Paper HW04

A Pragmatic Programmers Introduction to Data Integration Studio: Hands on Workshop

Gregory S. Nelson
ThotWave Technologies, Cary, North Carolina

Abstract

ETL is the process of moving data from a source system (such as operational systems or a table in a database) into a structure that supports analytics and reporting (target). This workshop will guide participants through a structured, hands-on exercise designed to give them a broad overview of what things we can accomplish with Data Integration Studio. Here we will prepare data for use by extracting data from an external file, creating transformations that enrich our data, combining it with other data for completeness and finally loading the data into tables that are part of a star schema. The goal of this workshop will be to get users comfortable with the tool and demonstrate its capability.

Introduction

One of the first tasks that we would like to cover is a basic understanding of where tools like Data Integration Studio (DIS) fits into the SAS 9 platform and what kinds of things we can use it for in our work.

DIS is a tool specifically written for a set of users that participate in the design and construction of a data warehouse. It is important to note that we won't attempt to try and define the term "data warehouse" expect to say that it is some data structure that is different from where the data originates. Whether we call it a data warehouse, data mart or data hut is irrelevant for this workshop. However, the goals that we hope to achieve by using a tool like this is simple: it supports the activities of those persons responsible for getting data out of the operational or transactional systems into a structure that can be used for reporting and analysis.

In this workshop, we will focus on how to use the tool, what some of its major functions are, and how we can use it to get data from the source to the target. All while paying attention to the rules and techniques that we use to perform those tasks.

For brevity's sake, this is not a paper that will necessarily stand on its own without the aid of the exercises and workshop discussion. There are a number of other authors who have done that – either with regard to the role of ETL in data warehousing (Kimball, 2004) or with respect the Data Integration Studio (Grasse and Nelson, 2006).

The SAS 9 Platform: an Architectural Perspective

Before we dive into the tool, it is important to understand how DIS fits 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.

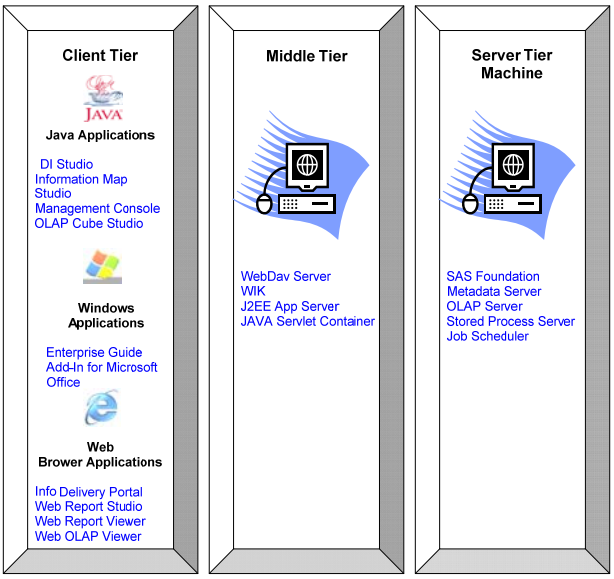


Figure 1. SAS Multi-Tier Architecture (Clients and Servers)

The SAS clients we will discuss in this context include Data Integration Studio, the SAS Management Console and SAS Enterprise Guide. Each of these play an important role in our data warehousing activities and talk 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 – Data Integration Studio. In order to do that, we need at least 2 servers: the metadata server and a workspace server. This should seem relatively transparent to us as users because we just launch the application, select our server and provide our credentials – and viola! – we're in and the magic is all invisible to us. But just like the first time we learned about the DATA step, we want to understand why things happen – so just as we learned about the Program Data Vector (PDV) in our first DATA step class, we will learn a little bit about how these things work behind the scenes.

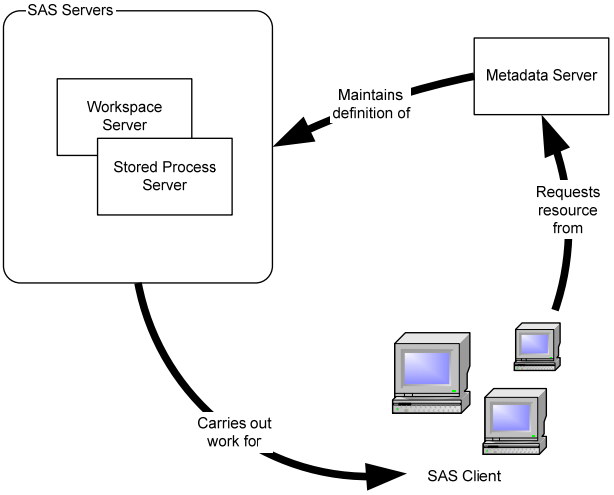


Figure 2. SAS 9 Logical Architecture

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

The Workshop

Thus far in this paper, we've set the stage for the workshop. In that vein, we wanted to share some background on the data that we are going to use for the exercises below.

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 that most of us have some familiarity with. After all, building a warehouse is typically done by extracting information from a transactional system and denormalizing the data and putting 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 the fact that we used this same database in an early paper on dimensional data warehousing (Nelson, 1999). In that paper we described how you would approach ETL from a classical perspective with SAS BASE. For a more detailed description of dimensional data warehousing techniques and its application to SAS – we refer you to that 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.

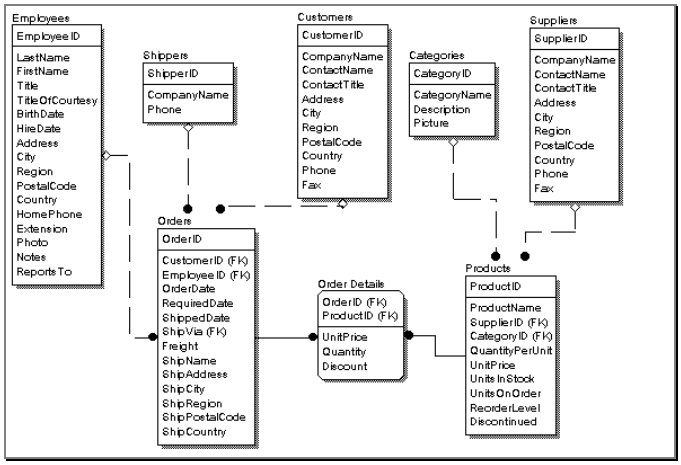


Figure 3. Main subject Area ERD for the OLTP Database (Source)

The diagram shows the relationships between tables in our database. Notice that the Order 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

As we prepare data for analysis and reporting, we want our data model design to reflect the careful balance between the needs of both the end user (consumer) and the technical and storage requirements. Therefore, we will create a star schema more appropriate for our needs

In the figure below, we depict the star-schema for our data mart based on the North wind database. We can easily see the main subject areas and relationships.

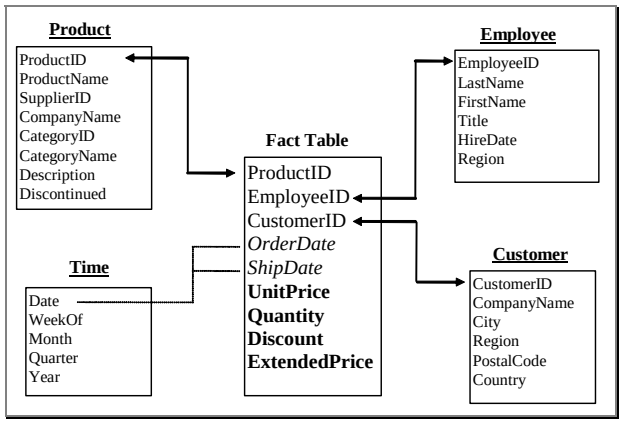


Figure 4. Star-schema model for the Northwinds Trading Company data mart.

The star schema model in Figure 2 is based on a central Fact Table surrounded by any number of Dimension Tables. The fact table has few columns or variables but many observations. These are linked by dimension tables that are typically short and wide, that is, many columns, and few observations.

THE BUSINESS PROBLEM

More often than not, we as technologists forget the reason that we set out to solve the problem. In this case, your task is simply to gain familiarity with a tool – Data Integration Studio. But remember, a solid understanding of the requirements for reporting and analysis are key to being successful. Just because we don't talk about the reports or how we might want to use the data doesn't mean that you should forget those tasks when diving into a new project. One of the ways that we described in the earlier paper (Nelson, 1999) was to learn to ask yourself: what questions do we want to answer of the data?

In our simple example described here, we want to know what the sales are for our fictitious company – Northwind Traders. We want to be able to generate reports and graphics about those sales by products and product categories, over time and categorized by customer attributes such as geographies.

Exercises

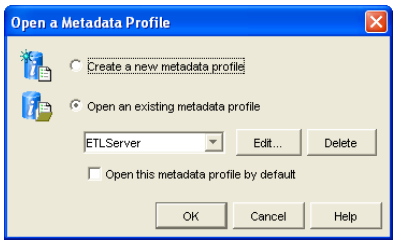
The remainder of this paper supports the steps that we would like you to follow in the Hands on Workshop. From a high level, we are going to run through a simple problem that is typical of the ETL programmer. Before we can do that effectively, we need to do some training on the basics of the tool. Once we go through how the tool works, we will run through our simple example. In this example, we will do 3 things:

1. Understanding the user interface
    - Authenticate ourselves to the metadata server
    - Review the user interface
    - Review some basic definitions
2. Creating libraries (references to data)
    - Create the connection to the ODBC data source (via Microsoft ODBC Administrator)
    - Creating the library in DI Studio
    - Explore the data libraries that we have available to us (source)
    - Add a new data library definition for our source and target tables
    - Create metadata for both the source and target tables
3. Create a new “job” that consists of:
    - Bringing in selected columns that we need from our source tables
    - Performing some transformations in order to enrich the data
    - Loading the fact tables through wipe and load (refresh)
    - Loading the dimension tables using special logic
    - Once the process flow has been defined, we will run the job and evaluate what happened including looking at the metadata that was generated

Task #1: Understanding the user interface

TASK 1 – STEP A: LOGGING IN (AUTHENTICATION)

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



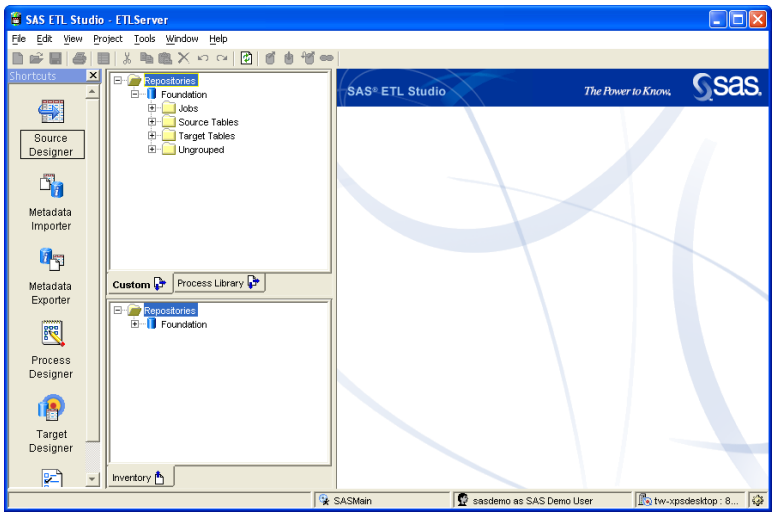
2. Select the correct metadata profile or create a new one and then provide your credentials.

