Paper 182-2008

Building OLAP Cubes with SAS 9: A Hands on Workshop

Gregory S. Nelson
ThotWave Technologies, Chapel Hill, North Carolina

Abstract

On-Line Analytical Processing (or OLAP) has long been part of the data storage and exploitation strategy for the SAS professional. From the early days of MDDBs to the current-day equivalent of OLAP Cubes (part of the SAS 9 OLAP Server), these data structures provide a means for us to store our data, which allow us to slice and dice and drill through and around our data. In a previous hands-on workshop (Nelson, 2006), we demonstrated how to use Data Integration Studio to load a sample star-schema and a similar workshop on the planning and design of the data warehouse (Nelson, 2007).

In this workshop, we will focus on what it takes to build a cube using a design time tool such as OLAP Cube Studio. Here, we will seek to understand our input data, data aggregations, summary options and how we create jobs to populate that structure. In addition, we will explore the concept of OLAP metadata and review how we create, register and manage that metadata. Finally, we will discuss best practices for refreshing data, promotion of changes in a production environment and setting permissions on cubes to ensure that your data is secure. In a separate "Hands on Workshop", we will learn how you can build web interfaces to exploit OLAP cubes (Nelson, 2008 - Exploiting OLAP Cubes with SAS 9: A Hands on Workshop).

Keywords: Data warehouse, DI Studio, OLAP Cube Studio, star schema, MDDB, MDX

Introduction

This hands-on workshop is a continuation of a series of papers on data warehousing that began at SUGI 22 (1997) with a paper entitled "*Implementing a Dimensional Data Warehouse with the SAS® System*". Since then, we have shown how you can plan for, design and build data warehouses using SAS. In this workshop, we look specifically at how to build multi-dimensional data stores (or cubes as they are often referred to) in SAS using OLAP Studio.

The purpose of an OLAP cube is to store data in such a way that an end user can slice and dice through the data and the storage structure itself (multidimensional) while allowing for very fast access to the data. Cubes are characterized by ***dimensions*** and ***measures***. Dimensions refer to how the data is organized and are usually made up of character fields which tell us about the data. In traditional reporting, dimensions are those attributes which we use to slice and dice (filter by groups in a report for example). Some examples of dimensions might include time along with its various ***levels*** (year, quarter, month, day of week, time of day), geography (country, region, state, county, city) and organization (business unit, district). Measures, on the other hand, are those elements in the data which quantify something (a number) and can usually be additive (though not always!). Examples of measures include quantity, sales revenue, average account balance and so on. Measures are comprised of both the number and the rules which define how they can be rolled up. For example, we wouldn't want to add average daily balance as that wouldn't really tell us how much money we had in the bank (although, I'm sure we'd all like the results of that number in our bank account!)

In our examples above, we had a number of levels within each category. When those are used together – that is, multiple levels, we call that a hierarchy. Hierarchies allow us to drill down through a particular dimension and a dimension can have one or more hierarchies associated with it. For example, in our time dimension, we have year, quarter, month, etc. We can "navigate" the hierarchy by traversing the levels in a natural progression.

OLAP Architecture: an Architectural Perspective

Before we dive into the tool, it is important to understand how OLAP Cubes fit into the overall SAS 9 architecture. While each customer site is different with respect to the number of servers and which SAS products are licensed, there are some fundamental similarities. In the diagram below, we note that for example, there are SAS servers and SAS clients.

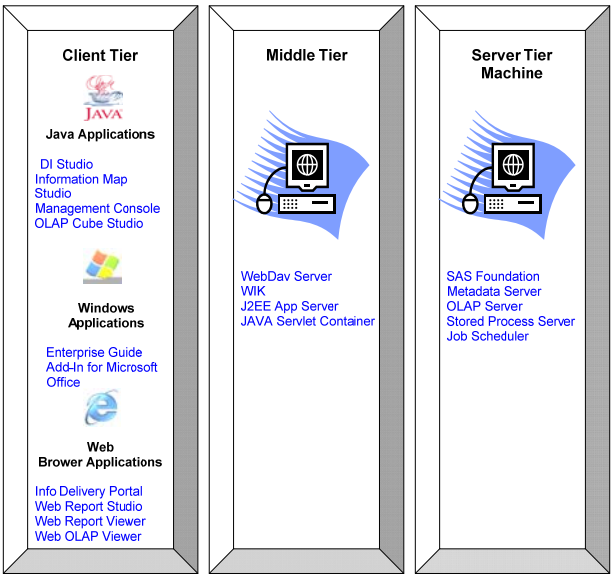


Figure 1. SAS Multi-Tier Architecture (Clients and Servers)

The SAS clients we will discuss in this context include OLAP Cube Studio, Data Integration Studio, the SAS Management Console and SAS Enterprise Guide. Each of these play an important role in our data warehousing activities and communicate to one or more of the SAS servers such as the metadata server, the workspace server, and the stored process server.

In this workshop, we will be interacting with one of these clients – OLAP Cube Studio. An OLAP server is NOT necessary to create a cube. You can use either PROC OLAP, which is part of Base SAS software, and run it as a batch job or as a stored process, or the Cube Designer wizard, which is available in OLAP Cube Studio and SAS ETL Studio. In the latter case, a SAS Workspace Server and an Object Spawner (optional) is needed.

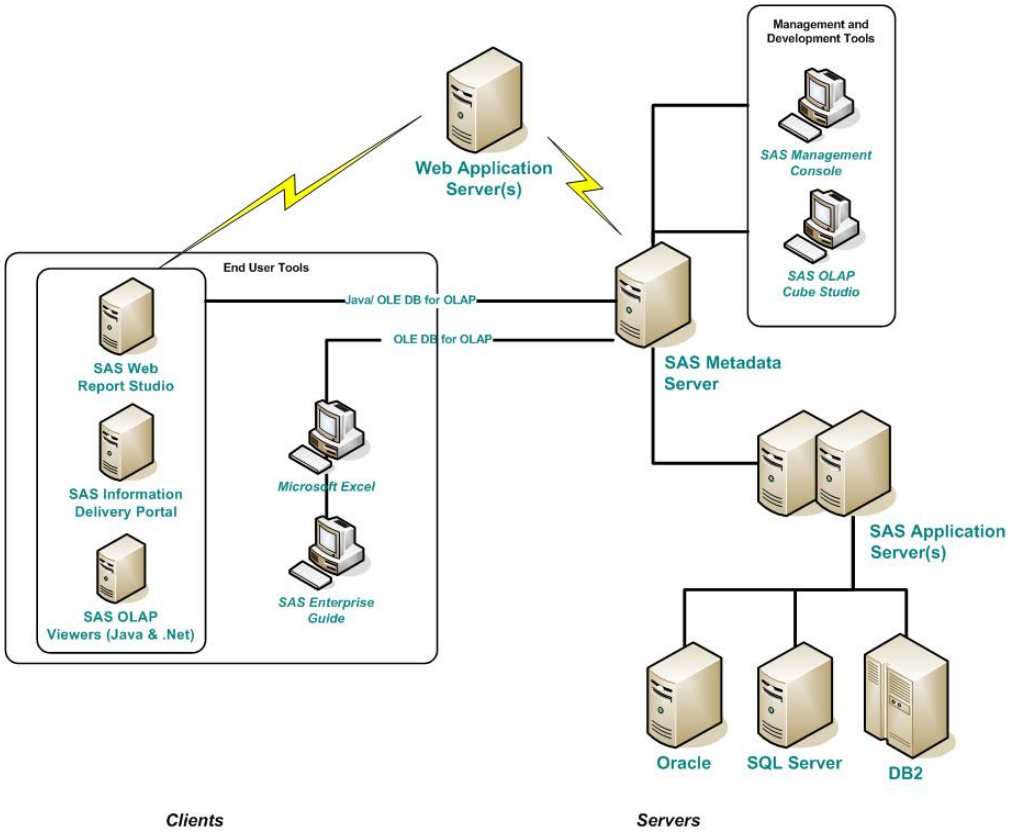


Figure 2. SAS 9 Logical Architecture for OLAP

In Figure 2, we note that the SAS clients (such as OLAP Cube Studio) first authenticate to the Metadata Server. The role of the metadata server is to initially provide access to the SAS 9 servers and then to pass on our requests to the appropriate resource. These servers (such as the workspace server) will then carry out the work and pass the results back to the requesting client.

