Replacement for Return Table in BI 7

Applies to:

Applies to SAP BI 7. For more information, visit the Business Intelligence homepage.

Summary

BW 3.x had an option of Return Table in the update rules. This option is no longer available in BI 7. In this paper I try to discuss alternative options which can be used instead of Return Table.

Author: Kalyan Reddy K

Company: Accenture Service Pvt Ltd

Created on: 24 February 2010

Author Bio

Authore works for Accenture Services Pvt Ltd and has an experience of over 4 years in the SAP BI domain.

Table of Contents

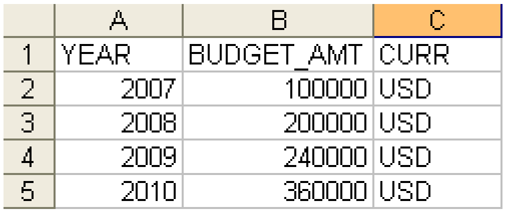
Introduction

Return Table was an option available in BW 3.x update rules, which gave us an option to split the data from one single record in source data into multiple records in the target, based on the specified rules. In BI 7, the option of return table is not available. In this paper, we will try explore the options we can use in order to achieve the same functionality as the Return Table.

Scenario

Imagine a situation where we get Budget Amount on a yearly basis. We want to split the annual Budget Amount equally for every month. Basically this would mean that one record in source would need to transform into 12 records in the target.

Below is the sample data I have created, we will use the same data through out this paper for the demonstration purpose.



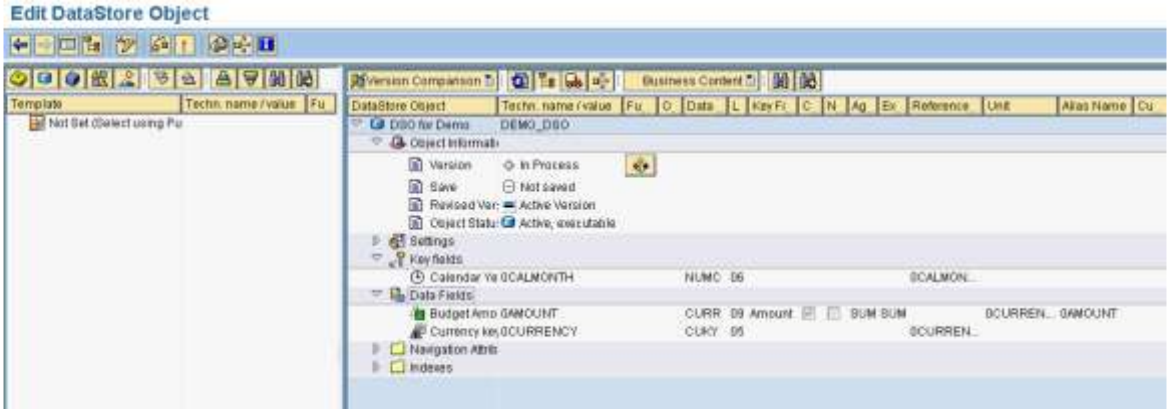
	A	B	C
1	YEAR	BUDGET_AMT	CURR
2	2007	100000	USD
3	2008	200000	USD
4	2009	240000	USD
5	2010	360000	USD

First approach

In the first approach we will use the new option provided by SAP called the Rule Group.

A Rule Group as the name suggests is a group of transformation rules. Every transformation has a default rule groups called as “Standard Group”. The Standard Rule group contains all the default rules for that transformation. In addition to the Standard Rule group, we have an option to have additional Rule Groups. The additional Rule Groups can be deleted but the Standard Rule Group cannot be deleted.

For demonstration purpose, I have created a Standard DSO, with below fields added to it. The DSO has only Calendar Month as the key field, since for our scenario we want to split the data which is by year equally for each month of the year.