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

TASK 1 – STEP B: UNDERSTANDING THE DI STUDIO INTERFACES

A. The Desktop

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



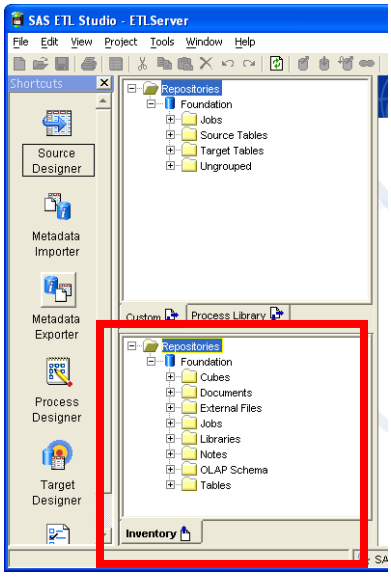
Some of these include: shortcuts, tree viewers and process editor.

B. Shortcuts

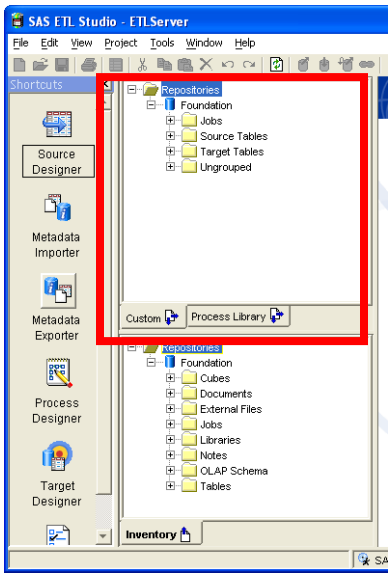
The shortcut bar is an optional pane of task icons on the left side of the SAS DI Studio desktop. Each icon displays a commonly-used window, wizard, or a selection window for wizards.

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

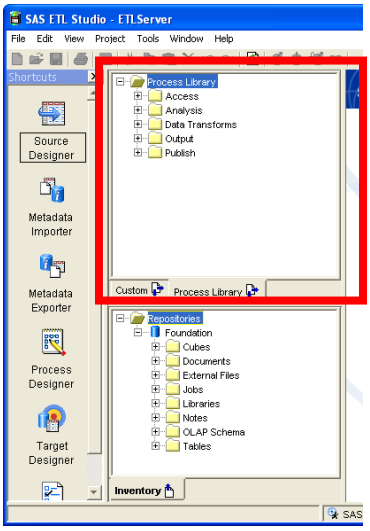
1. While your screen may look different, there are three primary tree views that we will use – these include:
2. **Inventory** – It organizes objects into a set of default groups, such as tables for all tables in a repository and Cubes for all cubes in a repository.



3. **Custom** – It enables you to create user-defined groups (folders and subfolders) that you can use to organize metadata objects in a way that is convenient for you.



4. **Process** – It displays a collection of transformation templates. A transformation template is a process flow diagram that includes drop zones for metadata that the user must supply.



D. Process Editor

To the right of the tree viewers, we see a large blank area. Later in this workshop we will see that this becomes our main work area where we define our process flows. More on this later.

TASK 1 – STEP C: A FEW DEFINITIONS

- **Sources –** in the context of data warehousing, a source is where the data originates.
- **Targets –** this is where the data will end up (this is the "target" table that will make up part of the data warehouse/ data mart.
- **Transformations –**a transformation is a metadata object that specifies how to extract data, transform data, or load data into data stores. Think of these as template where we fill in the blank. Each transformation that you specify in a process flow diagram generates or retrieves SAS code. You can specify user-written code for any transformation in a process flow diagram.
- **Jobs or Process Flows –** process flows is a visual depiction of what a job sequence looks like. It can contain any number of inputs and outputs. Inputs can be data sets, external files or databases and outputs can be tables or reports.
- **Source code** (user written or DIS generated) – Because DI Studio is built on top of the SAS 9 platform and we have the ability to use the whole of SAS, we can write our own code and include this in any transformation that we use. Source code is essentially SAS code.
- **Metadata** (and metadata objects) – metadata is "data about data" – in the context of DI Studio, metadata can help us define servers, schemas, libraries, tables, process flows, etc.
- **Authentication –**Authentication is the means by which we can guarantee that a person is who they say they are. It's important to realize that SAS does not do this - authentication is carried out by a third party, such as MSAD, or the UNIX host, or a database such as

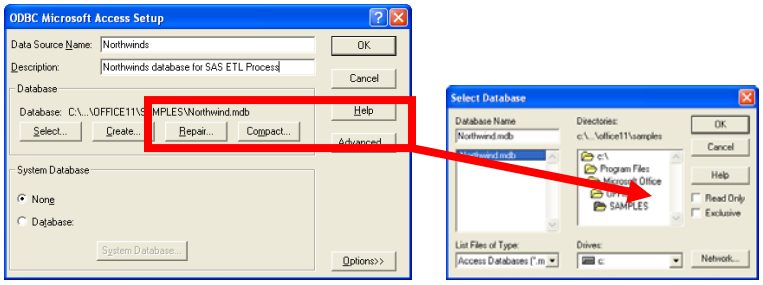
Oracle. The means by which we authenticate can be by providing a user ID and password; a fingerprint scan; or even a retinal scan. Once we (or the SAS Server) can trust the fact that a user is who they say they are, we can make decisions about what resources this known person has access to, which seamlessly leads us into the subject of Authorization.

- **Projects and repositories –** a project in DI Studio is really just a visual way to group all of the artifacts that you have created visually. A repository is the metadata location where you want to store this information.
- **Change management –** refers to the concept of how we work as developers collaboratively and not step on each others toes (or code). In the SAS Open Metadata Architecture, this takes on the form of metadata source control, metadata promotion, and metadata replication.

Task #2: Creating libraries (references to data)

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

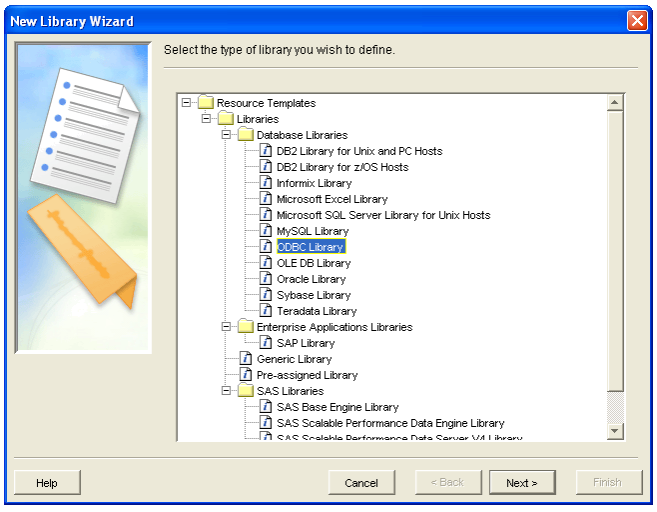
1. From the Desktop in Windows, select **Start → Control Panel → Administrative Tools → Data Sources (ODBC)**
2. Select the **System DSN** Tab → Click **Add** → Select **Microsoft Access Driver (.mdb)** → Click Finish
3. Complete the Data Source Name, description and select the Northwinds database (northwinds.mdb). For our workshop, we have place a copy of the Northwinds.mdb file in *C:\workshop\ws111\data\source*.



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

TASK 2 - STEP B: CREATE OUR SOURCE LIBRARY REFERENCES IN DI STUDIO

1. First we need to create a reference to the original source data (remember, our source data is in MS Access) and we will access the data via ODBC.
2. To do this, from the **Inventory** tab, expand the **Foundation** repository and right click on **Libraries** to create a **new Library**. The **New Library Wizard** will open.