When creating a cube, metadata about the cube is stored in a SAS Open Metadata Repository. Each component of the SAS 9 Intelligence Architecture is made available on a SAS Open Metadata Repository. This enables sharing of all application elements by both internal and external applications, including server definitions, cubes, tables, and libraries, while maintaining the applicable security definitions.

In order to load cubes a SAS Open Metadata Server must be running and should include the following elements.

- At least one repository
- An OLAP schema
- A database server and a database schema which is required only when using SAS/ACCESS tables
- One or more libraries
- One or more table data sources
- A SAS Workspace Server and an Object Spawner (optional) when using the Cube Designer wizard. The SAS Workspace Server is not needed when using PROC OLAP

- User-defined formats and informats (if they are being used in the input data columns).

Scope of this Workshop

In this workshop, we are going to take some retail data from the Northwinds database and structure the data in a cube – or multidimensional database – so that we can exploit that data using a variety of tools. In a follow-on paper we'll show you how to exploit the data through a variety of techniques including ways to leverage Microsoft Excel and web applications.

In this workshop, our goal is to build a single OLAP cube. We will need to go through the following steps. Note, in this workshop, not every step has an accompanying exercise, but you will gain enough information to perform the work on your own.

1. Understand the source data model (Supporting tool(s): entity relationship diagram using Microsoft Visio, DataFlux dfPower Studio or Base SAS)
    - Understand the relationships that exist in the source data
    - Validate the data and its content
    - Perform a gap analysis on what you do and do not have in terms of data.
2. Design your OLAP cube (Supporting tool(s): whiteboard, Microsoft Visio, OLAP Cube Studio)
    - Model out your new OLAP "schema"
    - Review your reporting requirements with your end users to determine if your cube will satisfy their requirements
3. Define the aggregation rules (Supporting tool(s): Source-to-Target Mapping document)
    - Create rules for mapping data from the source tables to the target cube
4. Convert the data from raw, denormalized data, into an OLAP Cube (Supporting tool(s): SAS OLAP Cube Studio)
    - In OLAP Cube Studio, create your source metadata then use the source to build the cube structure.
5. View the completed OLAP cube using SAS Enterprise Guide 3.0 OLAP Analyzer

The Workshop Data

Please note for continuity the data used, for the exercises below, is the same data used in the previous hands-on workshop. However, even if you did not attend the workshop you should be able to go back to your organization and use it today.

Data Used in this Workshop

The example we will be using throughout the paper will be data from the Northwinds Trading Company database that is distributed with Microsoft Access. The database is an order entry system for a company that sells a broad range of consumable/ food products. We selected this database for building our sample warehouse application for three reasons: (1) it is available to anyone who has MS-Access, (2) MS-Access is ODBC compliant and it was easy to get this data into SAS via SAS/ACCESS to ODBC, and (3) it provides a classic example of an operational system used in order processing with which most of us have some

familiarity. After all, building a warehouse is typically done by extracting information from a transactional system, denormalizing the data and then converting it into a structure more appropriate for reporting and analytics.

This particular data is a good choice since the data is relatively understandable by most everyone (people order things every day and we all eat food!) and we previously used it in an earlier paper on dimensional data warehousing (Nelson, 1999). In the previous paper we described how you would approach ETL from a classical perspective with BASE SAS. For a more detailed description of dimensional data warehousing techniques and its application to SAS we refer you to the previous paper.

THE SOURCE DATA MODEL

Typically the way in which we talk about a transactional system (OLTP) is by reviewing its' ERD or entity-relationship diagram. Figure 1 illustrates the Northwind ERD.

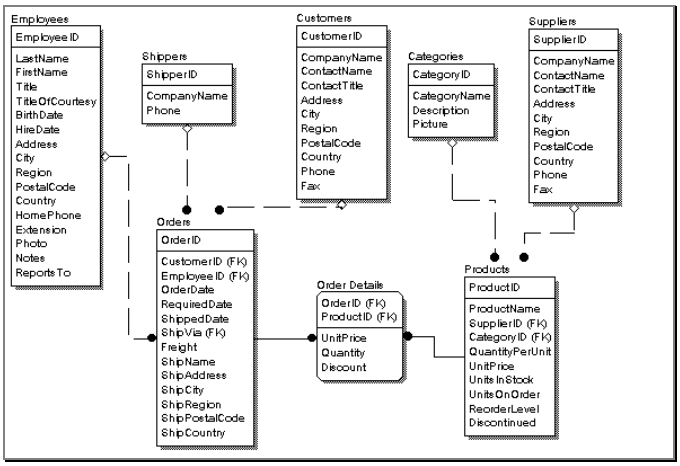


Figure 3. Main subject Area ERD for the OLTP Database (Source)

The diagram shows the relationships between tables in our database. Notice that the Orders table is where most tables feed. This will be a key factor in deciding which facts or subjects we want to model.

In our analysis tool, we don't really need or want the data in that format because we would have to perform complex joins whenever we want a simple report (for example, sales by product). In the next section, we will describe a star schema that would be much more appropriate for this data (our Target).

THE TARGET DATA MODEL

Just as we discussed in the data warehousing series of workshops, we need to spend a considerable amount of energy making sure that what we build is appropriate for the end users who will consume the data. Previously, we developed the star schema shown below.

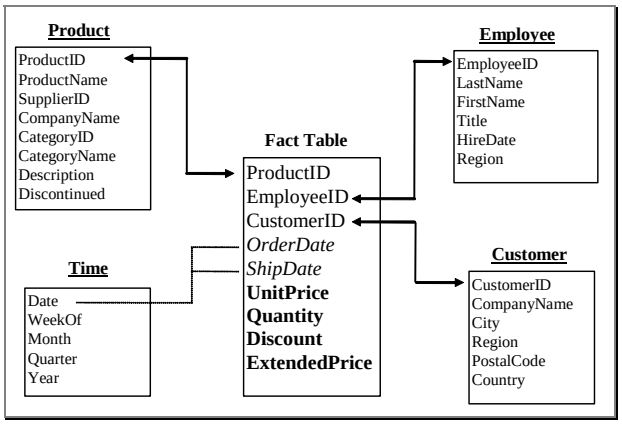


Figure 4. Star-schema model for the Northwinds Trading Company data mart.

In this workshop, instead of converting the data to a star schema, we intend to create OLAP cubes from the data. In order to build a cube, we first have to design it correctly. The first step in designing the cube is to figure out what our dimensions and measures (or facts) will be. Within each dimension, we also need to create the various hierarchies that users will use to drill through the data.

A WORD ABOUT HIERARCHIES, DRILL-DOWNS AND DIMENSIONS

In our warehouse, we know we have sales information for our four dimensions: Time, Customers, Employees and Products. Each of these has several levels that we know our users will be interested in seeing. The table below shows us the four dimensions that we picked along with the associated hierarchies.

Employee	Time	Customer	Product
Employee Name	Order Date	Company Name	Product Name
Region	Day of week	City	Category (Description)
Country	Quarter	Region	
	Year	Country	

Table 1. Northwinds warehouse dimensions

For example, take note of the figure below to review the employee hierarchy.

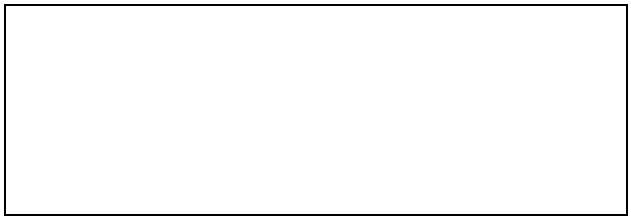


Figure 5. Employee dimension.