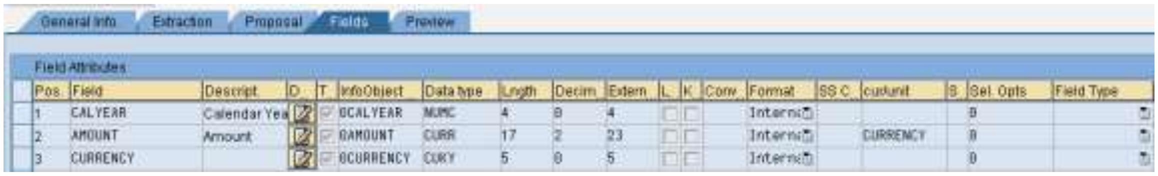


Edit DataStore Object

Template	Techn. name / value	Fu
Not Set (Select using Fu		

DataStore Object	Techn. name / value	Fu	O	Data	L	Key Fi	C	N	Ag	Ex	Reference	Unit	Alias Name	Cu
DSO for Demo	DEMO_DSO													
Object Informati														
Version	In Process													
Save	Not saved													
Revised Ver:	Active Version													
Object Statu:	Active, executable													
Settings														
Key fields														
Calendar Ye	0CALMONTH			NUMC	06						0CALMON...			
Data Fields														
Budget Amo	0AMOUNT			CURR	09	Amount	☑	□	SUM	SUM		0CURREN...	0AMOUNT	
Currency key	0CURRENCY			CUKY	05						0CURREN...			
Navigation Attrib														
Indexes														

A flat file Data Source is created in order to load the sample data into the DSO. The Data Source fields are as below.



Field Attributes

Pos	Field	Descript	D	T	InfoObject	Data type	Lngth	Decim	Extern	L	K	Conv	Format	SS C	cu/unit	S	Sel. Opts	Field Type
1	CALYEAR	Calendar Yea		☑	0CALYEAR	NUMC	4	0	4	□	□		Internal				0	
2	AMOUNT	Amount		☑	0AMOUNT	CURR	17	2	23	□	□		Internal		CURRENCY		0	
3	CURRENCY			☑	0CURRENCY	CUKY	5	0	5	□	□		Internal				0	

Using the Data Source above, create the transformations to the DSO we created. Since we want to have the data updated to the DSO at a monthly level, we write a routine for Calendar Month and Budget Amount in the Standard Rule group.

Since our requirement is to split the Budget Amount equally into 12 months of an year, we write the below routine for the Calendar Month to get value for January in the Standard Rule group.

```
 * Code for getting Calendar Month
     concatenate SOURCE_FIELDS-calyear '01' into RESULT.
```

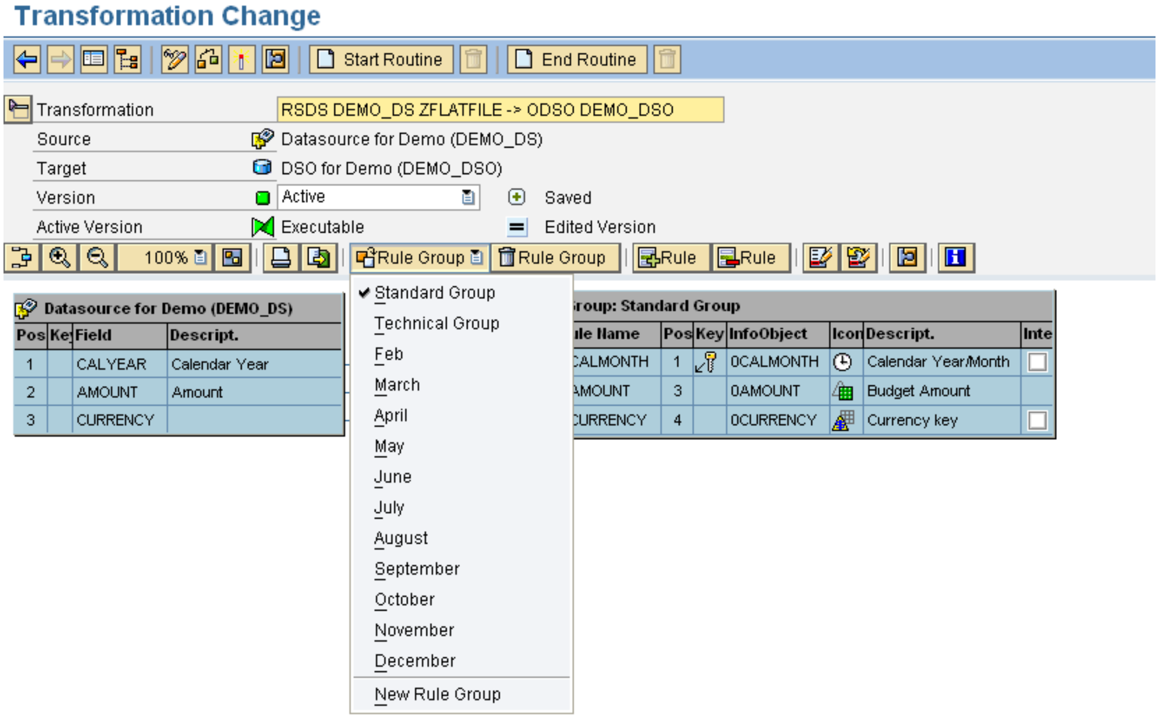
Below code is written for the Budget Key Figure, in order to split the value equally for every month.

```
 * In Order to distribute the amount equally for each month of a year
     RESULT = source_fields-amount / 12.
```