3. We are going to choose **ODBC Library**, name the Library *Northwinds* and set the libref to *SRCDATA*. Before we can continue, we need to define an ODBC Library. Select **New** when it asks you for a **database server**. Complete the following prompts:

a. **Name:** *ODBC Server*

b. **Data Source Type:** *ODBC – Microsoft Access*

c. **Datasrc:** *Northwinds* (this is what we provided in the ODBC Administrator earlier)

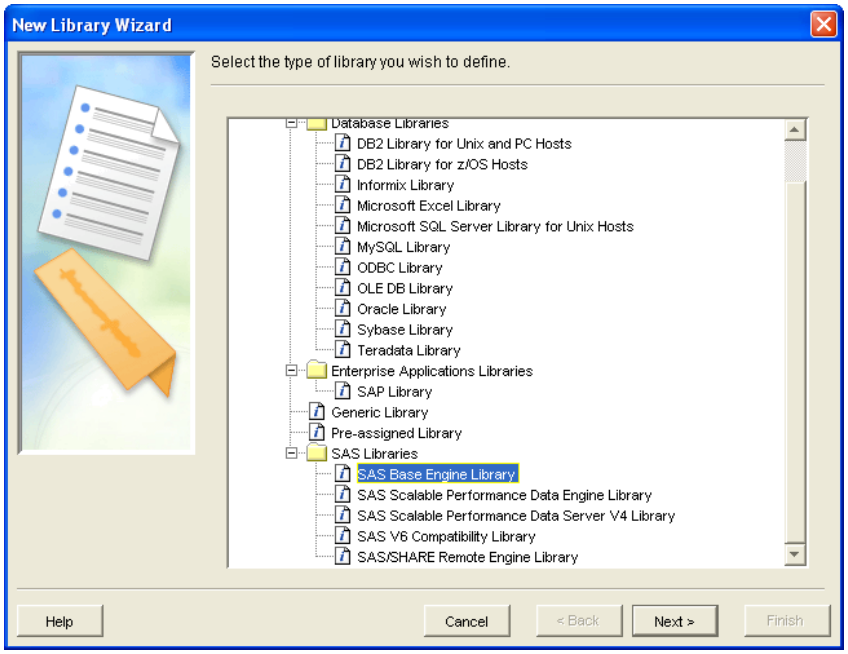
d. **Finish** the create database server wizard

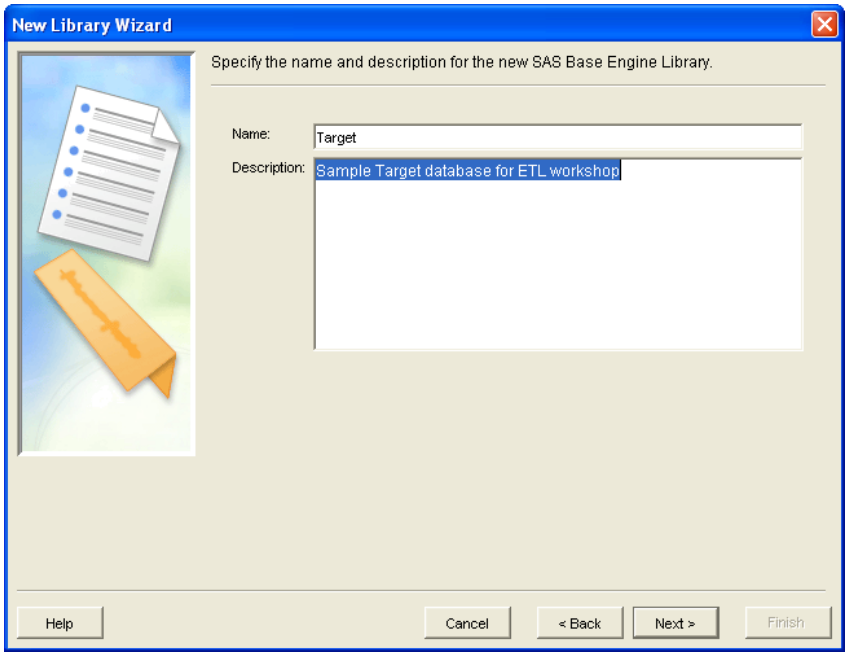
4. Once the ODBC Server has been created, we can complete the **New Library Wizard**. The only remaining task is to select the SAS Server where the library will be created (*SASMain*).

TASK 2 - STEP C: CREATE OUR TARGET LIBRARY REFERENCES IN DI STUDIO

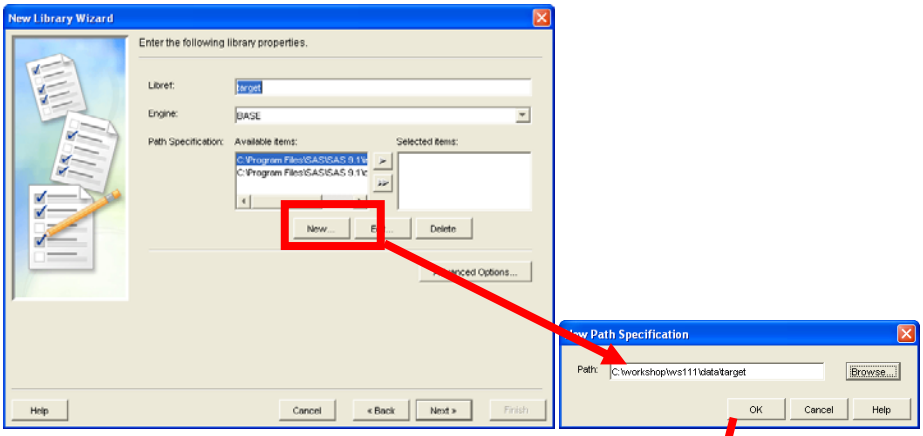
1. Create a new reference (libref) to our target database by creating a new library. To do that, simply select **Libraries** → *(right click)* **New**. And complete the wizard as shown below.

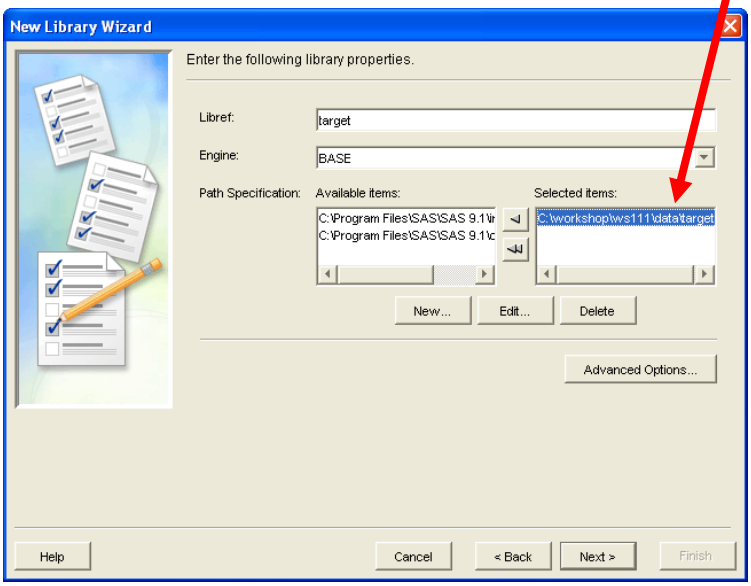
2. Select **SAS Base Engine Library → Next**





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





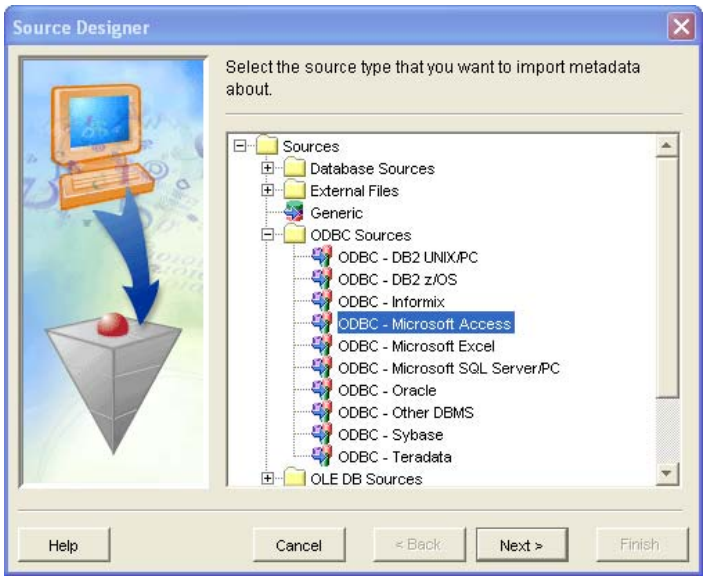
4. Select **SASMain → Next →** Review the information and click **Finish**.

TASK 2 – STEP D: CREATE METADATA FOR OUR SOURCE DATA

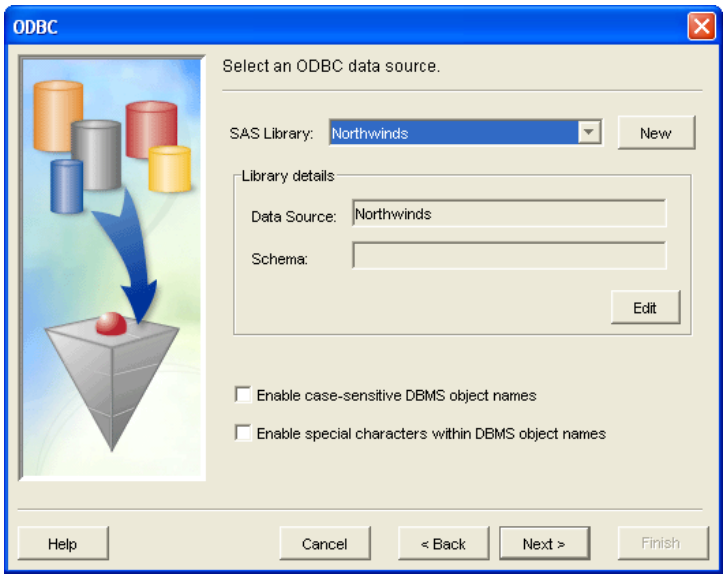
1. Under the **Foundation Repository** (Found on the **Inventory** tab), find **Libraries** and expand the libraries and you should see the *Northwinds library*.

2. Note that when we have created the reference to this data library, we don't have any table metadata beneath the library definition. We need to import this metadata with the Source Designer so that we can include them in our process flow.

3. On the left hand side, find **Source Designer** and click once. This will start the **Source Designer Wizard**. Expand the **ODBC Sources** and select **ODBC – Microsoft Access** and then click **Next**.



