name] = "Q1 2010"
529     )
530 )
531
532 // DAX Query
533 EVALUATE (
534     FILTER(
535         Time,
536         Time[Quarters Name] = "Q1 2010"
537     )
538 )
539
540 // DAX Query
541 EVALUATE (
542     FILTER(
543         Time,
544         Time[Quarters Name] = "Q1 2010"
545     )
546 )
547
548 // DAX Query
549 EVALUATE (
550     FILTER(
551         Time,
552         Time[Quarters Name] = "Q1 2010"
553     )
554 )
555
556 // DAX Query
557 EVALUATE (
558     FILTER(
559         Time,
560         Time[Quarters Name] = "Q1 2010"
561     )
562 )
563
564 // DAX Query
565 EVALUATE (
566     FILTER(
567         Time,
568         Time[Quarters Name] = "Q1 2010"
569     )
570 )
571
572 // DAX Query
573 EVALUATE (
574     FILTER(
575         Time,
576         Time[Quarters Name] = "Q1 2010"
577     )
578 )
579
580 // DAX Query
581 EVALUATE (
582     FILTER(
583         Time,
584         Time[Quarters Name] = "Q1 2010"
585     )
586 )
587
588 // DAX Query
589 EVALUATE (
590     FILTER(
591         Time,
592         Time[Quarters Name] = "Q1 2010"
593     )
594 )
595
596 // DAX Query
597 EVALUATE (
598     FILTER(
599         Time,
600         Time[Quarters Name] = "Q1 2010"
601     )
602 )
603
604 // DAX Query
605 EVALUATE (
606     FILTER(
607         Time,
608         Time[Quarters Name] = "Q1 2010"
609     )
610 )
611
612 // DAX Query
613 EVALUATE (
614     FILTER(
615         Time,
616         Time[Quarters Name] = "Q1 2010"
617     )
618 )
619
620 // DAX Query
621 EVALUATE (
622     FILTER(
623         Time,
624         Time[Quarters Name] = "Q1 2010"
625     )
626 )
627
628 // DAX Query
629 EVALUATE (
630     FILTER(
631         Time,
632         Time[Quarters Name] = "Q1 2010"
633     )
634 )
635
636 // DAX Query
637 EVALUATE (
638     FILTER(
639         Time,
640         Time[Quarters Name] = "Q1 2010"
641     )
642 )
643
644 // DAX Query
645 EVALUATE (
646     FILTER(
647         Time,
648         Time[Quarters Name] = "Q1 2010"
649     )
650 )
651
652 // DAX Query
653 EVALUATE (
654     FILTER(
655         Time,
656         Time[Quarters Name] = "Q1 2010"
657     )
658 )
659
660 // DAX Query
661 EVALUATE (
662     FILTER(
663         Time,
664         Time[Quarters Name] = "Q1 2010"
665     )
666 )
667
668 // DAX Query
669 EVALUATE (
670     FILTER(
671         Time,
672         Time[Quarters Name] = "Q1 2010"
673     )
674 )
675
676 // DAX Query
677 EVALUATE (
678     FILTER(
679         Time,
680         Time[Quarters Name] = "Q1 2010"
681     )
682 )
683
684 // DAX Query
685 EVALUATE (
686     FILTER(