Notice that region only has meaning in the USA, so we need to consider that when adding a region level to the data. To review the hierarchies in SAS, you can simply perform crossings with PROC FREQ or PROC SUMMARY. For example, the code below helps us understand the EMPLOYEE hierarchy.

```
 Proc summary data=target.denormalized;
  class salesrep region country;
  var extendedPrice;
  output out=sumstat sum=extended;
run;
```

This produces a summary output dataset containing all of the possible combinations of sales rep, region and country. We can use this in our end-user application for very quick manipulations among the different levels within the dimensions. Navigating the employee hierarchy is referred to as “rolling-up” and “drilling-down”.

In Figure 3, we saw that users can roll-up and drill-down through a single dimension, EMPLOYEE. Well designed warehouses also allow users to roll-up and drill down through multiple dimensions concurrently. Thus, in this example, an end-user could hold the products, customers and employees constant, while drilling-down or rolling up through sales figures over the TIME dimension.

We could also use PROC SQL to create a summary table for salesrep by region by country. Although, not as comprehensive as PROC SUMMARY, PROC SQL has a tremendous amount of power when it comes to the calculation of new information when you create these roll-ups.

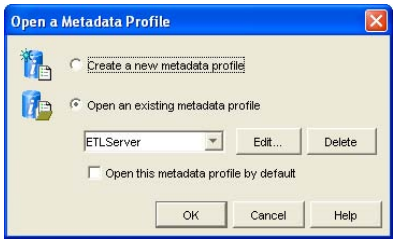
In addition to PROC SUMMARY as a method for creating these aggregate or summary tables, we can also use PROC SQL or PROC OLAP (or its predecessor in SAS Version 8 PROC MDDB). Since this workshop is designed to use the OLAP Cube Studio, let’s begin the exercises.

Exercises

Task #1: Understanding the user interface

TASK 1 – STEP A: LOGGING IN (AUTHENTICATION)

1. The first step in getting into OLAP Cube Studio is authenticating yourself to the metadata server. We do this through the login prompt after launching OLAP Cube Studio.



2. Select the correct metadata profile or create a new one and then provide your credentials.

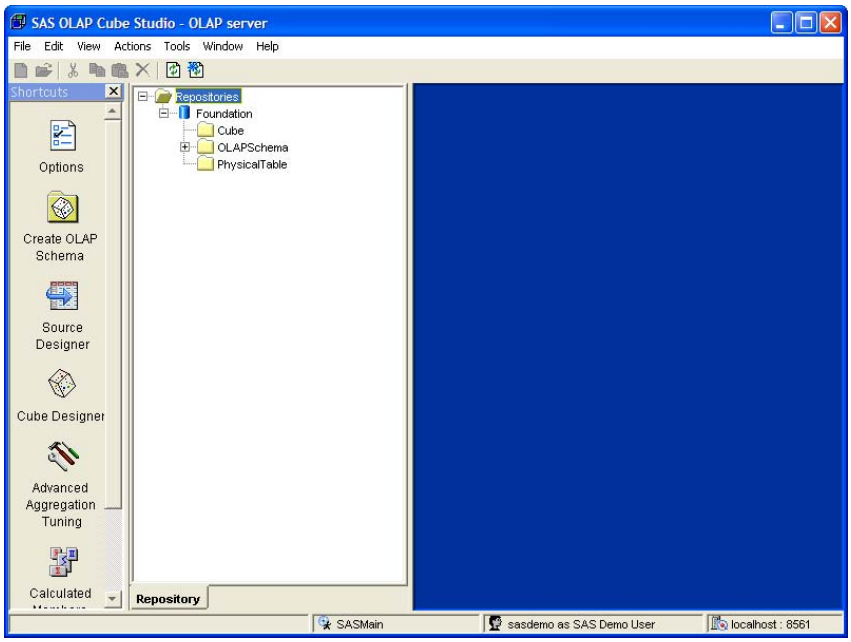
3. You should now be presented with the Main Interface which we can explore. When you start OLAP Cube Studio, the Open a Metadata Profile window and the SAS OLAP Cube Studio desktop display.

At this point, let's pause and think about what we just did. A *Metadata Profile* is a collection of information which helps you to sign in to a specific SAS Metadata Server *and also a specific Metadata Repository*. Throughout this tutorial, we will be working directly on the Foundation Repository – for reasons of clarity and time.

TASK 1 – STEP B: UNDERSTANDING THE SAS OLAP CUBE STUDIO INTERFACES

A. The Desktop

The primary interface for SAS OLAP Cube Studio is shown below. We will highlight some of the things that we will use during this workshop.



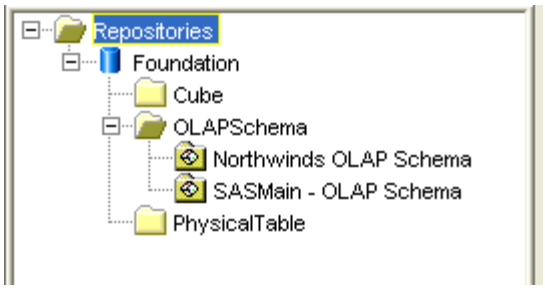
Some of these include: shortcuts, tree viewers and process editor.

B. Shortcuts

The shortcut bar is an optional pane of task icons on the left side of the SAS OLAP Cube Studio desktop. Each icon displays a commonly-used window, wizard, or a selection window for wizards.

C. Tree view (including Inventory, Custom, Process)

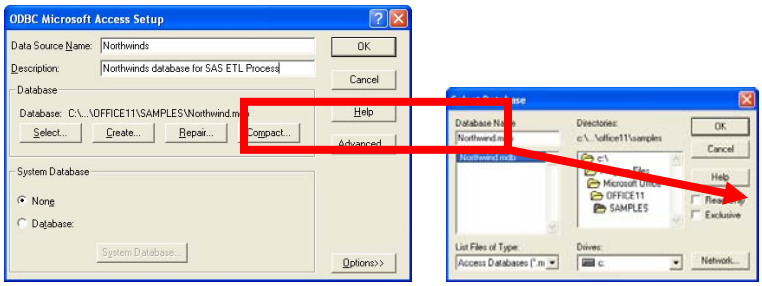
1. **Repositories** – It organizes objects into a set of default groups, such as tables for all tables in a repository and Cubes for all cubes in a repository.



Task #2: Creating libraries (references to data)

TASK 2 - STEP A: CREATE THE CONNECTION TO THE ODBC DATA SOURCE

1. From the Desktop in Windows, select **Start ->Settings -> Control Panels -> Administrative Tools -> Data Sources (ODBC)**
2. Select the **System DSN** Tab → Click **Add** → Select **Microsoft Access Driver (.mdb)** → Click Finish
3. Complete the Data Source Name, description and select the Northwinds database (northwinds.mdb). For our workshop, we have place a copy of the Northwinds.mdb file in *C:\how\nelson_182\data\source*.



4. Finish the wizard by clicking **OK**.

TASK 2 - STEP B: PREPARE OUR DATA FOR BUILDING OUR CUBE

1. First we need to massage the original data that is stored in the relational tables. Since we are using a Microsoft Access database, we will use the SAS/Access driver for ODBC.
2. In creating an OLAP cube, we can use either the star schema tables or a denormalized format. We have prepared the programs to create the fact and dimensions tables as well as a program that will denormalize the data for us.
3. For each of the programs contained in the directory *C:\how\nelson_182\programs*, please include those in the SAS Display Manager and submit each one in order.
   a. autoexec.sas – creates the library references that we will use in our programs.
   b. 2. fact_loader.sas – program that creates the primary fact table in our star schema

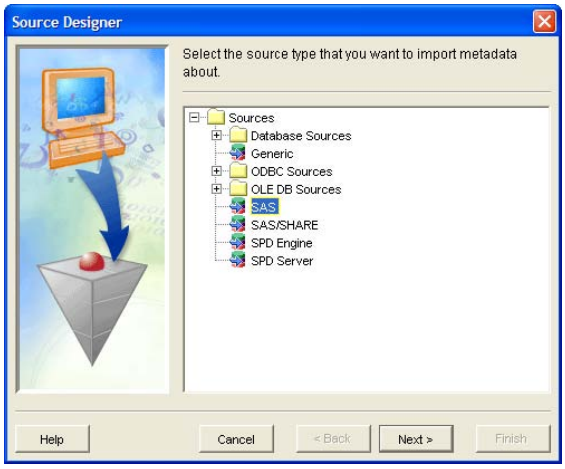
c. 3. dimension_loader.sas – program that creates each of the 4 dimensions in our star schema

d. 4. denormalize.sas – program that takes all of our previously created tables and generates one wide table with all of the columns from the various dimensions.