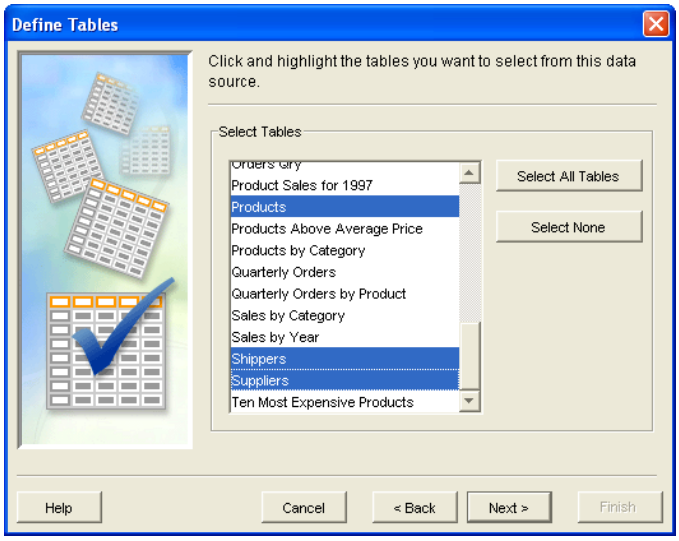
4. Select the ***Northwinds*** data as our **SAS Library** and click **Next**.



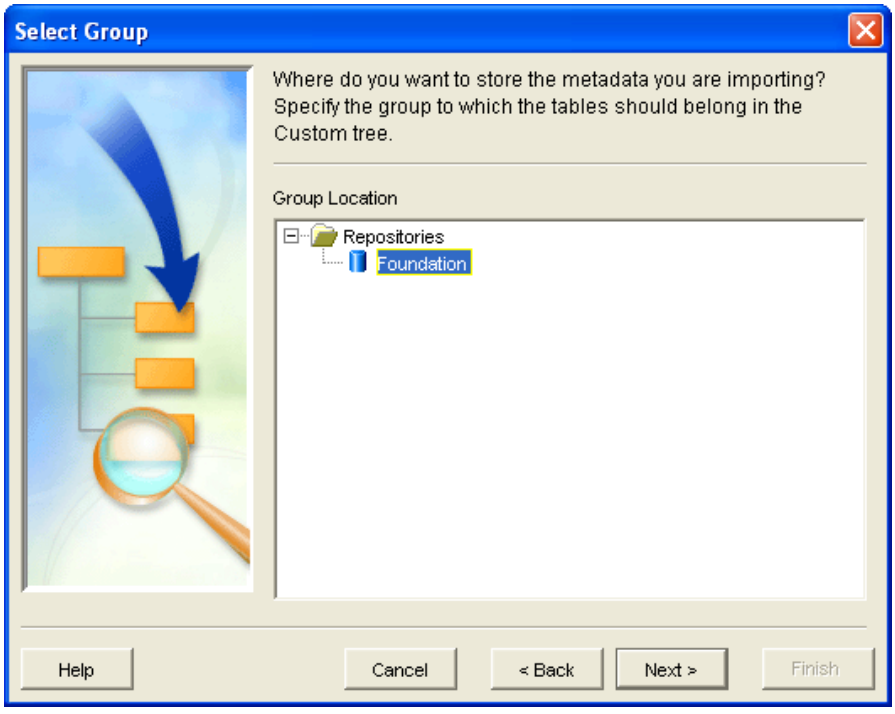
5. Our first Task is to only bring in the source tables that we will be using. In our case, we want the following tables:

   - Categories

- Customers
- Employees
- Order Detail
- Orders
- Products
- Shippers
- Suppliers

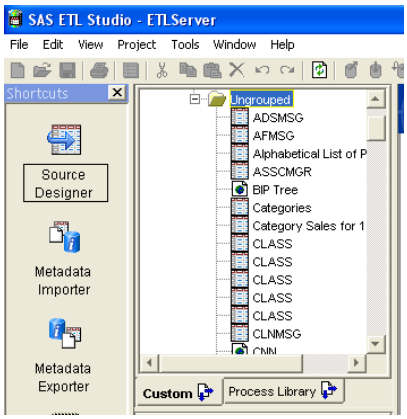


6. Select the correct tables then click Next.



7. Select the **Foundation** repository to store our metadata then click **Next**. Review the confirmation screen and click **Finish**.

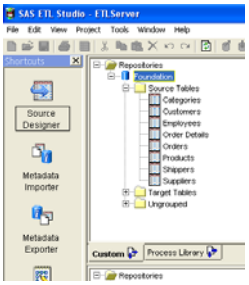
8. Note that these tables were added to our Foundation Repository (as metadata) but they are still ungrouped which makes it difficult to view if we have a lot of external data referenced in the metadata.



9. We want to create two new folders to "Group" our metadata. To do that, we want to right click on the **Foundation** repository and select **New Group**.

10. Create a new folder called **Target Tables** and one called **Source Tables**.

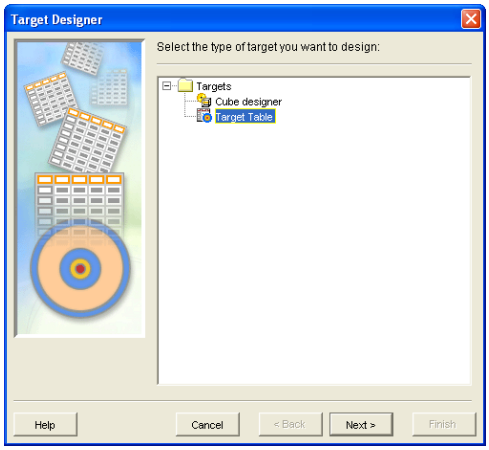
11. Move the tables you created earlier into the folder called **Source Tables**. (To move a table, simply right click on the object and select **Group**... then select the **Source Tables** folder from the hierarchy.



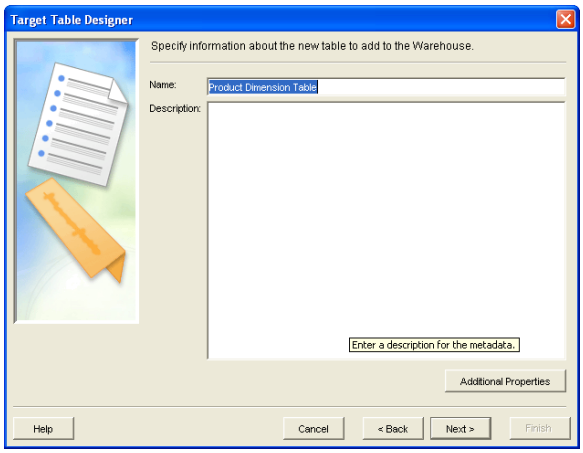
TASK 2 – STEP E: CREATE METADATA FOR TARGETS

1. To create the metadata for the target tables, we need to utilize the Target Designer.

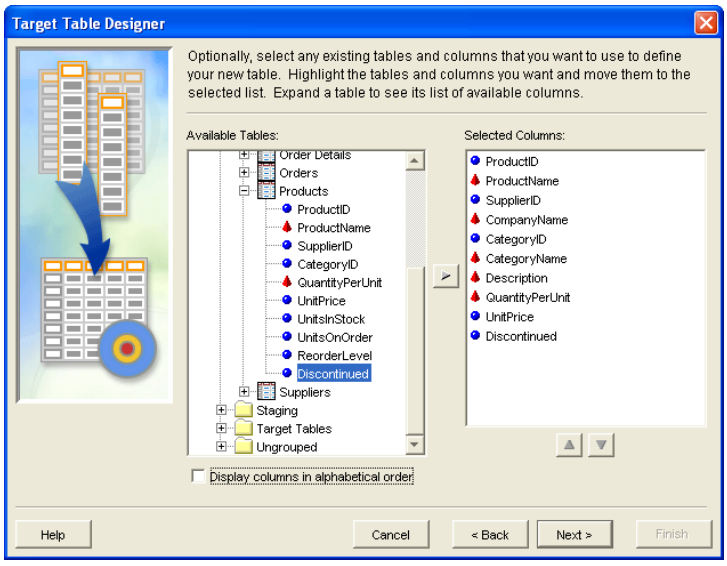
2. Select the **Target Designer** from the left hand pane, select **Target Table** and click **Next**.



3. The first table we will create is the one of the dimension tables. Based on our data model outlined above, we will call this our *Product Dimension Table*. Click **Next**.

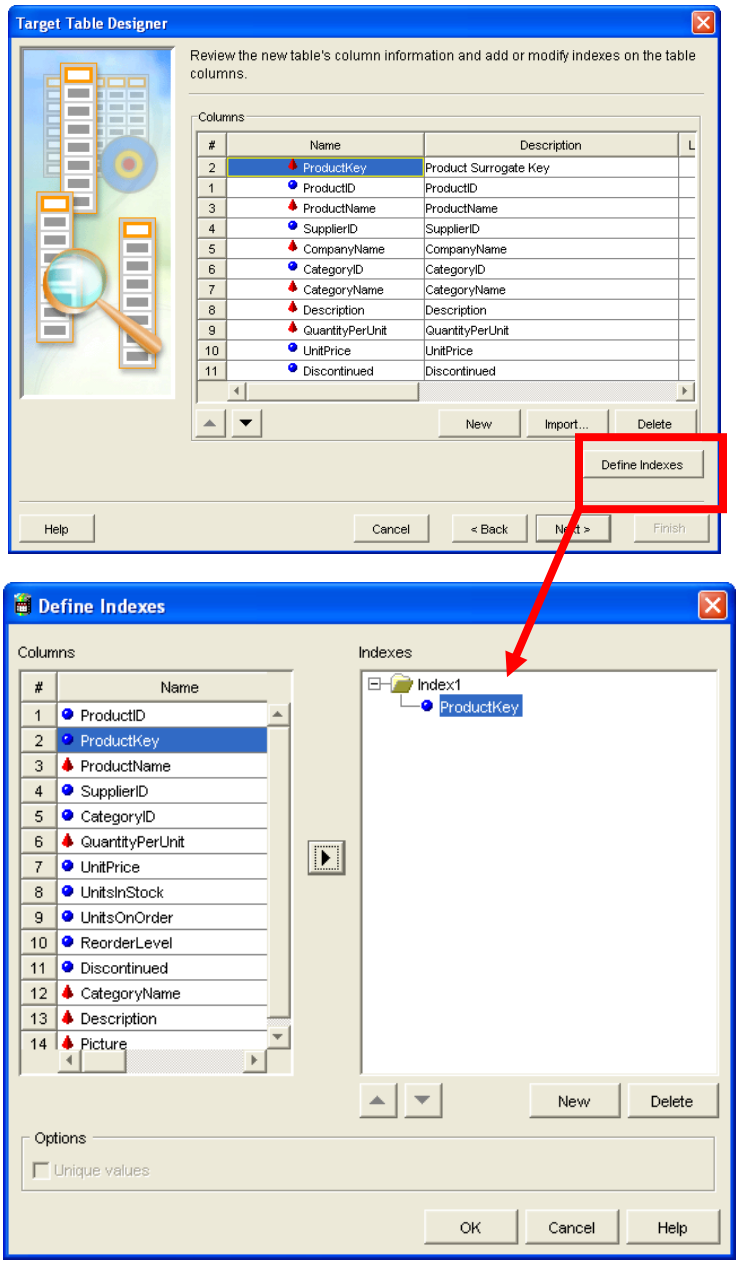


4. Next we have to bring in metadata from the source tables that will help us determine the correct lengths, formats and variable types. We know that product information is actually contained in two different tables in the source system: *Products* and *Categories*. So we select the columns we want from each of those tables and select them (shown on the right).