Once the above routines have been written for the Calendar Month and Budget Amount, we create a new Rule Group one for every month from February till December. In each Rule Group, the routine for Budget Amount Key figure remains the same, only the routine for the Calendar Month would change. Below I am showing the routine I have added for Calendar Month in the Rule Group for February

```
   concatenate SOURCE_FIELDS-calyear '02' into RESULT.
```

Similarly as above, we need to add a routine for each month till December. Once all the rule groups are created, we should be able to see all of them as below



Transformation Change

Transformation	RSDS DEMO_DS ZFLATFILE -> ODSO DEMO_DSO
Source	Datasource for Demo (DEMO_DS)
Target	DSO for Demo (DEMO_DSO)
Version	Active — Saved
Active Version	Executable — Edited Version

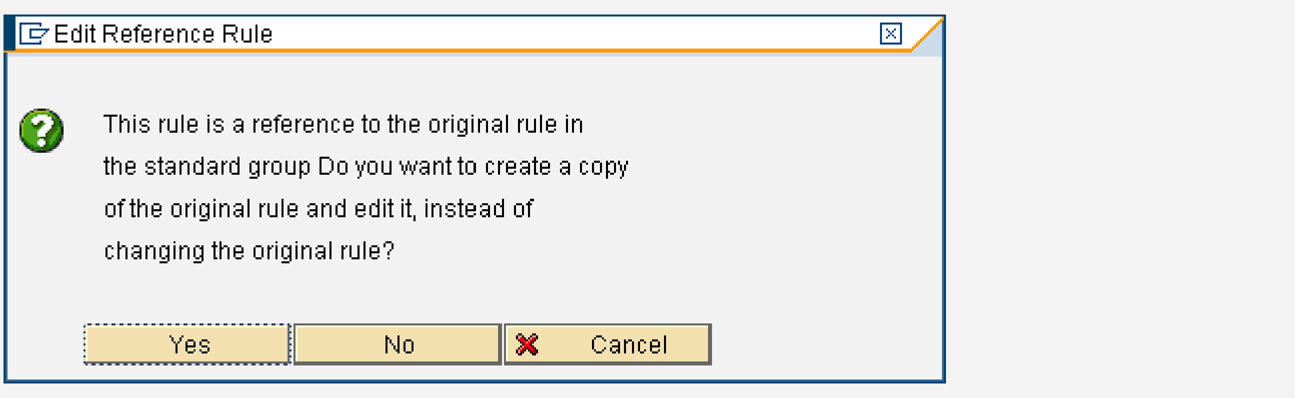
Datasource for Demo (DEMO_DS)