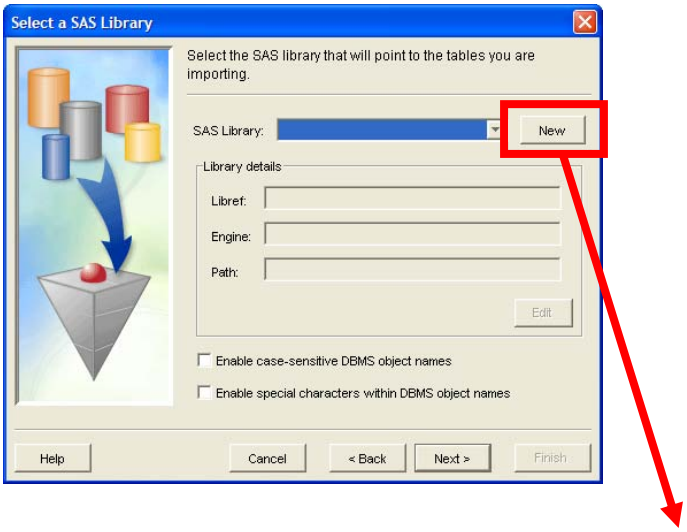
e. Confirm that all of the programs above have been submitted and there are no errors in the log. View the 6 tables that have been created in the Target library.

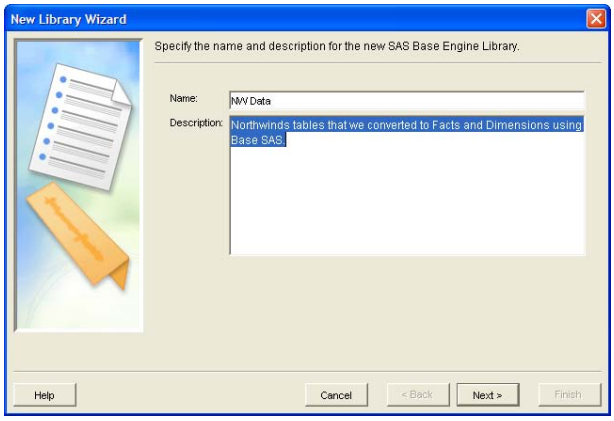
TASK 2 - STEP C: CREATE OUR LIBRARY REFERENCES IN SAS OLAP STUDIO AND REGISTER THE METADATA

1. Start OLAP Cube Studio and log in.
2. Import the metadata from the data that we just created. To do that, simply select **Source Designer** (Found on the shortcuts on the left side). And complete the wizard as shown below.
3. Select **SAS → Next**

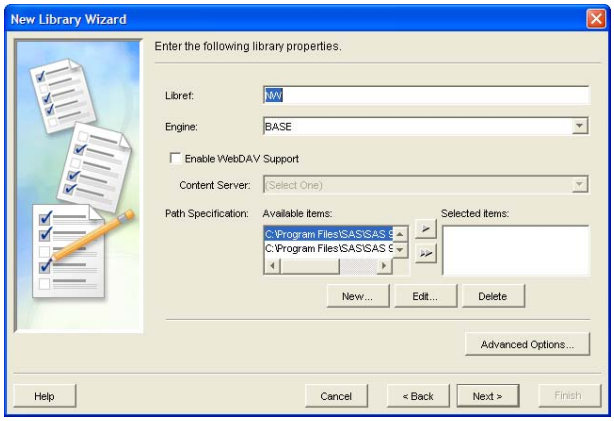


4. Provide a libref and click **New** to define a new path specification and then click **Next**.





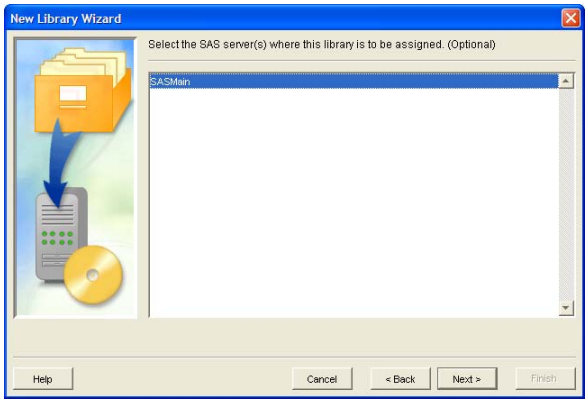
Click **Next**.



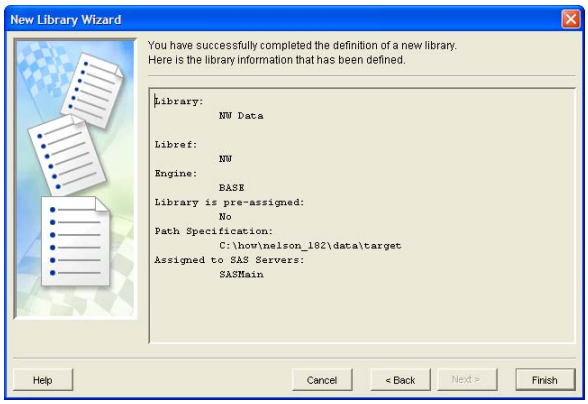
Type the libref "**NW**" and select **New**... to tell SAS where our data lives.

Press **Ok** and then **Next**.

Select the SAS server (**SAS Main**)

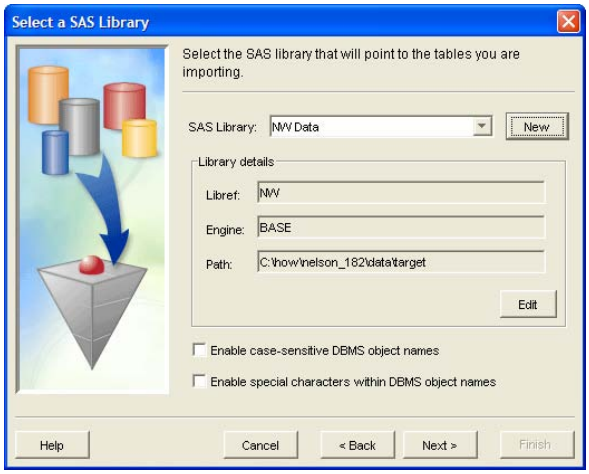


Click **Next**.

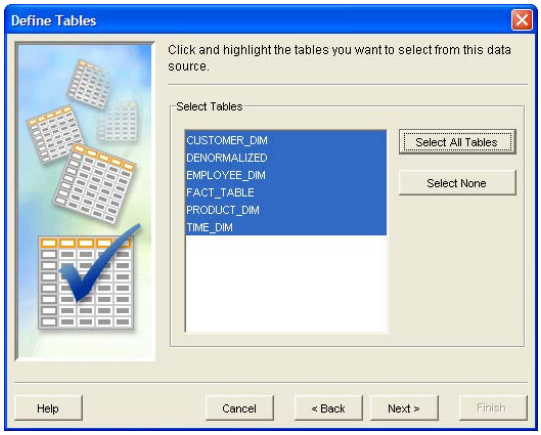


Select **Finish**.