687         Time,
688         Time[Quarters Name] = "Q1 2010"
689     )
690 )
691
692 // DAX Query
693 EVALUATE (
694     FILTER(
695         Time,
696         Time[Quarters Name] = "Q1 2010"
697     )
698 )
699
700 // DAX Query
701 EVALUATE (
702     FILTER(
703         Time,
704         Time[Quarters Name] = "Q1 2010"
705     )
706 )
707
708 // DAX Query
709 EVALUATE (
710     FILTER(
711         Time,
712         Time[Quarters Name] = "Q1 2010"
713     )
714 )
715
716 // DAX Query
717 EVALUATE (
718     FILTER(
719         Time,
720         Time[Quarters Name] = "Q1 2010"
721     )
722 )
723
724 // DAX Query
725 EVALUATE (
726     FILTER(
727         Time,
728         Time[Quarters Name] = "Q1 2010"
729     )
730 )
731
732 // DAX Query
733 EVALUATE (
734     FILTER(
735         Time,
736         Time[Quarters Name] = "Q1 2010"
737     )
738 )
739
740 // DAX Query
741 EVALUATE (
742     FILTER(
743         Time,
744         Time[Quarters Name] = "Q1 2010"
745     )
746 )
747
748 // DAX Query
749 EVALUATE (
750     FILTER(
751         Time,
752         Time[Quarters Name] = "Q1 2010"
753     )
754 )
755
756 // DAX Query
757 EVALUATE (
758     FILTER(
759         Time,
760         Time[Quarters Name] = "Q1 2010"
761     )
762 )
763
764 // DAX Query
765 EVALUATE (
766     FILTER(
767         Time,
768         Time[Quarters Name] = "Q1 2010"
769     )
770 )
771
772 // DAX Query
773 EVALUATE (
774     FILTER(
775         Time,
776         Time[Quarters Name] = "Q1 2010"
777     )
778 )
779
780 // DAX Query
781 EVALUATE (
782     FILTER(
783         Time,
784         Time[Quarters Name] = "Q1 2010"
785     )
786 )
787
788 // DAX Query
789 EVALUATE (
790     FILTER(
791         Time,
792         Time[Quarters Name] = "Q1 2010"
793     )
794 )
795
796 // DAX Query
797 EVALUATE (
798     FILTER(
799         Time,
800         Time[Quarters Name] = "Q1 2010"
801     )
802 )
803
804 // DAX Query
805 EVALUATE (
806     FILTER(
807         Time,
808         Time[Quarters Name] = "Q1 2010"
809     )
810 )
811
812 // DAX Query
813 EVALUATE (
814     FILTER(
815         Time,
816         Time[Quarters Name] = "Q1 2010"
817     )
818 )
819
820 // DAX Query
821 EVALUATE (
822     FILTER(
823         Time,
824         Time[Quarters Name] = "Q1 2010"
825     )
826 )
827
828 // DAX Query
829 EVALUATE (
830     FILTER(
831         Time,
832         Time[Quarters Name] = "Q1 2010"
833     )
834 )
835
836 // DAX Query
837 EVALUATE (
838     FILTER(
839         Time,
840         Time[Quarters Name] = "Q1 2010"
841     )