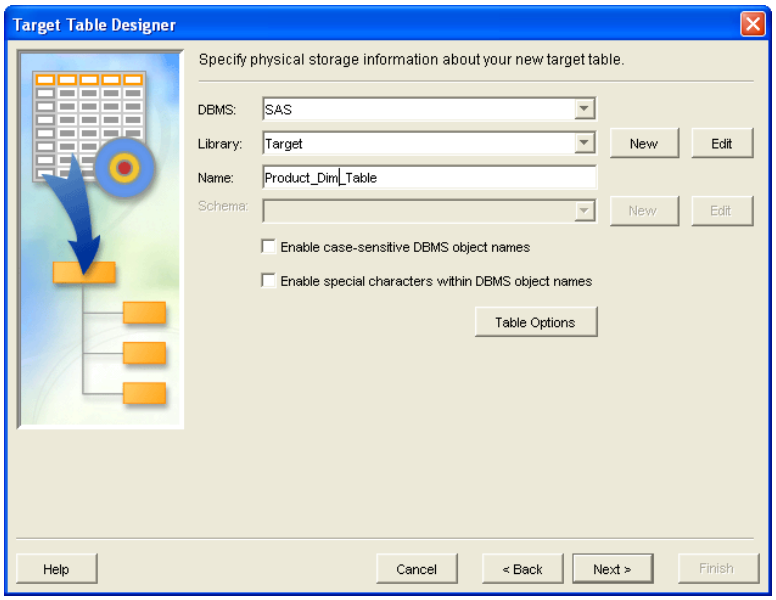


5. Select Next and we now have an opportunity to change the names of variables as well as other metadata. For our example, we will do a few things:

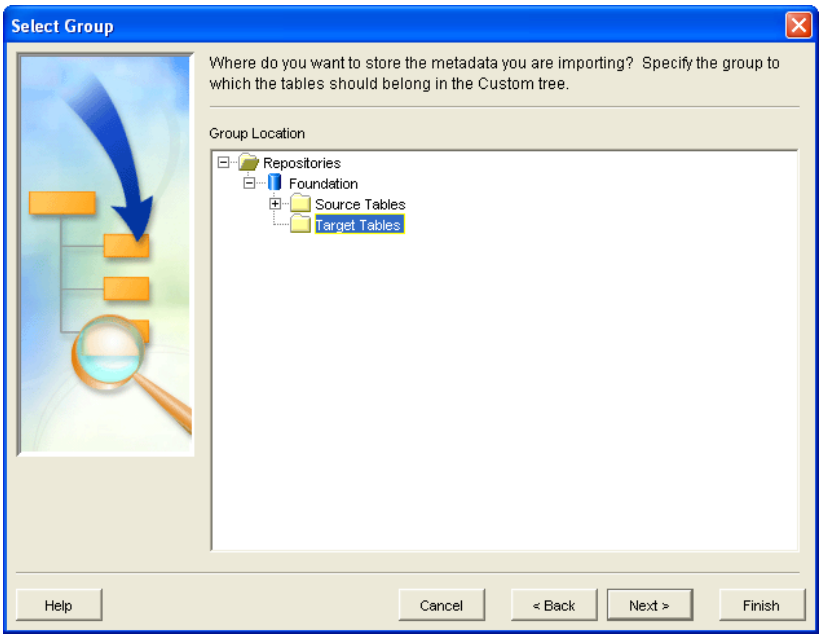
    a. Create a new field that will act as our surrogate key so that we don't have to be tied to the original source system ID. (Click **New** beneath the column information and create a new column called **ProductKey** with a description of Product Surrogate Key). We also want to scroll to the right and change our new variable from Character to Numeric and create a new Index called **ProductKey** with **ProductKey** as our index variable.



6. Next we want to set the options on this newly created table. For this table (*Product_Dim_Table*), we want to store this in the **Target** library as a **SAS dataset**.



7. Next select where we want to store the metadata. We should choose the *Target Tables* group.



8. And Finish the wizard after the confirmation screen appears.

9. At this point only the metadata exists for this next Target Table. We will soon learn how to actually populate this table. But first, we need to follow the same process to creating the other dimension and fact tables. The next three steps describes what we need in each of these tables. Note: we will not complete the steps to load the remaining dimension tables in this workshop – this is left to the student to complete on their own.

TASK 2 – STEP F: EMPLOYEE_DIM_TABLE (EMPLOYEE DIMENSION TABLE)

- Source data: All columns from Employee table

- New column: EmployeeKey (Description: Employee Surrogate Key)

TASK 2 – STEP G: CUSTOMER_DIM_TABLE (CUSTOMER DIMENSION TABLE)

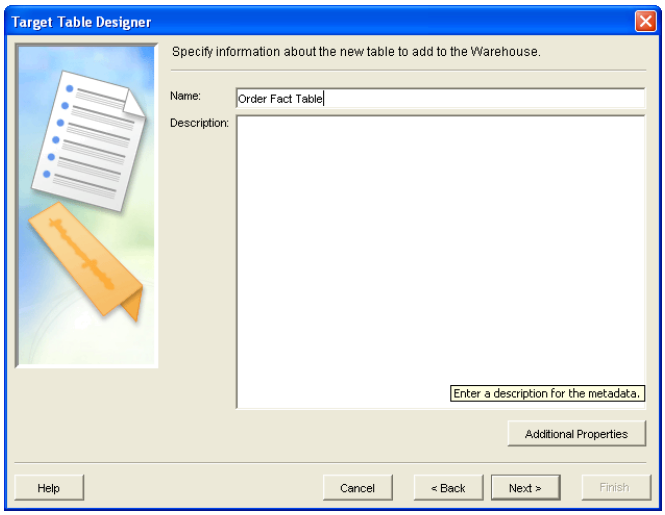
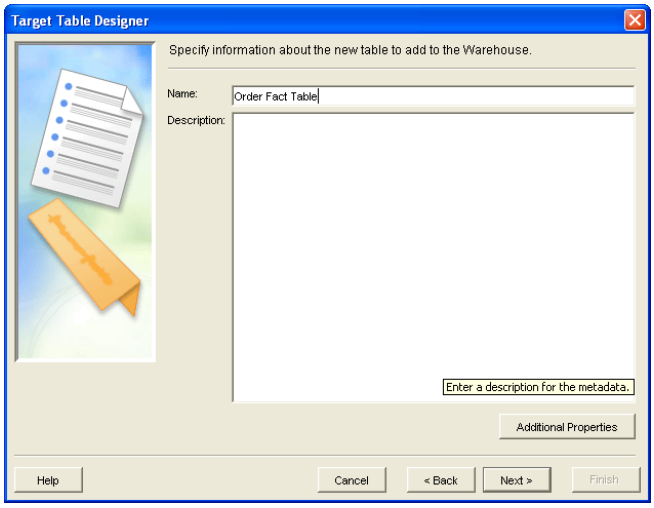
- Source data: All columns from Customer table
- New column: EmployeeKey (Description: Employee Surrogate Key)