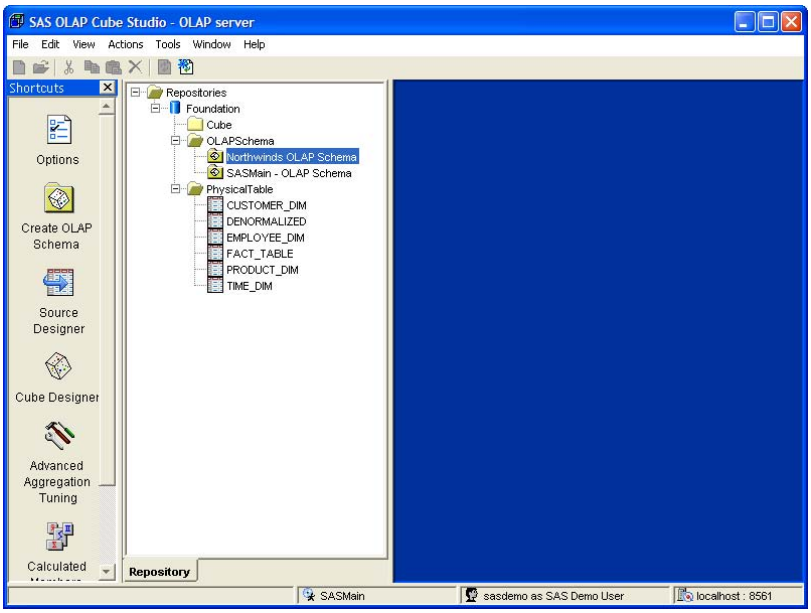
You should now see the following:



5. Next we need to define the tables that we intend to use. Select all of the tables and select Next.



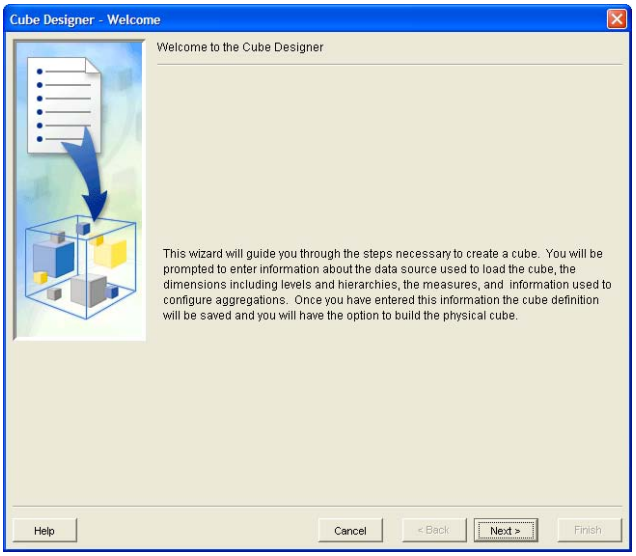
6. Finish the wizard and you should now see a screen that looks like the following.



Task #3: Building the OLAP Cube

TASK 3 - STEP A: CREATE OUR LIBRARY REFERENCES IN SAS OLAP STUDIO AND REGISTER THE METADATA

1. Select the Cube Designer from the shortcut icons on the left in SAS OLAP Cube Studio. This will launch the Cube Designer.



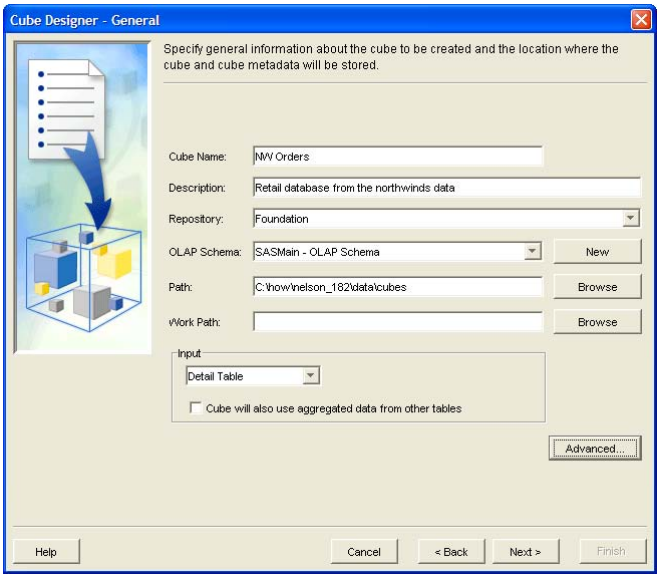
2. We are presented with a screen that asks us to define our cube. Complete the following fields:

   i. **Cube Name:** NW_Orders

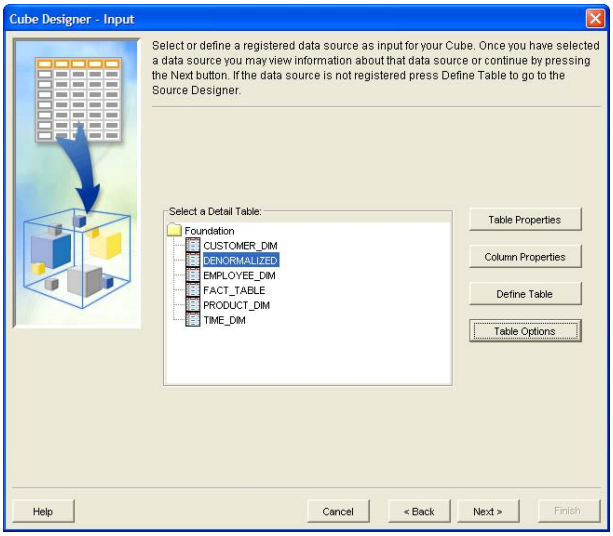
   ii. **Description:** Retail database from the Northwinds data

   iii. **Path:** C:\how\nelson_182\data\cubes

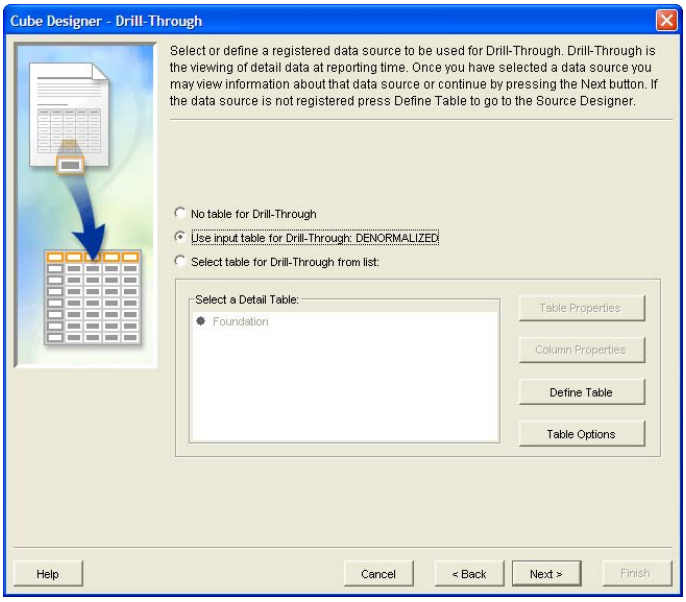
   iv. **Input:** Detail Table



b. Click **Next**. Here we select the table that we want to use as input to our cube. Note: you will need to expand the Foundation repository to see the detail tables. We will select the **denormalized** table.