842 )
843
844 // DAX Query
845 EVALUATE (
846     FILTER(
847         Time,
848         Time[Quarters Name] = "Q1 2010"
849     )
850 )
851
852 // DAX Query
853 EVALUATE (
854     FILTER(
855         Time,
856         Time[Quarters Name] = "Q1 2010"
857     )
858 )
859
860 // DAX Query
861 EVALUATE (
862     FILTER(
863         Time,
864         Time[Quarters Name] = "Q1 2010"
865     )
866 )
867
868 // DAX Query
869 EVALUATE (
870     FILTER(
871         Time,
872         Time[Quarters Name] = "Q1 2010"
873     )
874 )
875
876 // DAX Query
877 EVALUATE (
878     FILTER(
879         Time,
880         Time[Quarters Name] = "Q1 2010"
881     )
882 )
883
884 // DAX Query
885 EVALUATE (
886     FILTER(
887         Time,
888         Time[Quarters Name] = "Q1 2010"
889     )
890 )
891
892 // DAX Query
893 EVALUATE (
894     FILTER(
895         Time,
896         Time[Quarters Name] = "Q1 2010"
897     )
898 )
899
900 // DAX Query
901 EVALUATE (
902     FILTER(
903         Time,
904         Time[Quarters Name] = "Q1 2010"
905     )
906 )
907
908 // DAX Query
909 EVALUATE (
910     FILTER(
911         Time,
912         Time[Quarters Name] = "Q1 2010"
913     )
914 )
915
916 // DAX Query
917 EVALUATE (
918     FILTER(
919         Time,
920         Time[Quarters Name] = "Q1 2010"
921     )
922 )
923
924 // DAX Query
925 EVALUATE (
926     FILTER(
927         Time,
928         Time[Quarters Name] = "Q1 2010"
929     )
930 )
931
932 // DAX Query
933 EVALUATE (
934     FILTER(
935         Time,
936         Time[Quarters Name] = "Q1 2010"
937     )
938 )
939
940 // DAX Query
941 EVALUATE (
942     FILTER(
943         Time,
944         Time[Quarters Name] = "Q1 2010"
945     )
946 )
947
948 // DAX Query
949 EVALUATE (
950     FILTER(
951         Time,
952         Time[Quarters Name] = "Q1 2010"
953     )
954 )
955
956 // DAX Query
957 EVALUATE (
958     FILTER(
959         Time,
960         Time[Quarters Name] = "Q1 2010"
961     )
962 )
963
964 // DAX Query
965 EVALUATE (
966     FILTER(
967         Time,
968         Time[Quarters Name] = "Q1 2010"
969     )
970 )
971
972 // DAX Query
973 EVALUATE (
974     FILTER(
975         Time,
976         Time[Quarters Name] = "Q1 2010"
977     )
978 )
979
980 // DAX Query
981 EVALUATE (
982     FILTER(
983         Time,
984         Time[Quarters Name] = "Q1 2010"
985     )
986 )
987
988 // DAX Query
989 EVALUATE (
990     FILTER(
991         Time,
992         Time[Quarters Name] = "Q1 2010"
993     )
994 )
995
996 // DAX Query
997 EVALUATE (
998     FILTER(
999         Time,