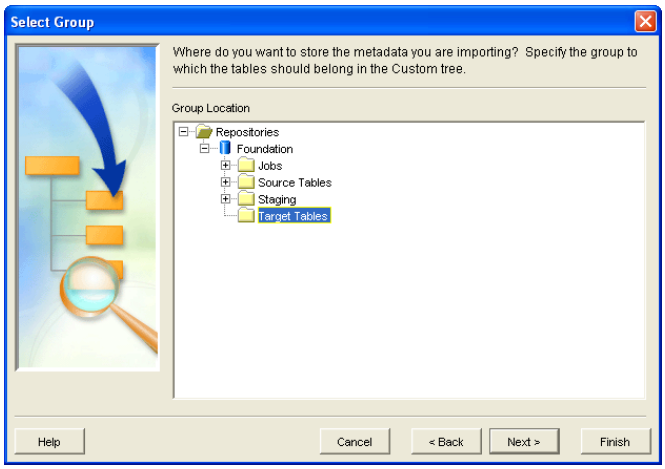
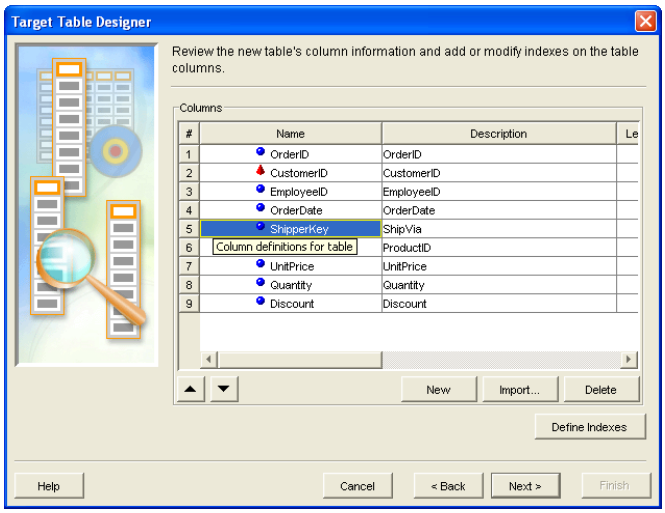
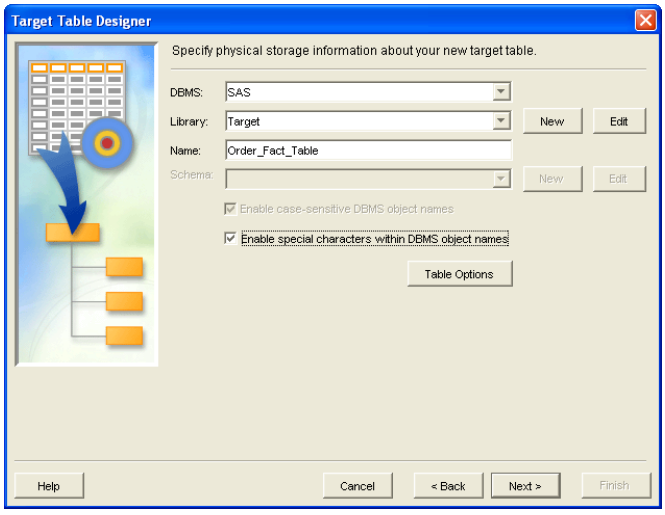
TASK 2 – STEP H: DATE_DIM_TABLE

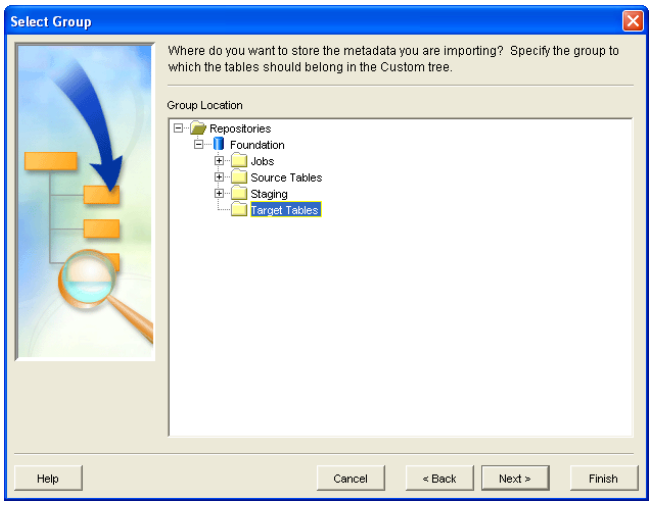
Column	Description	Variable Type	Length	Format	Informat
DateKey (Index)	Date Surrogate Key	Numeric	8		
Date	Date	Numeric	8	Datetime20.	Datetime20.
Week	Week Number	Numeric	8	8.	
Month	Month	Numeric	8	Month.	
Qtr	Quarter	Numeric	8	Qtr.	
Year	Year	Numeric	8	Year.	

TASK 2 – STEP I: ORDER_FACT_TABLE

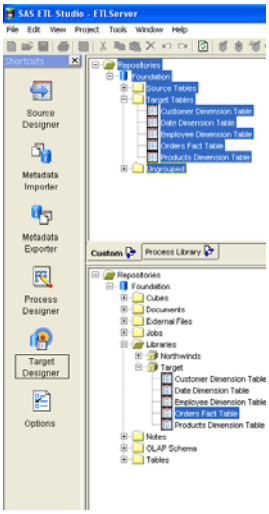
1. Let's load the Orders Fact Table:

Step	Description	Screen shot
1	New job wizard (from the **Target Designer**)	
2	Select the tables to be loaded	

3	Select the location of where we want to store the metadata. Once we confirm our choices, click finish.	
4.	Modify the column ShipVia to ShipperKey	
5.	Select the correct location for the table. Be sure to “Enable special characters within DBMS object names”.	

6.	Store the metadata in the Targets folder	

10. When we have finished creating the metadata for our target tables, we should see a listing of tables beneath the Target Table grouping in our Foundation repository.



Task #3: Loading the Data Warehouse: Creating the Process Flow

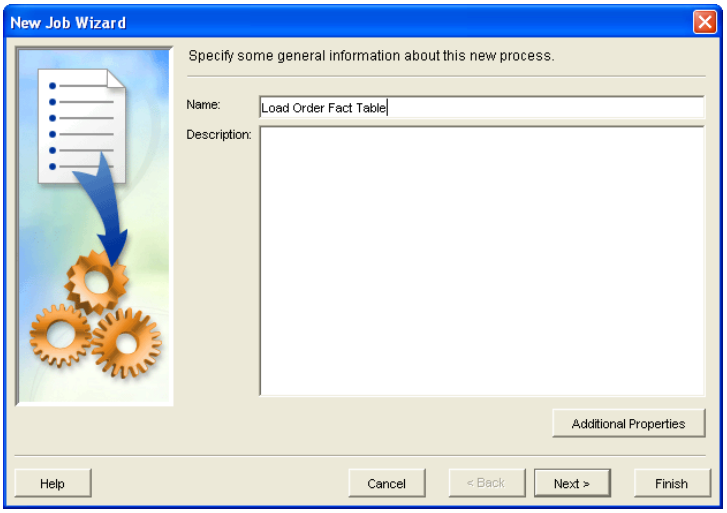
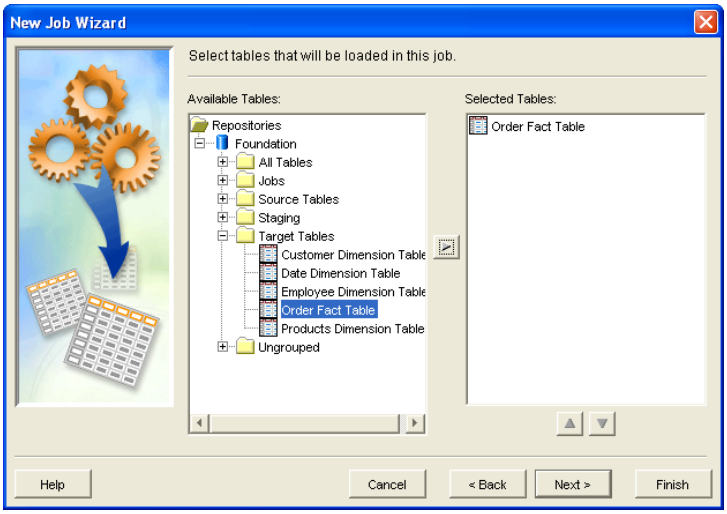
Now that we have created all of the metadata that we need for the Source and Target tables, we are ready to begin creating the job that will actually populate our data mart.

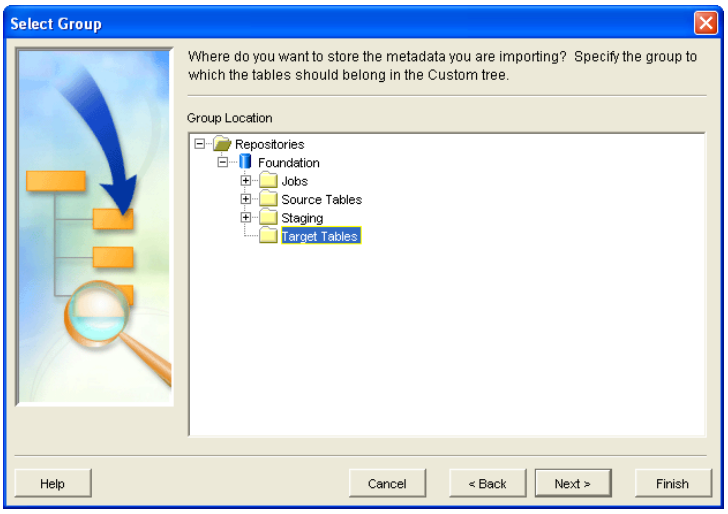
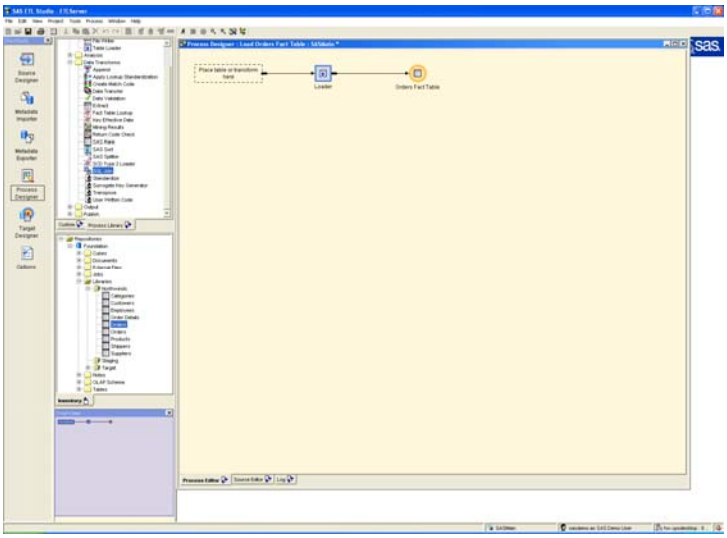
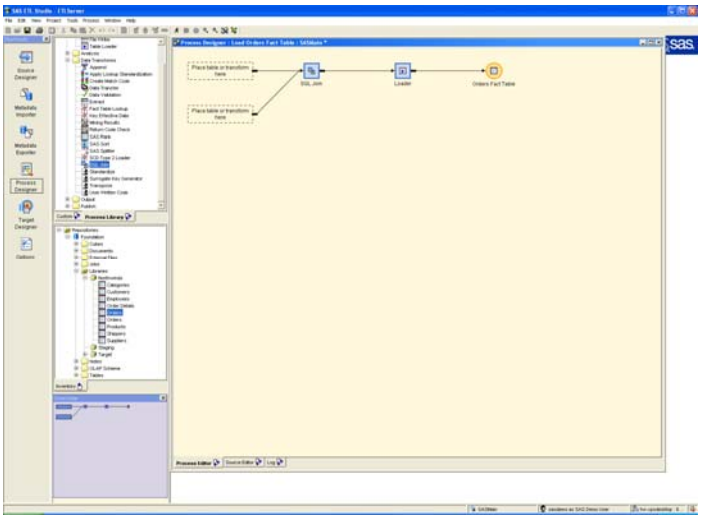
We will first populate the dimension tables. The only one that will be a little different is the Date Dimension table since we just need to create a range of dates along with the other columns and that is not really something we load from the source tables. This also gives us a chance to create our own custom code. On your own, write a custom transformation that will load a Date dimension table.

