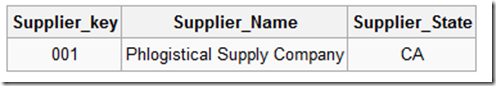
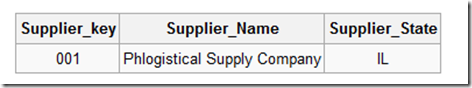
The **Type 1** methodology overwrites old data with new data, and therefore does not track historical data at all.

Another example would be of a database table that keeps supplier information.

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/7450.clip_5F00_image0024_5F00_7D53A4CF.png)

Now imagine that this supplier moves their headquarters to Illinois. The updated table would simply overwrite this record:

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/4341.clip_5F00_image0034_5F00_5D751B47.png)

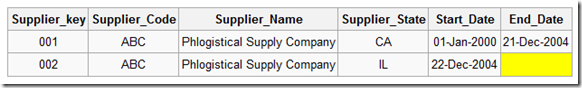
The obvious disadvantage to this method of managing SCDs is that there is no historical record kept in the data warehouse. But an advantage to this is that these are very easy to maintain.

The **Type 2** method tracks historical data by creating multiple records in the dimensional tables with separate keys. With Type 2, we have unlimited history preservation as a new record is inserted each time a change is made.

In the same example, if the supplier moves to Illinois, the table would look like this:

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/5822.clip_5F00_image0044_5F00_4183E64F.png)

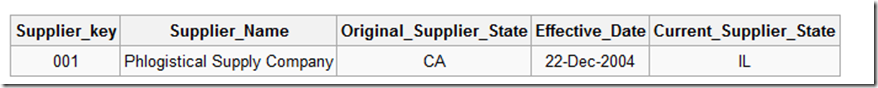
Another popular method for tuple versioning is to add effective date columns.

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/1207.clip_5F00_image0054_5F00_3D49E8BD.png)

Null **End\_Date** signifies current tuple version. In some cases, a standardized surrogate high date (e.g. 9999-12-31) may be used as an end date, so that the field can be included in an index.

Transactions that reference this Surrogate Key (Supplier\_Key) are then permanently bound to these time slices defined by each row in the slowly changing dimension table. If there are retrospective changes made to the contents of the dimension, or if a new set of attributes are added to the dimension (for example a Sales Rep column) which have different effective dates to those already defined, then this can result in the existing transactions needing to be updated to reflect the new situation. This can be an expensive database operation, so Type 2 SCD are not a good choice if the dimensional model is subject to change.

The **Type 3** method tracks changes using separate columns. Whereas Type 2 had unlimited history preservation, Type 3 has limited history preservation, as it's limited to the number of columns we designate for storing historical data. Where the original table structure in Type 1 and Type 2 was very similar, Type 3 will add additional columns to the tables:

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/7506.clip_5F00_image0064_5F00_2DE660E1.png)

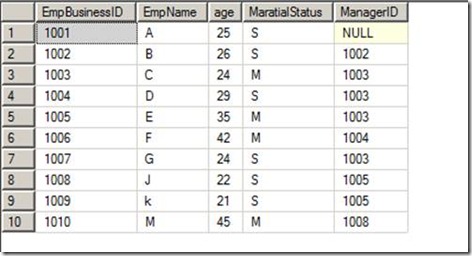
Note that this record cannot track all historical changes, such as when a supplier moves twice. Although, one possible version of this type is to create the field Previous\_Supplier\_State instead of Original\_Supplier\_State which will then track all historical changes.