Pos	Ke	Field	Descript.
1		CALYEAR	Calendar Year
2		AMOUNT	Amount
3		CURRENCY	

roup: Standard Group

ule Name	Pos	Key	InfoObject	Icon	Descript.	Inte
CALMONTH	1		0CALMONTH		Calendar Year/Month	□
AMOUNT	3		0AMOUNT		Budget Amount	
CURRENCY	4		0CURRENCY		Currency key	□

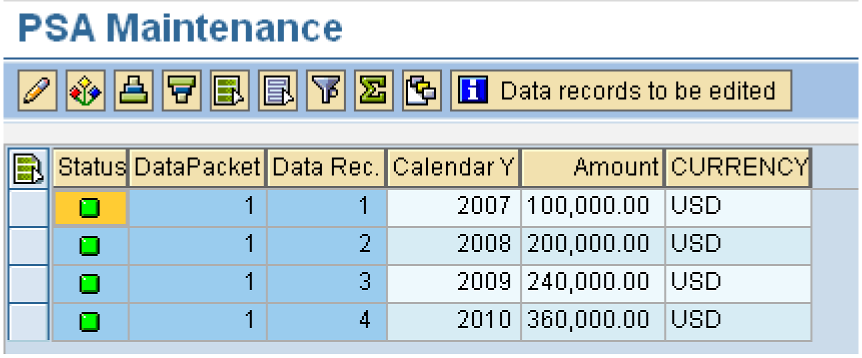
Note: When creating a new rule group, you will see a message pop up as below, Click on Yes, so the the Standard Rule group is copied and you can make changes to it. If you select No, any changes you make in the new rule group would reflect in the Standard rule group as well.



Edit Reference Rule

This rule is a reference to the original rule in the standard group Do you want to create a copy of the original rule and edit it, instead of changing the original rule?

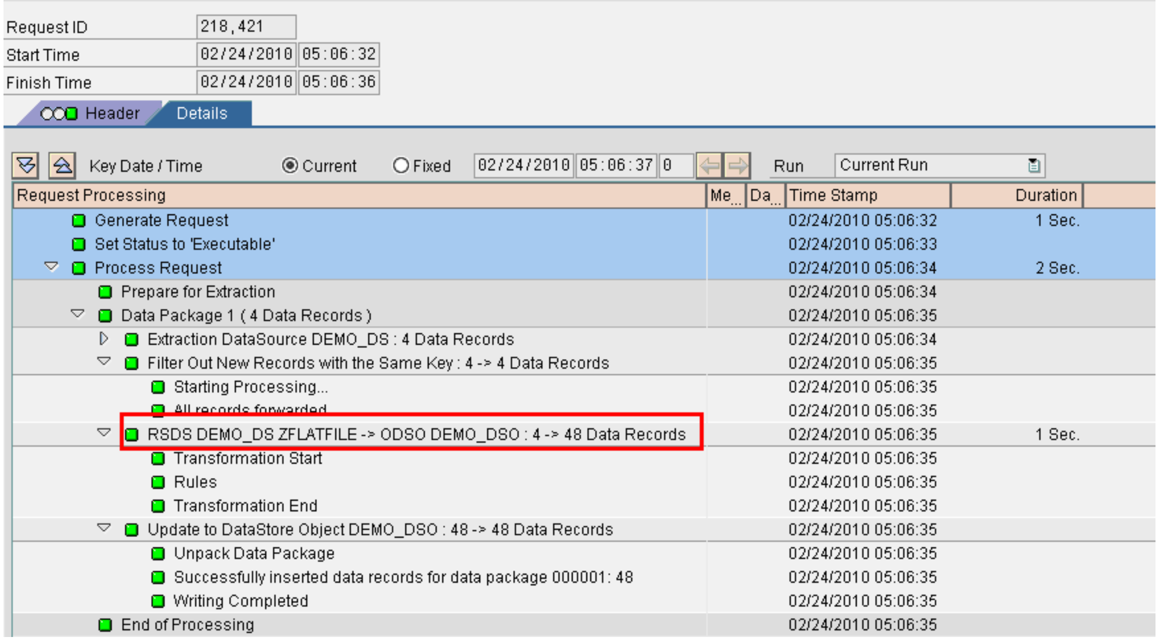
Once the Rule Groups are all created, we will start loading the data. Below is the screenshot showing source data loaded to the PSA.