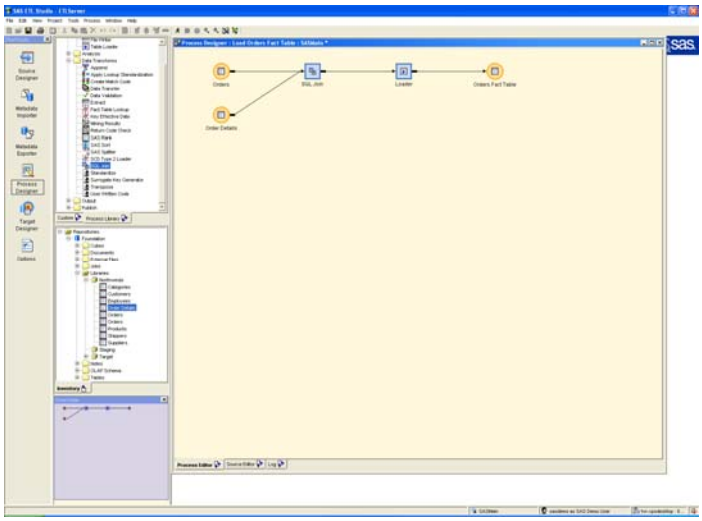
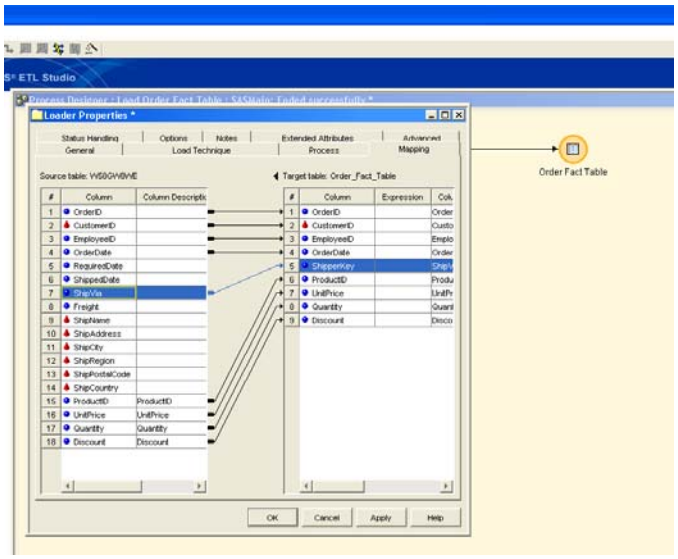
The first step is to create a new job. To do that, we select Process Designer from the left hand pane. We will create one of these for loading the Fact table and each of the Dimension Tables (Customer Dimension, Employee Dimension, Date Dimension and the Products Dimension Table.)

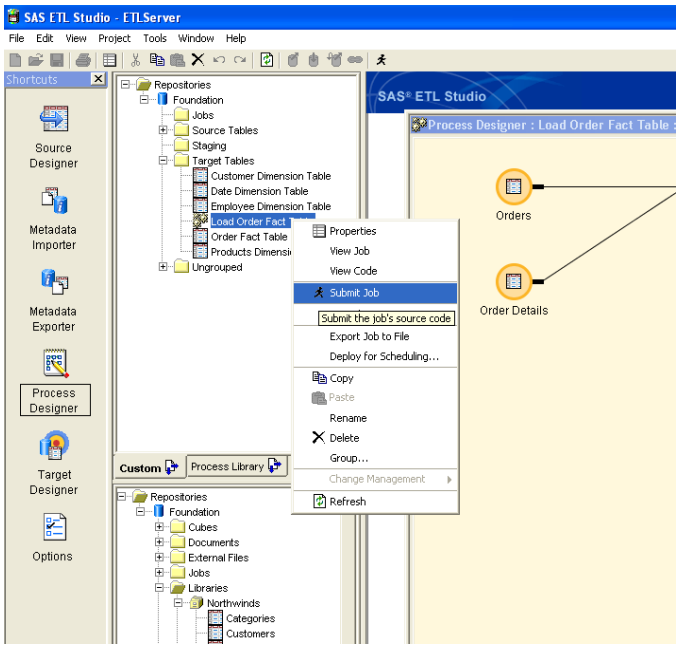
TASK 3 – STEP A: CREATE THE ORDERS FACT TABLE

1. Let's start with the Orders Fact Table:

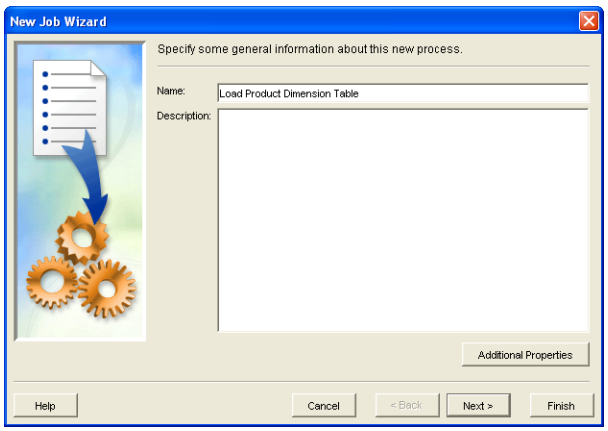
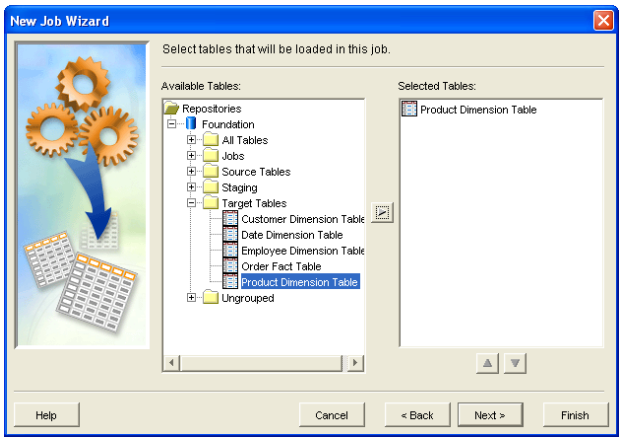
Step	Description	Screen shot
1	New job wizard (from the **Target Designer**)	
2	Select the **Order Fact Table** to be loaded	

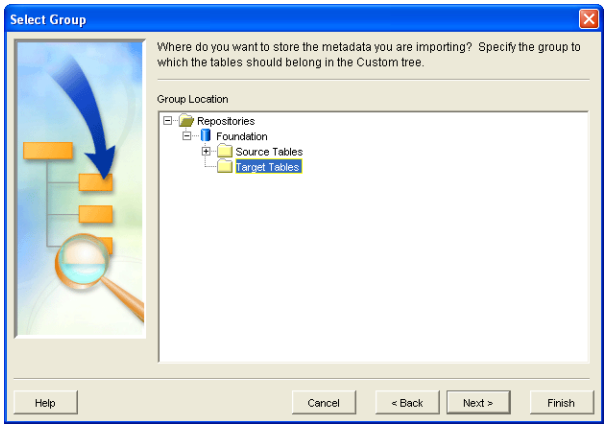
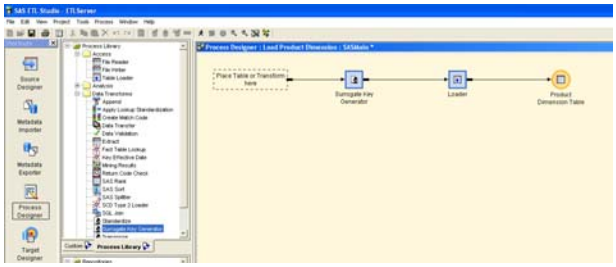
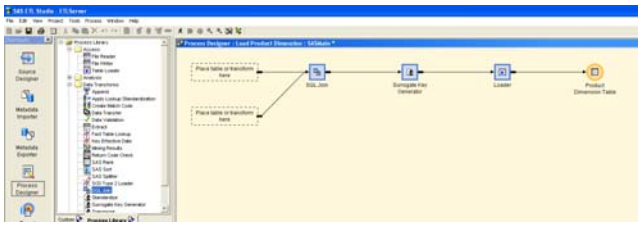
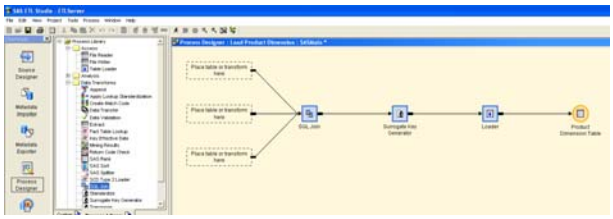
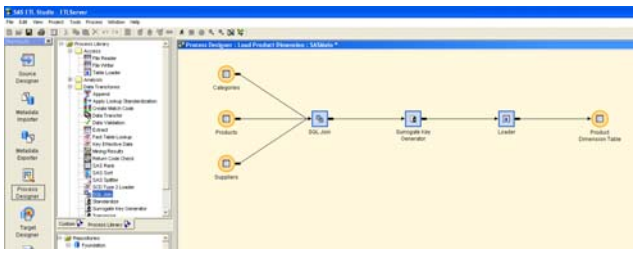
3	Select the location of where we want to store the metadata. Once we confirm our choices, click **finish**.	
4.	The process editor is displayed.	
5.	We want to find the **SQL Join** Transformation (**Process Library → Data Transforms**) and drag that onto the area called **Place table or transform here**.	