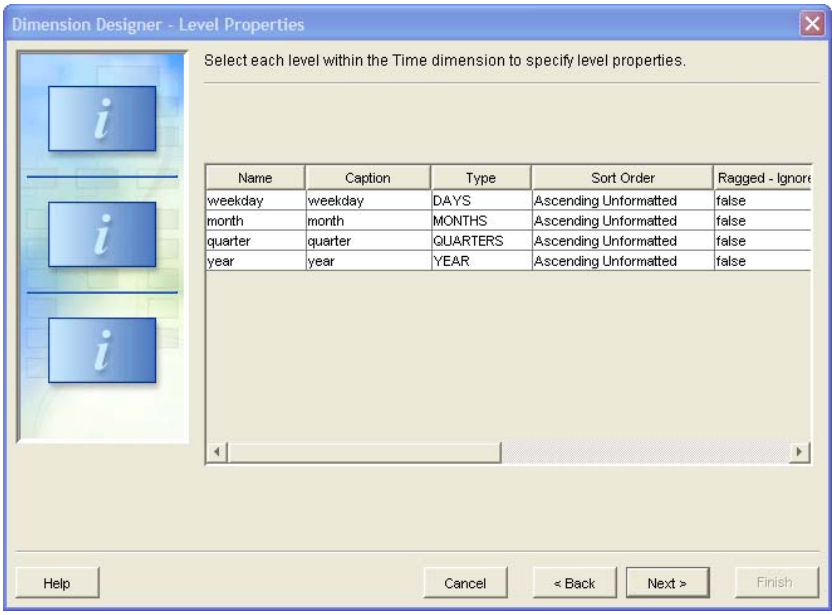
3. Click Next. In our workshop, we want to make sure that we allow users to drill through to the detail table, so we will select the **Use input table for Drill-through: Denormalized.**

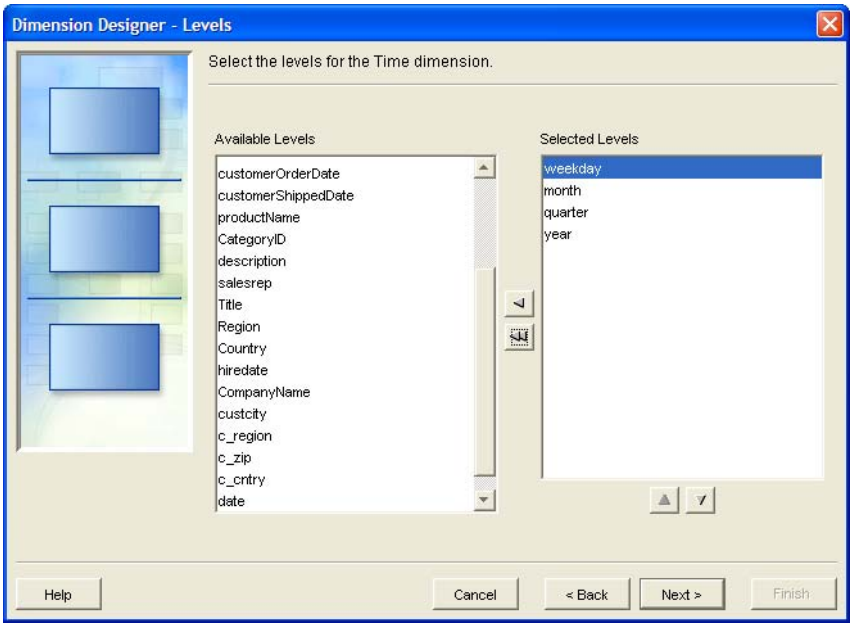


4. Click **Next**. In the next series of tasks, we are going to create the hierarchies for our cube. When presented with the **Define Dimensions, Hierarchies and Levels** page, click **Add**. We are going to create the following dimensions: customer, product, time and employee with the following levels and hierarchies.

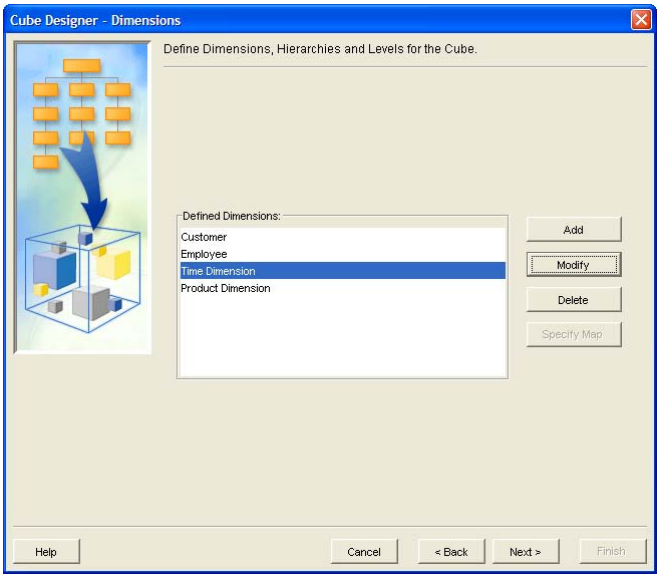
Dimension	Levels	Hierarchies
Customer	companyName custCity c_region c_cntry	Customer Hierarchy (with the 4 columns identified in level)
Employee	salesrep region country	Employee Hierarchy (with the 3 columns identified in level)
Time	weekday month quarter year	Time Hierarchy (with the weekday, Month, Quarter and Year)
Product	ProductName Description	Product Hierarchy (with the 2 columns identified in level)

Note that when we are editing the Level Properties for Time, we'll need to specify the Type for our variables. Note, you need to specify the type of TIME when creating the time dimension, in order for the the time dimension to work properly.

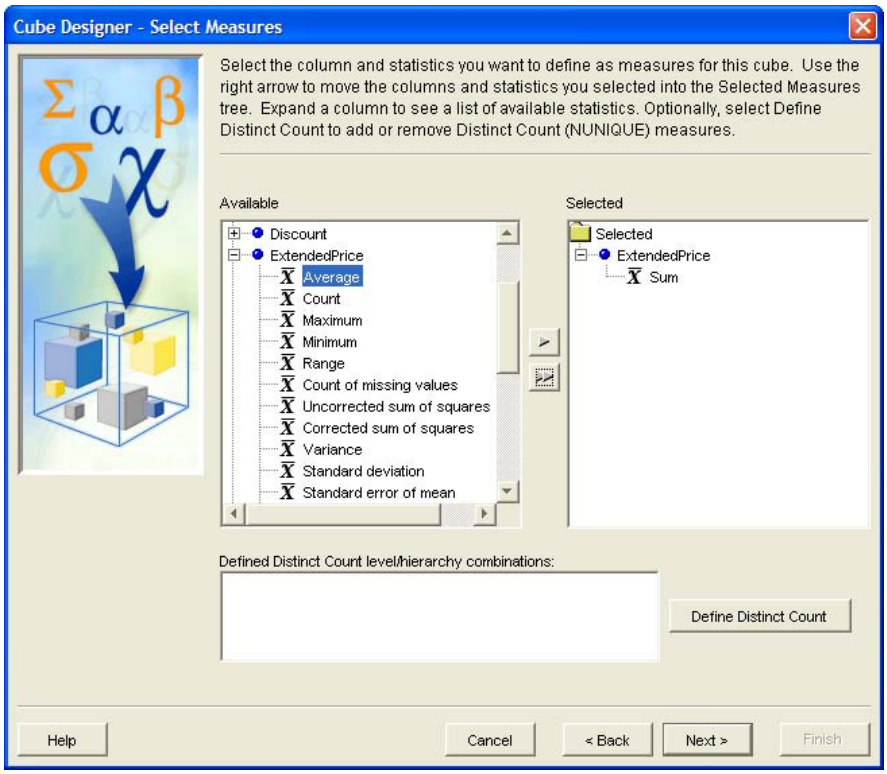




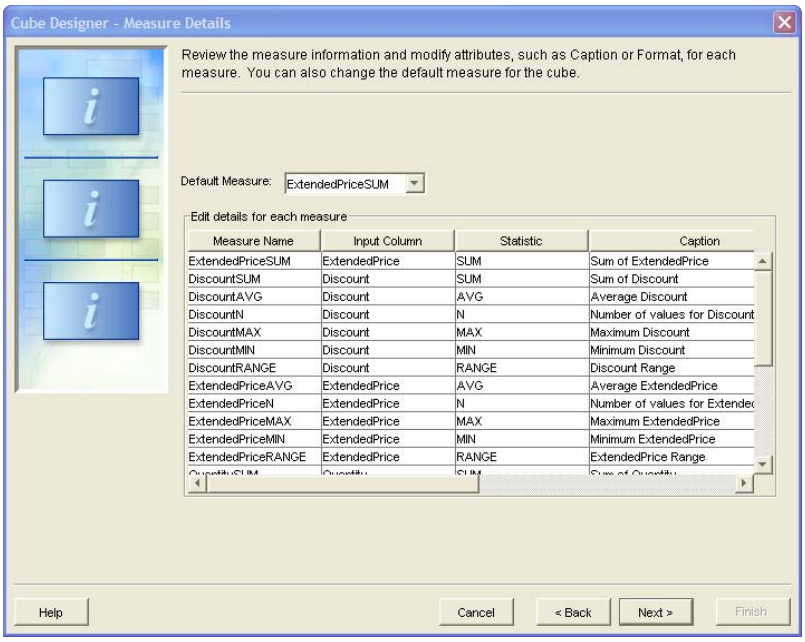
When you have completed adding the dimensions, levels and hierarchies, you should click **Next**.