1000         Time[Quarters Name] = "Q1 2010"
1001     )
1002 )
1003
1004 // DAX Query
1005 EVALUATE (
1006     FILTER(
1007         Time,
1008         Time[Quarters Name] = "Q1 2010"
1009     )
1010 )
1011
1012 // DAX Query
1013 EVALUATE (
1014     FILTER(
1015         Time,
1016         Time[Quarters Name] = "Q1 2010"
1017     )
1018 )
1019
1020 // DAX Query
1021 EVALUATE (
1022     FILTER(
1023         Time,
1024         Time[Quarters Name] = "Q1 2010"
1025     )
1026 )
1027
1028 // DAX Query
1029 EVALUATE (
1030     FILTER(
1031         Time,
1032         Time[Quarters Name] = "Q1 2010"
1033     )
1034 )
1035
1036 // DAX Query
1037 EVALUATE (
1038     FILTER(
1039         Time,
1040         Time[Quarters Name] = "Q1 2010"
1041     )
1042 )
1043
1044 // DAX Query
1045 EVALUATE (
1046     FILTER(
1047         Time,
1048         Time[Quarters Name] = "Q1 2010"
1049     )
1050 )
1051
1052 // DAX Query
1053 EVALUATE (
1054     FILTER(
1055         Time,
1056         Time[Quarters Name] = "Q1 2010"
1057     )
1058 )
1059
1060 // DAX Query
1061 EVALUATE (
1062     FILTER(
1063         Time,
1064         Time[Quarters Name] = "Q1 2010"
1065     )
1066 )
1067
1068 // DAX Query
1069 EVALUATE (
1070     FILTER(
1071         Time,
1072         Time[Quarters Name] = "Q1 2010"
1073     )
1074 )
1075
1076 // DAX Query
1077 EVALUATE (
1078     FILTER(
1079         Time,
1080         Time[Quarters Name] = "Q1 2010"
1081     )
1082 )
1083
1084 // DAX Query
1085 EVALUATE (
1086     FILTER(
1087         Time,
1088         Time[Quarters Name] = "Q1 2010"
1089     )
1090 )
1091
1092 // DAX Query
1093 EVALUATE (
1094     FILTER(
1095         Time,
1096         Time[Quarters Name] = "Q1 2010"
1097     )
1098 )
1099
1100 // DAX Query
1101 EVALUATE (
1102     FILTER(
1103         Time,
1104         Time[Quarters Name] = "Q1 2010"
1105     )
1106 )
1107
1108 // DAX Query
1109 EVALUATE (
1110     FILTER(
1111         Time,
1112         Time[Quarters Name] = "Q1 2010"
1113     )
1114 )
1115
1116 // DAX Query
1117 EVALUATE (
1118     FILTER(
1119         Time,
1120         Time[Quarters Name] = "Q1 2010"
1121     )
1122 )
1123
1124 // DAX Query
1125 EVALUATE (
1126     FILTER(
1127         Time,
1128         Time[Quarters Name] = "Q1 2010"
1129     )
1130 )
1131
1132 // DAX Query
1133 EVALUATE (
1134     FILTER(
1135         Time,
1136         Time[Quarters Name] = "Q1 2010"
1137     )
1138 )
1139
1140 // DAX Query
1141 EVALUATE (
1142     FILTER(
1143         Time,
1144         Time[Quarters Name] = "Q1 2010"
1145     )
1146 )
1147