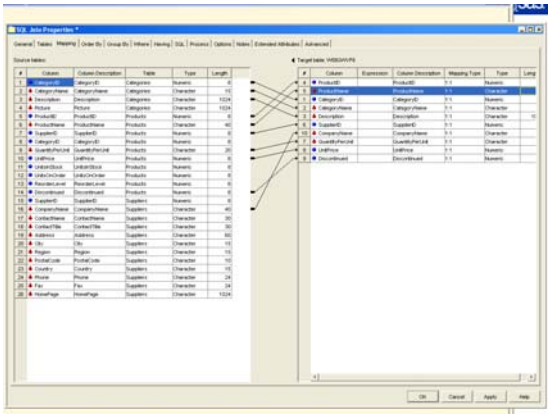
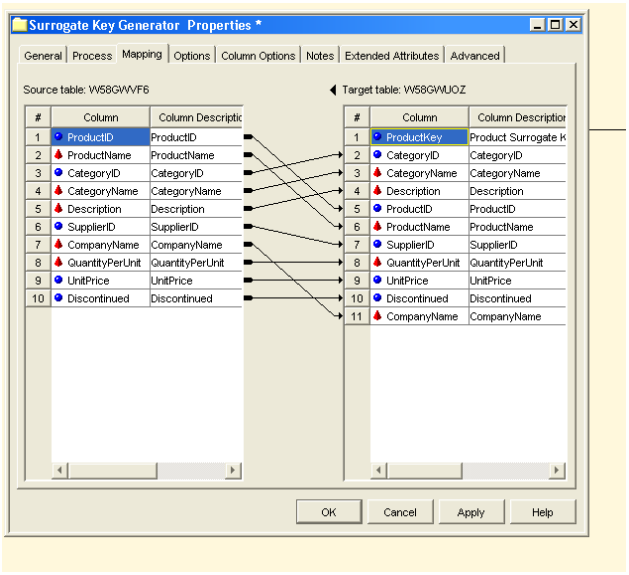
6.	From the **Inventory** Tab, drag the *Orders* table and the *Orders Detail* table (Found in the Northwinds library) onto the two open rectangles.	
7.	Map the *ShipperKey* that we renamed above to the *ShipVia* column found in the *Orders* table. (We do this by right clicking on the loader icon and choosing **Properties → Mapping**. The click on *ShipVia* and drag across to *ShipperKey*. Click ok.	

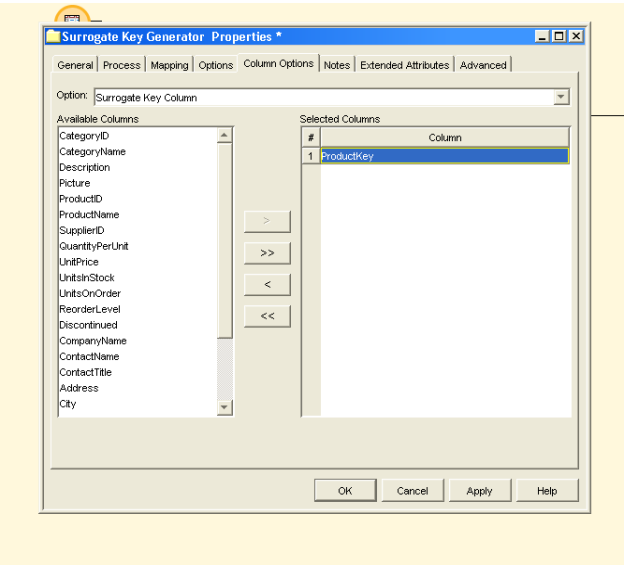
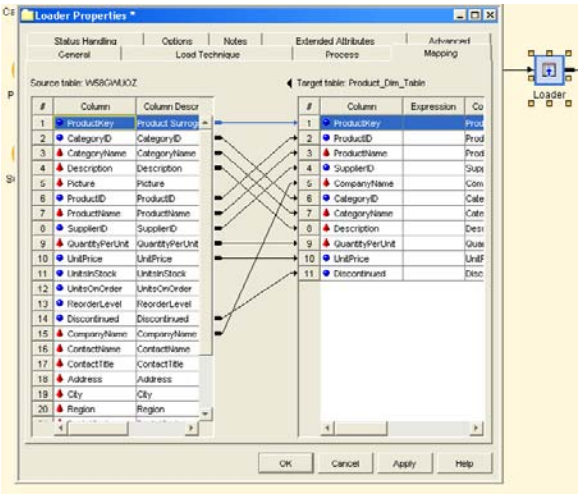
8.	The last step is to test our job to see if it runs correctly. Right click on the process we just created and select **Submit Job**. If all goes well, we can view the log and open the dataset.	

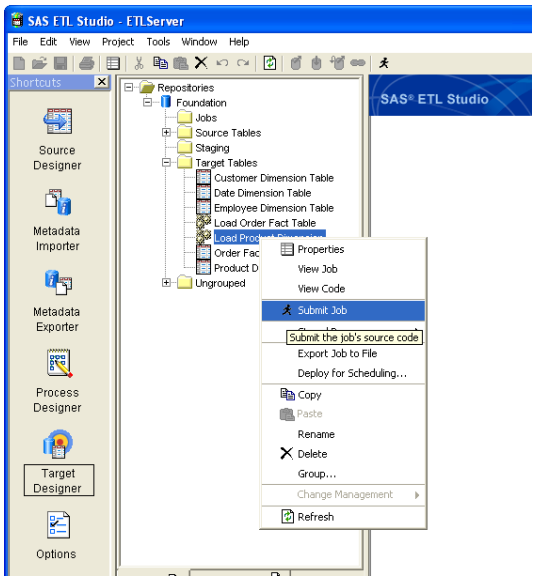
TASK 3 – STEP B: CREATE THE PRODUCT DIMENSION TABLE:

Step	Description	Screen shot
1	**New job wizard**	
2	Select the tables to be loaded	

3	Select the location of where we want to store the metadata. Once we confirm our choices, click finish.	
4.	The default Process is displayed. We need to add two custom transforms to our process: • **Surrogate Key Generator** • **SQL Loader**	Add the Surrogate Key Generator:  Add the SQL Join Transformation:  Add a third input table:  Drag and drop the tables onto the transformation: (Products, Suppliers, Categories): 

5.	Once the process has both transformations, modify the mappings found in the SQL Join to only bring in the columns we want.	Right click on the **SQL Join** icon and select the **Mappings** tab.  Remove all columns on the target table except the ones above (and in the same order).
6.	For the surrogate key generator, we need to customize a few options.	Add *ProductKey* to the target table. To do this, select **Mapping** tab, right click somewhere in the target table and select **Add Column**. The properties for this new column are: • **Column**: *ProductKey* • **Column Description**: *Product Surrogate Key* • **Mapping type**: *none* • **Type**: *Numeric* • **Length**: *8* And remove the non-mapped columns (EXCEPT ProductKey)  Next, select the **Column Options Tab**: • Select the **Option** pull down menu and change the option to *Surrogate Key Column*.

		• Select *ProductKey* and move the field to the right hand column • Select the **Option** pull down menu and change the option to *Business Key Column*. • Select *ProductID* and move the field to the right hand column  Finish by clicking **OK**.
7.	From the **Process Editor**, double click the **Loader** icon. Add the mapping from the **Surrogate Key Generator** to the **Loader** step.	Right click on the **loader** icon. Select **mapping** and map the *ProductKey* in both tables.  Finish by clicking **OK**.

8.	The last step is to test our job to see if it runs correctly. Right click on the process we just created and select **Submit Job**. If all goes well, we can view the log and open the dataset.	

Exercise Summary

In the steps above, we just loaded two tables – a fact table and a dimension table. We hope that you continue your education by trying to load the rest of the dimension table at home. You should also play around with Data Integration Studio and learn all you can about its capabilities. 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 DIS, 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 a solution with DIS.

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
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.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002764619.htm http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002982401.htm
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	Once the data warehouse is loaded, often aggregate tables will be constructed. To learn more about building cubes with SAS, refer to these sources.	http://support.sas.com/onlinedoc/913/getDoc/en/etlug.hlp/a002764598.htm
Surrogate key	In data warehousing, it is beneficial to have a key for your fact tables that are not the same as your business	http://www.dmreview.com/article_sub.cfm?articleId=6136

Concept	Description / Purpose	References
generator	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.	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
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

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.

Inmon, W. H., Claudia Imhoff and Ryan Sousa. The Corporate Information Factory. Wiley. New York. 2nd edition. 2002.

Inmon, W.H., Building the Data Warehouse. QED/Wiley, 1991

Imhoff, Claudia. 2001. "*Intelligent Solutions: Oper Marts – An Evolution in the Operational Data Store.*" DM Review. Brookfield. 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, Gregory S. "*Implementing a Dimensional Data Warehouse with the SAS System*." Invited Paper presented at the SAS Users Group International Conference. San Diego, CA. March, 1999.

Acknowledgements

I am fortunate to work with a number of talented individuals who live and breathe SAS and software engineering. To these kind folks who let me represent them, I thank you. Specifically, I would like to thank Richard Phillips, Crystal Vierhout, Danny Grasse and Jeff Wright for their "thot-ful" and insightful comments on this manuscript.

Biography

Greg Nelson, President and CEO

Greg has just celebrated his 20th year in the SAS eco-system. Starting out as a Social Psychology student doing statistical analysis then quickly moving into applications development. Greg is the President and CEO of ThotWave Technologies where he supports an entire organization focused on helping customers leverage their investment in SAS. 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 recognizes the difference between simply accessing data and making data work for business, and works at the juncture of business and technology to help companies improve their operational and strategic performance. 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