PSA Maintenance

Status	DataPacket	Data Rec.	Calendar Y	Amount	CURRENCY
🟩	1	1	2007	100,000.00	USD
🟩	1	2	2008	200,000.00	USD
🟩	1	3	2009	240,000.00	USD
🟩	1	4	2010	360,000.00	USD

Below is the screenshot showing the log for the load which was done to the DSO. As we can see in the log, 4 data records were transferred and 48 records were added to the DSO. Which is what we expected to see, as the four data records we had in the source contained Budget Amount for 4 years. In the target DSO, we wanted to store data at monthly level. Therefore, 4 years * 12 months/year = 48 months, which is the number of records that are added to the DSO.

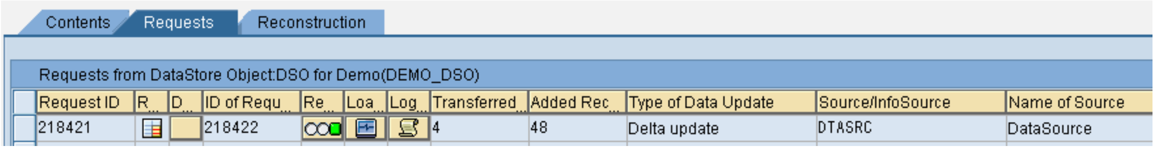


Request ID	218,421	
Start Time	02/24/2010	05:06:32
Finish Time	02/24/2010	05:06:36

Key Date / Time ● Current ○ Fixed 02/24/2010 05:06:37 0 Run Current Run

```
Request Processing                                                              Me… Da… Time Stamp           Duration
  Generate Request                                                                      02/24/2010 05:06:32  1 Sec.
  Set Status to 'Executable'                                                            02/24/2010 05:06:33
▽ Process Request                                                                       02/24/2010 05:06:34  2 Sec.
    Prepare for Extraction                                                              02/24/2010 05:06:34
  ▽ Data Package 1 ( 4 Data Records )                                                   02/24/2010 05:06:35
    ▷ Extraction DataSource DEMO_DS : 4 Data Records                                    02/24/2010 05:06:34
    ▽ Filter Out New Records with the Same Key : 4 -> 4 Data Records                    02/24/2010 05:06:35
        Starting Processing...                                                          02/24/2010 05:06:35
        All records forwarded                                                           02/24/2010 05:06:35
    ▽ RSDS DEMO_DS ZFLATFILE -> ODSO DEMO_DSO : 4 -> 48 Data Records                    02/24/2010 05:06:35  1 Sec.
        Transformation Start                                                            02/24/2010 05:06:35
        Rules                                                                           02/24/2010 05:06:35
        Transformation End                                                              02/24/2010 05:06:35
    ▽ Update to DataStore Object DEMO_DSO : 48 -> 48 Data Records                       02/24/2010 05:06:35
        Unpack Data Package                                                             02/24/2010 05:06:35
        Successfully inserted data records for data package 000001: 48                  02/24/2010 05:06:35
        Writing Completed                                                               02/24/2010 05:06:35
    End of Processing                                                                   02/24/2010 05:06:35
```