1148 // DAX Query
1149 EVALUATE (
1150     FILTER(
1151         Time,
1152         Time[Quarters Name] = "Q1 2010"
1153     )
1154 )
1155
1156 // DAX Query
1157 EVALUATE (
1158     FILTER(
1159         Time,
1160         Time[Quarters Name] = "Q1 2010"
1161     )
1162 )
1163
1164 // DAX Query
1165 EVALUATE (
1166     FILTER(
1167         Time,
1168         Time[Quarters Name] = "Q1 2010"
1169     )
1170 )
1171
1172 // DAX Query
1173 EVALUATE (
1174     FILTER(
1175         Time,
1176         Time[Quarters Name] = "Q1 2010"
1177     )
1178 )
1179
1180 // DAX Query
1181 EVALUATE (
1182     FILTER(
1183         Time,
1184         Time[Quarters Name] = "Q1 2010"
1185     )
1186 )
1187
1188 // DAX Query
1189 EVALUATE (
1190     FILTER(
1191         Time,
1192         Time[Quarters Name] = "Q1 2010"
1193     )
1194 )
1195
1196 // DAX Query
1197 EVALUATE (
1198     FILTER(
1199         Time,
1200         Time[Quarters Name] = "Q1 2010"
1201     )
1202 )
1203
1204 // DAX Query
1205 EVALUATE (
1206     FILTER(
1207         Time,
1208         Time[Quarters Name] = "Q1 2010"
1209     )
1210 )
1211
1212 // DAX Query
1213 EVALUATE (
1214     FILTER(
1215         Time,
1216         Time[Quarters Name] = "Q1 2010"
1217     )
1218 )
1219
1220 // DAX Query
1221 EVALUATE (
1222     FILTER(
1223         Time,
1224         Time[Quarters Name] = "Q1 2010"
1225     )
1226 )
1227
1228 // DAX Query
1229 EVALUATE (
1230     FILTER(
1231         Time,
1232         Time[Quarters Name] = "Q1 2010"
1233     )
1234 )
1235
1236 // DAX Query
1237 EVALUATE (
1238     FILTER(
1239         Time,
1240         Time[Quarters Name] = "Q1 2010"
1241     )
1242 )
1243
1244 // DAX Query
1245 EVALUATE (
1246     FILTER(
1247         Time,
1248         Time[Quarters Name] = "Q1 2010"
1249     )
1250 )
1251
1252 // DAX Query
1253 EVALUATE (
1254     FILTER(
1255         Time,
1256         Time[Quarters Name] = "Q1 2010"
1257     )
1258 )
1259
1260 // DAX Query
1261 EVALUATE (
1262     FILTER(
1263         Time,
1264         Time[Quarters Name] = "Q1 2010"
1265     )
1266 )
1267
1268 // DAX Query
1269 EVALUATE (
1270     FILTER(
1271         Time,
1272         Time[Quarters Name] = "Q1 2010"
1273     )
1274 )
1275
1276 // DAX Query
1277 EVALUATE (
1278     FILTER(
1279         Time,
1280         Time[Quarters Name] = "Q1 2010"
1281     )
1282 )
1283
1284 // DAX Query
1285 EVALUATE (
1286     FILTER(
1287         Time,
1288         Time[Quarters Name] = "Q1 2010"
1289     )
1290 )
1291
1292 // DAX Query
1293 EVALUATE (
1294     FILTER(
1295         Time,