**Note: Type 3, keeps separate** **columns for both the old and new attribute values—sometimes called “alternate** **realities.” In our experience, Type 3 is less common because it involves** **changing the physical tables and is not very scalable.**

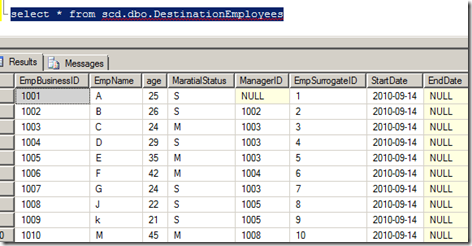
**SSIS has built in support for SCD (Sample Database is Attached)**

This is how Slowly Changing Dimension Wizard - Data Flow Looks like

|  |  |
| --- | --- |
|  |  |
| [clip_image008[4]](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/4278.clip_5F00_image0084_5F00_67C57E00.jpg) | Source Table:  Select \* from scd.dbo.SourceEmployees  Destination Table:  DestinationEmployee |

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/7024.clip_5F00_image0104_5F00_23FE1D1E.jpg)

Execute Package for the first time will give this, which is nothing but a copy of Source Table

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/5488.clip_5F00_image0134_5F00_390BD72C.png)

Now let us fire following statement.

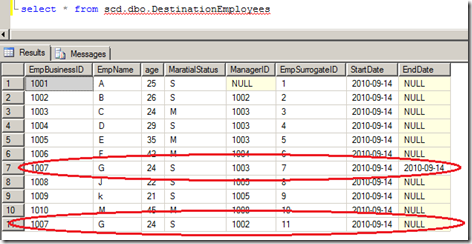
update scd.dbo.SourceEmployees

set managerid=1002

where empname='G'

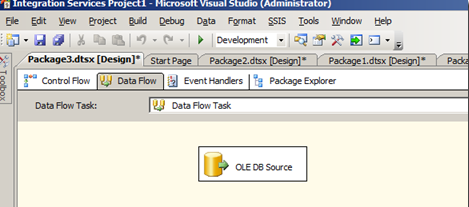
Execute Package and now you will find the difference:

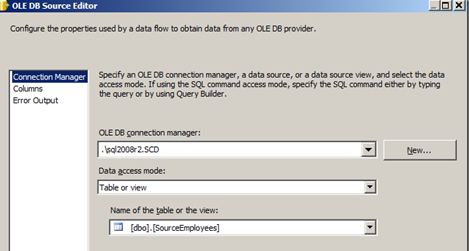
We get one more record for EmpBusinessID 1007 with EmpSurrogateID as 11 and EmpDate as Null and for old record EndDate inserted which signifies that this record is valid till 14th Sep 2010

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/1581.clip_5F00_image0144_5F00_0F67E62C.png)

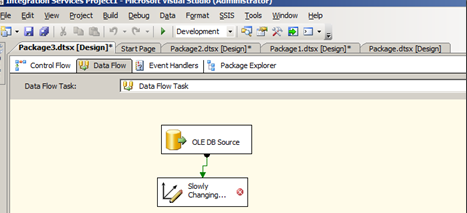
Step of Implementing SCD from SSIS

1) Got to Data Tab and drop one OLE DB Source & Configure it for database ->Source Table

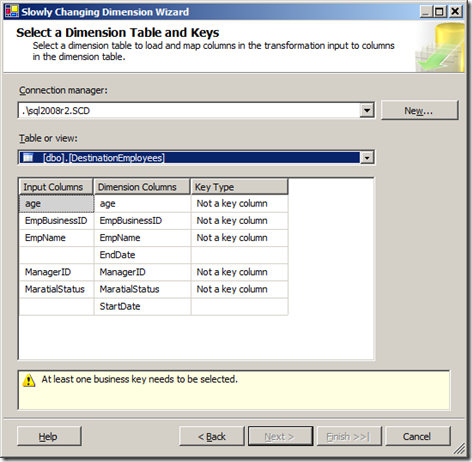




2) From Toolbox drag & drop in Data Flow, connect OLE DB Source created in step 2 to SCD