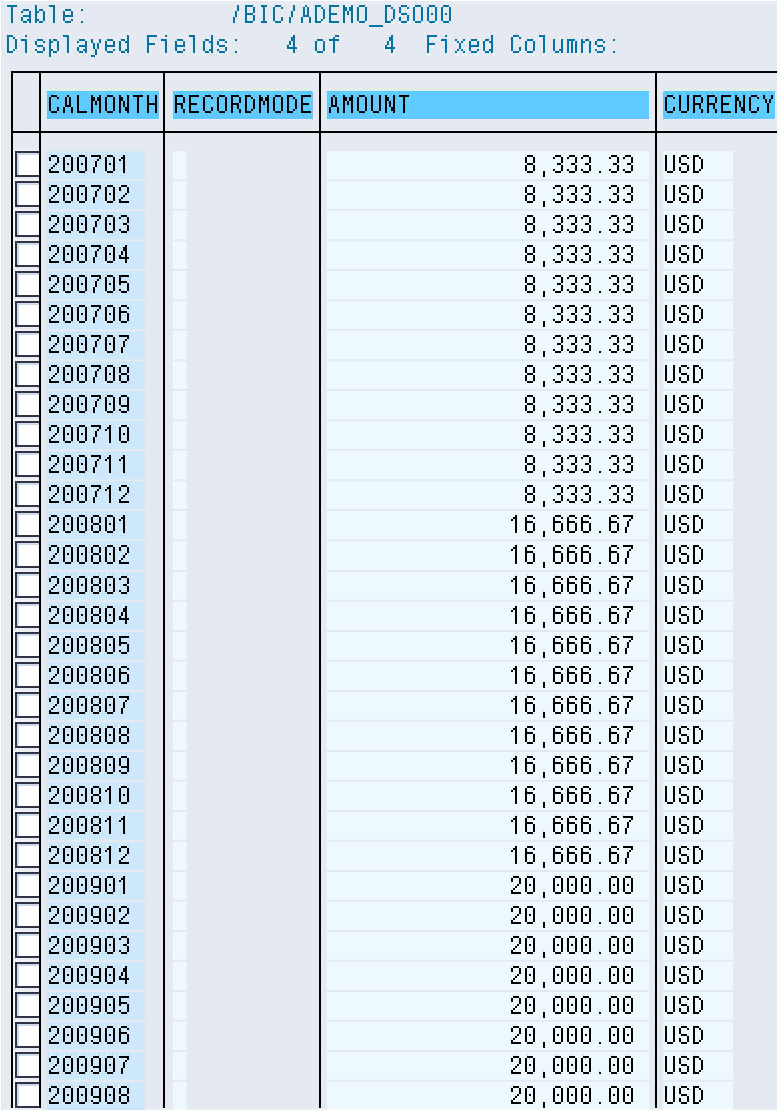
Below is the screenshot showing the request status in the DSO.



Requests from DataStore Object:DSO for Demo(DEMO_DSO)

Request ID	R...	D...	ID of Requ...	Re...	Loa...	Log...	Transferred...	Added Rec...	Type of Data Update	Source/InfoSource	Name of Source
218421			218422	○○●			4	48	Delta update	DTASRC	DataSource

Data Loaded to the DSO is as below.



```
Table:          /BIC/ADEMO_DS000
Displayed Fields:   4 of   4  Fixed Columns:
```

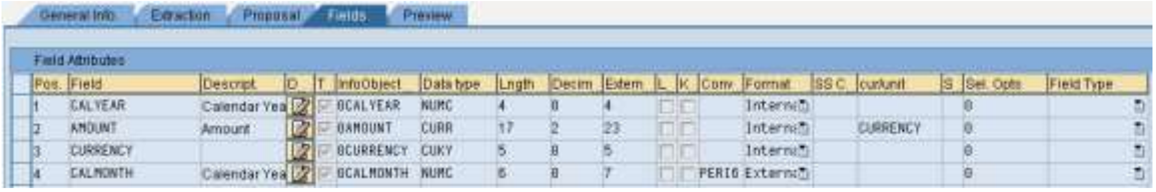
	CALMONTH	RECORDMODE	AMOUNT	CURRENCY
□	200701		8,333.33	USD
□	200702		8,333.33	USD
□	200703		8,333.33	USD
□	200704		8,333.33	USD
□	200705		8,333.33	USD
□	200706		8,333.33	USD
□	200707		8,333.33	USD
□	200708		8,333.33	USD
□	200709		8,333.33	USD
□	200710		8,333.33	USD
□	200711		8,333.33	USD
□	200712		8,333.33	USD
□	200801		16,666.67	USD
□	200802		16,666.67	USD
□	200803		16,666.67	USD
□	200804		16,666.67	USD
□	200805		16,666.67	USD
□	200806		16,666.67	USD
□	200807		16,666.67	USD
□	200808		16,666.67	USD
□	200809		16,666.67	USD
□	200810		16,666.67	USD
□	200811		16,666.67	USD
□	200812		16,666.67	USD
□	200901		20,000.00	USD
□	200902		20,000.00	USD
□	200903		20,000.00	USD
□	200904		20,000.00	USD
□	200905		20,000.00	USD
□	200906		20,000.00	USD
□	200907		20,000.00	USD
□	200908		20,000.00	USD