1296         Time[Quarters Name] = "Q1 2010"
1297     )
1298 )
1299
1300 // DAX Query
1301 EVALUATE (
1302     FILTER(
1303         Time,
1304         Time[Quarters Name] = "Q1 2010"
1305     )
1306 )
1307
1308 // DAX Query
1309 EVALUATE (
1310     FILTER(
1311         Time,
1312         Time[Quarters Name] = "Q1 2010"
1313     )
1314 )
1315
1316 // DAX Query
1317 EVALUATE (
1318     FILTER(
1319         Time,
1320         Time[Quarters Name] = "Q1 2010"
1321     )
1322 )
1323
1324 // DAX Query
1325 EVALUATE (
1326     FILTER(
1327         Time,
1328         Time[Quarters Name] = "Q1 2010"
1329     )
1330 )
1331
1332 // DAX Query
1333 EVALUATE (
1334     FILTER(
1335         Time,
1336         Time[Quarters Name] = "Q1 2010"
1337     )
1338 )
1339
1340 // DAX Query
1341 EVALUATE (
1342     FILTER(
1343         Time,
1344         Time[Quarters Name] = "Q1 2010"
1345     )
1346 )
1347
1348 // DAX Query
1349 EVALUATE (
1350     FILTER(
1351         Time,
1352         Time[Quarters Name] = "Q1 2010"
1353     )
1354 )
1355
1356 // DAX Query
1357 EVALUATE (
1358     FILTER(
1359         Time,
1360         Time[Quarters Name] = "Q1 2010"
1361     )
1362 )
1363
1364 // DAX Query
1365 EVALUATE (
1366     FILTER(
1367         Time,
1368         Time[Quarters Name] = "Q1 2010"
1369     )
1370 )
1371
1372 // DAX Query
1373 EVALUATE (
1374     FILTER(
1375         Time,
1376         Time[Quarters Name] = "Q1 2010"
1377     )
1378 )
1379
1380 // DAX Query
1381 EVALUATE (
1382     FILTER(
1383         Time,
1384         Time[Quarters Name] = "Q1 2010"
1385     )
1386 )
1387
1388 // DAX Query
1389 EVALUATE (
1390     FILTER(
1391         Time,
1392         Time[Quarters Name] = "Q1 2010"
1393     )
1394 )
1395
1396 // DAX Query
1397 EVALUATE (
```

Properties for Model Contoso AS

Search

Identification

Name	Contoso AS
ID	CONTOSOAS
Class	Model
Subclass	Source Data
Category	Development (ID: DEVELOPMENT)
Description	Copy of object template: Blank Model

Structure

Source Data	Contoso Fact (ID: CONTOSOFACT)
-------------	--------------------------------

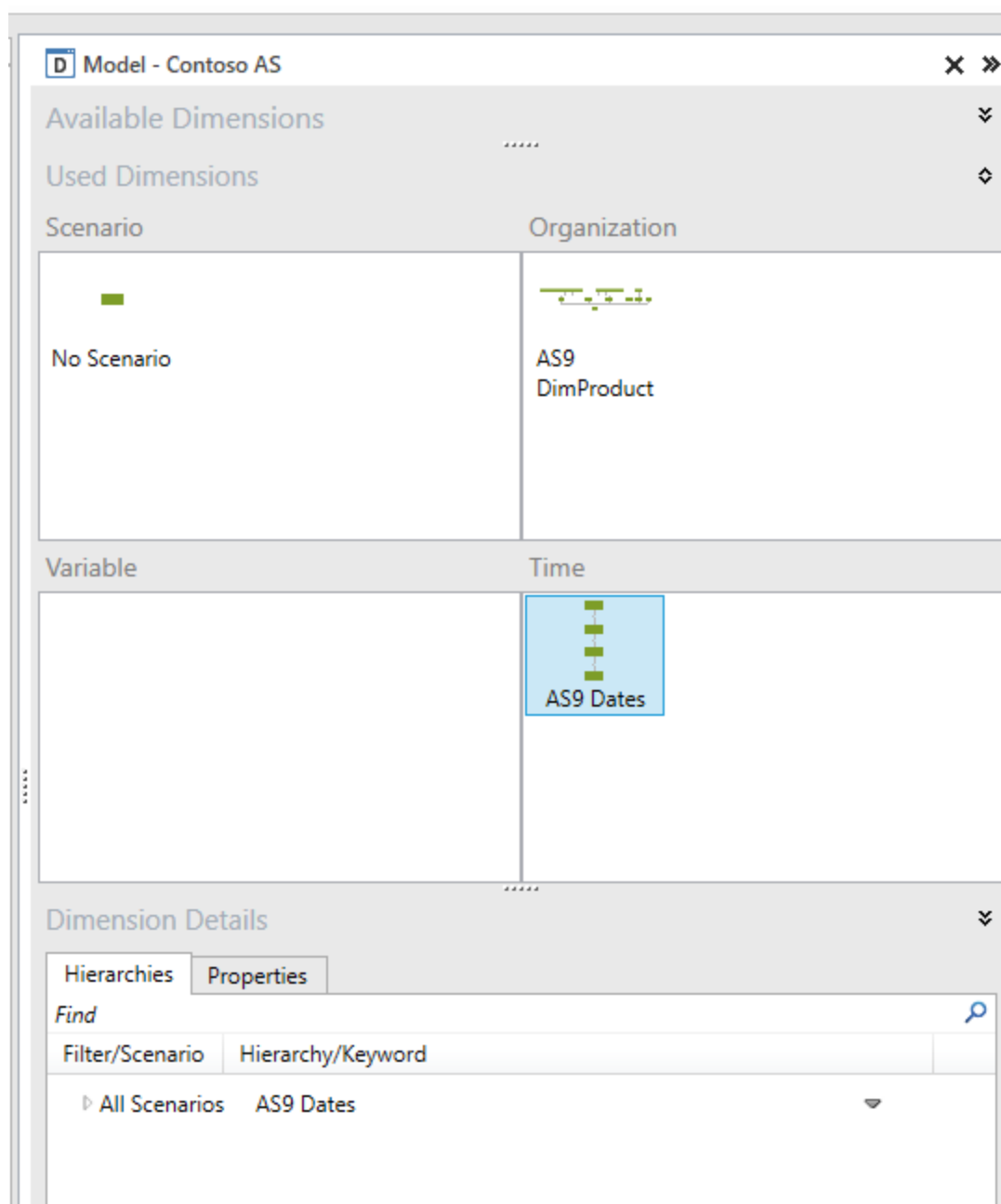
Logging

Accessibility

Miscellaneous

OK Cancel

Then add the dimensions you need (note that they don't need to be derived from the underlying AS model and can contain levels and attributes that do not exist in the model.)



Finally, define a mapping which connects the levels of the dimensions to columns of the query result:

Mapping - Contoso AS

Choose Aspect Types

To map aspects, drag aspect from the tree to the worksheet.