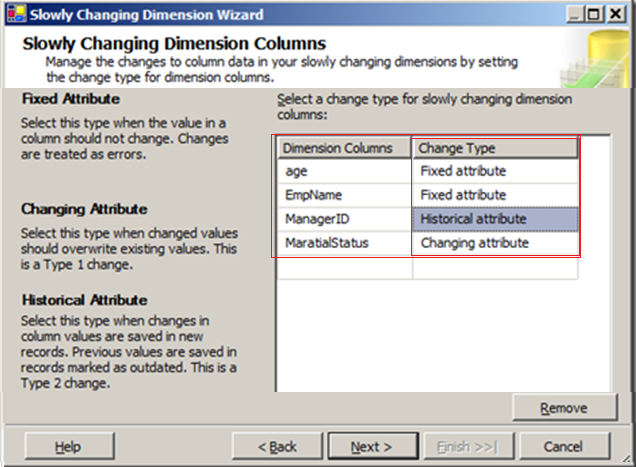
3) Double Click on SCD will initiate SCD Wizard. Select a **Dimension Table and Keys**, here you need to specify the destination table.  **DestinationEmployees**

[](http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Blogs-Components-WeblogFiles/00-00-01-14-37-metablogapi/7220.clip_5F00_image0184_5F00_2652C976.png)

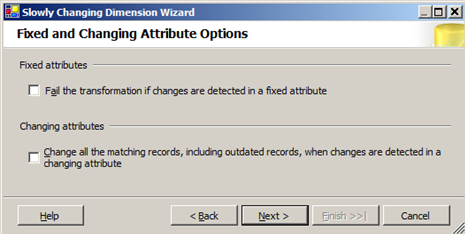
4) Specify **EmpBusinessID** as **Business Key** & Click on next tab, this will allow you to specify the Change Type.

There are three kinds of Attributes:

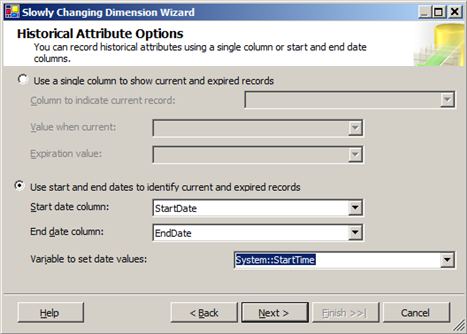
* Fixed Attribute - For which value is not changing
* Changing Attribute - For which value will be over-written (**Type 1**)
* Historical Attribute- For which value is saved as a new record. Previous values are saved in the records marked as outdated (**Type 2**)



5) You can check or uncheck as per your requirement, I am un-checking these two



6) Specify how you want record changed value, using single column or start / end date



7) While loading data in Fact tables we usually see a scenario where the fact data is available but there is no corresponding business key in the related dimension.

In this case we choose multiple options to resolve the issue.

Ignore that fact

Insert the associated business key in dimension table and return the newly generated surrogate key from dimension table. And now store the data in Fact table with the surrogate key.

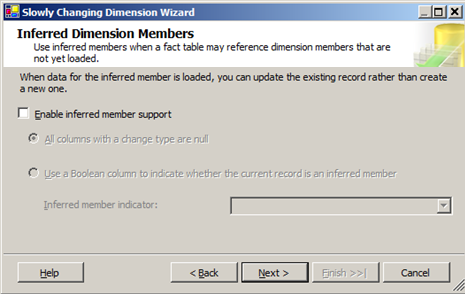
The second approach relates to a term called “Inferred members”. All the other attributes of that dimension will also be updated in next run of dimension load (usually nightly load).

This is al-together a lengthy topic for discussion and outside the scope of this session, for further information refers below mentioned links:-

[SSIS approach to handle Inferred Members](http://mohitnayyar.blogspot.com/2008/05/ssis-approach-to-handle-inferred.html)

[Inferred Members Implementation Best Practices](http://www.bp-msbi.com/2009/08/inferred-members-implementation-best/)

For this demo, I am not checking Enable Inferred member support



Click Next and Finish