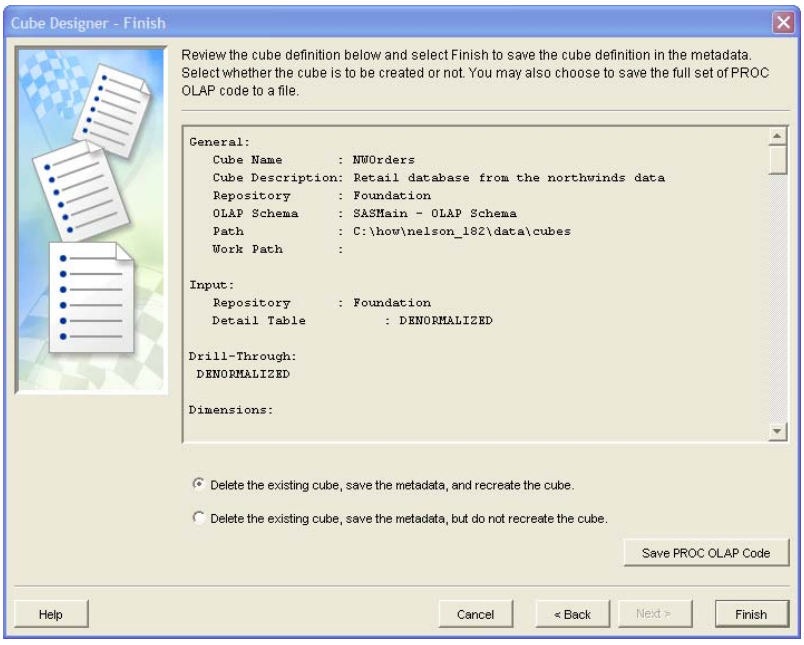
5. Select the measures as shown in the screen below.



6. Our default measure should be **ExtendedPriceSum**. Click **Next**.



7. The next few screens we will leave alone (Define members, Additional Aggregations).
8. When presented with the Summary screen (Finish), review the information and click Finish.



Task #4: Viewing the OLAP Cube

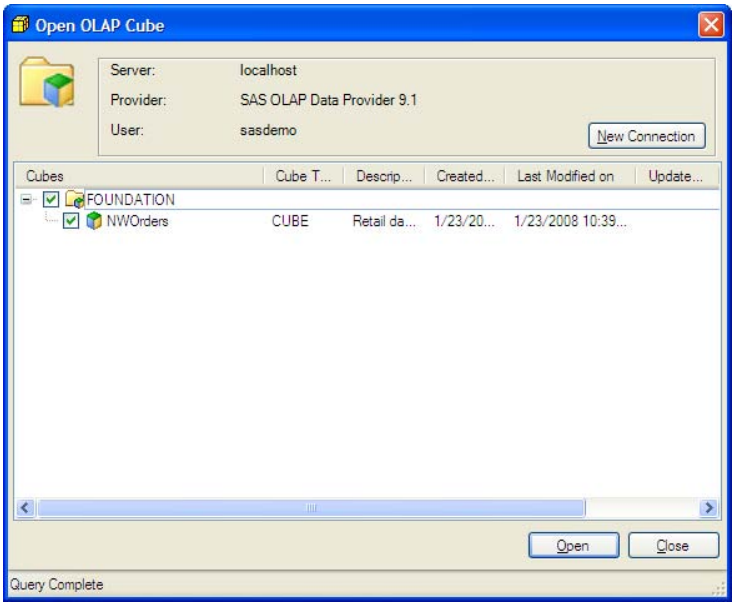
TASK 4 - STEP A: OPEN THE OLAP CUBE IN ENTERPRISE GUIDE

1. To open a SAS cube in Enterprise Guide, start Enterprise Guide and login (if required).
2. Select **File → Open → OLAP Cube**. The **OLAP Cube Login** window opens.

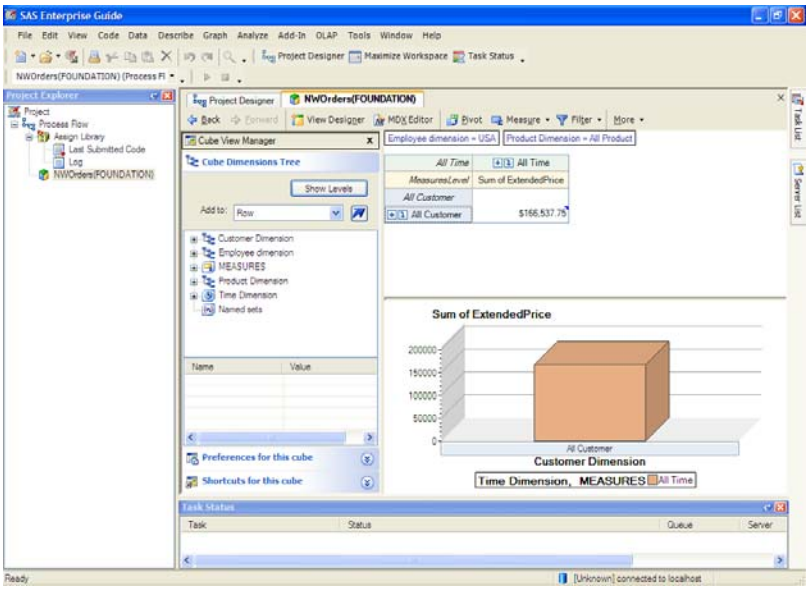
3. In the **OLAP Server Name** box, specify the name of the OLAP server that contains the cube that you want to open. (for this workshop, we will use the name of the machine or **localhost**.)

4. From the **Provider** drop-down list, select the provider. We will use : SAS OLAP Provider 9.1

5. Provide the userid and password (userid: sgf\sasdemo and a password: how). After selecting OK, you should see the following screen.



6. In the **Open OLAP Cube** window, select the check box next to the cube that you want to open and click **Open**. The cube opens in the **OLAP Analyzer** window.



Exercise Summary

In the steps above, we built an OLAP Cube using SAS OLAP Cube Studio. We followed the steps of designing the cube, preparing the data and then building the cube interactively. We hope that you continue your education by trying to repeat the process at home and playing around with changing dimensions, levels and hierarchies and even building the cube from the star schema instead of the denormalized dataset. In the next section, we will highlight some additional reading on topics that you will no doubt want to learn about as you really try to use this for real work.

Advanced Topics

While the intention of this workshop is not to leave the reader completely satiated with regard to the entire menu of capabilities found within Data Integration Studio and SAS OLAP Cube Studio, we did want to leave you with a few pointers to how you can find more information on some important topics. We have outlined what we believe are the key tasks that are required to really implement an OLAP solution. Note: some of these links are reproduced from an earlier paper from the author.