Aspects	Source	Source Text
Contoso AS		
No Scenario (Scenario Dimension)		
AS9 DimProduct (Organization Dimension)		
AS9 ProductSubcategory (Organization Level)		
AS9 ProductCategoryKey (Organization Level)		
AS9 ProductCategory (Organization Level)		
AS9 Status (Organization Level)		
AS9 DimProduct Total (Organization Level)		
AS9 SizeRange (Organization Level)		
AS9 SizeUnitMeasure (Organization Level)		
AS9 Brand (Organization Level)		
AS9 Manufacturer (Organization Level)		
AS9 ProductSubcategoryKey (Organization Lev		
AS9 Color (Organization Level)		
AS9 UnitOfMeasure (Organization Level)		
AS9 WeightUnitMeasure (Organization Level)		
AS9 Style (Organization Level)		
AS9 StockType (Organization Level)		
AS9 Class (Organization Level)		
AS9 ProductKey (Organization Level)	ProductKey	
AS9 Product (Organization Level)		
AS9 Dates (Time Dimension)		
AS9 Dates Total (Time Level)		
AS9 Year (Time Level)		
AS9 Month (Time Level)		
AS9 Date (Time Level)	DateKey	

Aspect Properties

☐ Immediate update


Update

Maximize



CONTROL®

Now you can create any views you like on the datasource, using filters and branches of the dimensions and filtering or ordering the columns of the query:

	A	B	C	D	E	F	G	H	I	J	K	L	M	
1	Filters: Pages:  COMPANY NAME Company Slogan	Scenario (None)												
2		Product by Color (Adventure Wor.)	A59 Date (Year 2008)											
4														
5														
6														
7														
9														
10		DatasourceColumns												
11	Column1	ProductKey	PromotionKey	CurrencyKey	UnitCost	UnitPrice	SalesQuantity	ReturnQuantity	ReturnAmount	DiscountQuantity	DiscountAmount	TotalCost	SalesAmount	ET
12	*All*	387	1	1	321	699	10	-	-	-	-	3,214	6,990	
13		381	1	1	321	699	10	-	-	-	-	3,214	6,990	
14		369	1	1	321	699	10	-	-	-	-	3,214	6,990	
15		387	1	1	321	699	10	-	-	-	-	3,214	6,990	
16		363	1	1	321	699	10	-	-	-	-	3,214	6,990	
17		387	1	1	321	699	10	-	-	-	-	3,214	6,990	
18		363	1	1	321	699	10	-	-	-	-	3,214	6,990	
19		381	1	1	321	699	10	-	-	-	-	3,214	6,990	
20		494	1	1	128	279	10	-	-	-	-	1,283	2,790	
21		141	1	1	153	300	10	-	-	-	-	1,529	3,000	
22		124	1	1	129	280	10	-	-	-	-	1,288	2,800	
23		367	1	1	166	326	10	-	-	-	-	1,662	3,260	
24		376	1	1	195	383	10	-	-	-	-	1,952	3,830	
25		516	1	1	30	90	10	-	-	-	-	298	900	
26		120	1	1	61	120	10	-	-	-	-	612	1,200	
27		374	1	1	430	1,299	10	-	-	-	-	4,304	12,990	
28		504	1	1	288	869	10	-	-	-	-	2,879	8,690	
29		370	1	1	195	383	10	-	-	-	-	1,952	3,830	
30		494	1	1	128	279	10	-	-	-	-	1,283	2,790	
31		370	1	1	195	383	10	-	-	-	-	1,952	3,830	
32		143	1	1	153	300	10	-	-	-	-	1,529	3,000	
33		128	1	1	73	143	10	-	-	-	-	731	1,434	
34		502	1	1	30	90	10	-	-	-	-	298	900	
35		142	1	1	153	300	10	-	-	-	-	1,529	3,000	
36		142	1	1	153	300	10	-	-	-	-	1,529	3,000	