Concept	Description / Purpose	References
Data Integration Studio	The references here are general resources if you want more information on the product and general data warehouse design concepts.	http://www.sas.com/technologies/dw/etl/etlstudio/factsheet.pdf http://www.sas.com/technologies/dw/etl/etlstudio/ http://support.sas.com/software/91x/etlstudiowhatsnew.htm http://www.sas.com/ctx/whitepapers/whitepapers.jsp?code=226 http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002761998.htm http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002734624.htm http://freedatawarehouse.com/tutorials/dmtutorial/Dimensional%20Modeling%20Tutorial.aspx
OLAP Cube Studio	Information about OLAP Cube Studio, Enterprise Guide OLAP Analyzer, SAS viewers for OLAP cubes and exploiting OLAP cubes with MDX.	http://www.sas.com/technologies/bi/tourext/olapstudio_itour_flash.html http://www.sas.com/technologies/bi/query_reporting/guide/index.html http://www.sas.com/technologies/bi/query_reporting/webolapviewer/ http://www.sas.com/technologies/bi/olap/index.html http://www.sas.com/offices/europe/uk/downloads/olap.pdf http://www2.sas.com/proceedings/sugi28/186-28.pdf http://support.sas.com/rnd/papers/sugi31/ESRIandOLAP.pdf http://support.sas.com/documentation/onlinedoc/91pdf/sasdoc_91/olap_mdx_7002.pdf http://support.sas.com/rnd/olap/index.html http://www.sqlmag.com/articles/index.cfm?articleid=5112&
Loading Techniques	DI Studio can effectively load data into a target table using any number of out of the box approaches. These include wipe and load (refresh), append or update. For Type 2 changes in dimensions, SAS has a transformation included. Of course for complex rules about slowly changing dimensions, you can always write your own code. Loading cubes initially is fairly straightforward, but feeding them updates is more challenging if you aren't familiar with the process. The links here highlight some of those challenges.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002764619.htm http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002982401.htm http://support.sas.com/rnd/olap/Non-leafmemberload.pdf http://etl-tools.info/en/scd.html http://www2.sas.com/proceedings/forum2007/322-2007.pdf

Concept	Description / Purpose	References
Scheduling	DI Studio comes with LSF Scheduler from Platform Computing. To schedule a job from DI Studio, you simply provide the deployment information when you right click the job and choose deploy for scheduling.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002734641.htm#a002764629 http://support.sas.com/onlinedoc/913/getDoc/en/bicag.hlp/a003069776.htm http://support.sas.com/onlinedoc/913/getDoc/en/bicag.hlp/a002650381.htm http://support.sas.com/onlinedoc/913/getDoc/en/mcug.hlp/a002688103.htm
Exception Handling	As with any good data warehouse, you will want to configure your code in such a way as to provide proactive notification that something went wrong (or some cases - right). There are a number of options available in DI Studio and third party options that allow for this.	http://support.sas.com/onlinedoc/913/getDoc/en/bicag.hlp/a002938721.htm http://www.thotwave.com/products/eventsystem.jsp
Building cubes (Design)	Once the data warehouse is loaded, often aggregate tables will be constructed. To learn more about building cubes with SAS, refer to these sources. Loading techniques including information on best practices related to improving performance.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002764598.htm http://support.sas.com/rnd/scalability/papers/OLAP_Performance.pdf
Surrogate key generator	In data warehousing, it is beneficial to have a key for your fact tables that are not the same as your business keys. The Surrogate Key Generator transformation enables you to create a unique identifier for records, a surrogate key. The surrogate key can be used to perform operations that would be difficult or impossible to perform on the original key. For example, a numeric surrogate key could be generated for an alphanumeric original key, to make sorting easier.	http://www.dmreview.com/article_sub.cfm?articleId=6136 See the SAS help for DI Studio for the following topics: • Example: Load an Intersection Table and Add a Surrogate Key
Slowly changing dimensions	A technique described by Ralph Kimball that is used to track changes in a dimension table. A type 1 SCD is updated by writing a new value over an old value. A type 2 SCD is updated by creating a new row when a value changes in an old row. A type 3 SCD is updated by moving an old value into a new column and then writing a new value into the column that contains the most recent value.	See the SAS help for DI Studio for the following topics: • Maintaining Slowly Changing Dimensions http://www.dmreview.com/article_sub.cfm?articleId=2998 http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002982401.htm
User written components (transforms)	There is no doubt that at some point, you will need to do something different that what DI Studio has to offer. For that, we have the facility for writing your own extensions or custom transformations.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002762005.htm#a002766183 http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002804130.htm#a002774747 http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002767307.htm#a002807999 http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002805009.htm

Concept	Description / Purpose	References
Impact analysis	A search that seeks to identify the tables, columns, and transformations that would be affected by a change in a selected table or column. See also transformation, data lineage.	See the following topics in the DI Studio Help: • Using Impact Analysis and Reverse Impact Analysis
Promotion and team development	For setting up and managing team development environments. Metadata can be managed in such a way as to facilitate change management.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002792237.htm

References and Author Contact Information

References and Recommended Reading

Clover, L. and Ruben, E. (2004) *Introducing the New Rich-Client OLAP Analyzer from SAS -- SAS Enterprise Guide 3.0 OLAP Analyzer*. Presented at the SAS Users Group International Conference. SUGI 29. May, 2004.

Diamond, J. and Gagliano, T. (2002) *Accessing the Version 9 OLAP Server from Java*. Presented at the SAS Users Group International Conference. SUGI 27. April, 2002.

Ender, M. (2004) *Taking Your SAS/MDDB® Server Applications to the Next Level*. Presented at the SAS Users Group International Conference. SUGI 29. May, 2004.

Grasse, D. and Nelson, G. *"Base SAS vs. Data Integration Studio: Understanding ETL and the SAS tools used to support it"*. Invited Paper presented at the SAS Users Group International Conference. San Francisco, CA. March, 2006.

Henderson, G. (2001) *OLAP Best Practices What You Need to Consider When Building and Deploying an OLAP Application*. Presented at the SAS Users Group International Conference. SUGI 26. April, 2001.

Kimball, Ralph. The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. John Wiley & Sons, 1996

Kimball, Ralph. *"The 38 Subsystems of ETL: To create a successful data warehouse, rely on best practices, not intuition."* Intelligent Enterprise." December 4, 2004.

Kimball, Ralph and Conserta, Joe. The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data. John Wiley & Sons, 2004

Kimball, Ralph, Laura Reeves, Margy Ross, and Warren Thornthwaite. The Data Warehouse Lifecycle Toolkit: Tools and Techniques for Designing, Developing, and Deploying Data Warehouses John Wiley & Sons, 1998

Kimball, Ralph and Ross, Margy. The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling (Second Edition). John Wiley & Sons, 2002

Nelson, G. (1999) *Implementing a Dimensional Data Warehouse with the SAS System*. Invited Paper presented at the SAS Users Group International Conference. San Diego, CA. March, 1999.

Nelson, G. (2006) *A Pragmatic Programmers Introduction to Data Integration Studio: Hands on Workshop*. Hands on Workshop presented at the SAS Users Group International Conference. San Francisco, CA. March, 2006.

Nelson, G. (2007) *Planning for and Designing a Data Warehouse: A Hands-On Workshop*. Hands on Workshop presented at the SAS Global Forum Conference. Orlando, Florida. April, 2007.

Ressler, D. (2002) *V9 OLAP: An Architectural Overview*. Presented at the SAS Users Group International Conference. SUGI 27. April, 2002.

Weinberger, A. and Ender, M. (2000) *The Power of Hybrid OLAP in a Multidimensional World*. Presented at the SAS Users Group International Conference. SUGI 25. April, 2000.

Acknowledgements

I would like to thank Shawn Edney and Danny Grasse for their “thot-ful” and insightful comments on this manuscript.

Biography

Greg Nelson, President and CEO

Greg has recently started his third decade in the SAS eco-system as a programmer, analyst, architect, and teacher. Greg is the President and CEO of ThotWave Technologies where he supports customers in a variety of industries. Prior to ThotWave, Mr. Nelson spent several years in consulting, media and marketing research, database marketing, and large systems support. Mr. Nelson holds a B.A. in Psychology and PhD level work in Social Psychology and Quantitative Methods.

About ThotWave

ThotWave Technologies, LLC is a Cary, NC-based consultancy and a market leader in real-time decision support, specializing in regulated industries, such as life sciences, energy and financial services. ThotWave works at the juncture of business and technology to help companies improve their operational and strategic performance and recognizes the difference between simply accessing data and making data work for business. Through products, partnerships and services, ThotWave enables businesses to leverage data for faster, more intelligent decision making.

Contact information

Your comments and questions are valued and encouraged. Contact the authors at:

Greg Nelson greg@thotwave.com

ThotWave Technologies, LLC

2504 Kildaire Farm Road

Cary, NC 27511

(800) 584 2819

http://www.thotwave.com