As we can see in the above screenshot, it can be seen that the data which was at Yearly level is split equally for every month and a new records is added for each month to the DSO.

Second Approach

For second approach we can use ABAP for splitting the data in the source at Yearly level into Monthly level. The ABAP can be used either in Start Routine or End Routine or even in Expert Routine, for our demo, we will use ABAP in Start Routine.

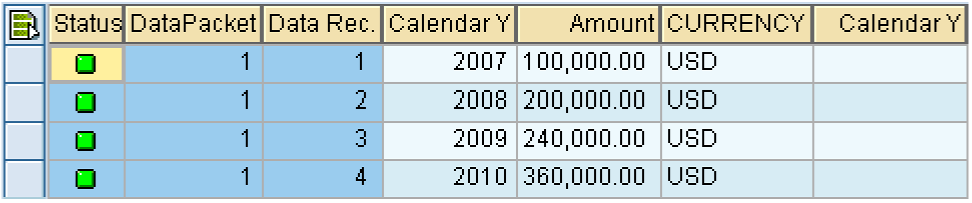
In order to accommodate our requirement, I have made a small change to the data source by adding the Calendar month field to it. Thought this is not needed, and can still be achieved by making some modifications in the ABAP, I have added the Calendar Month field to the Data Source in order to make the demonstration simpler. Below is the screenshot of the enhanced data source



Field Attributes

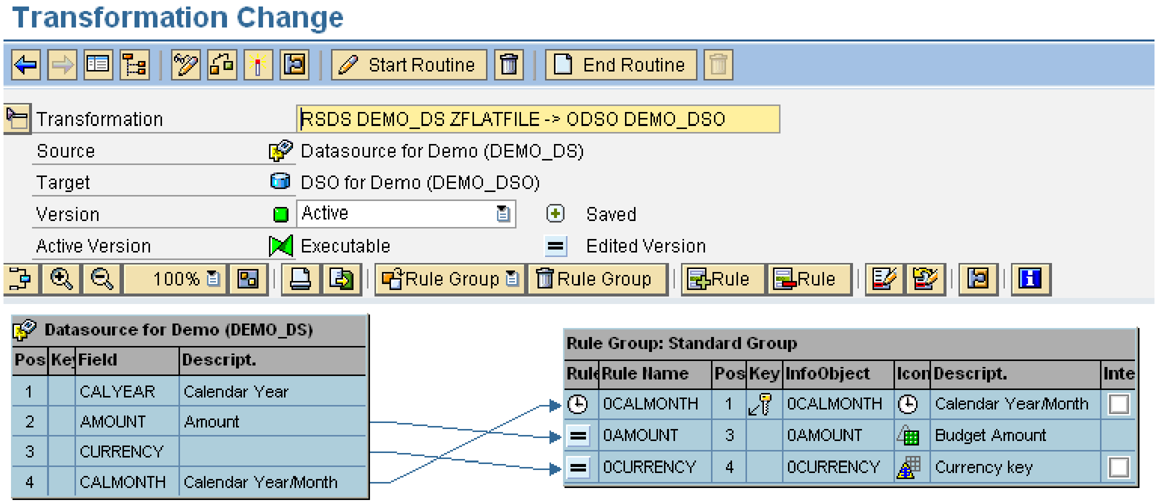
Pos.	Field	Descript.	D	T	InfoObject	Data type	Lngth	Decim	Extern	L	K	Conv.	Format	SS C	cur/unit	S	Sel. Opts	Field Type
1	CALYEAR	Calendar Yea			0CALYEAR	NUMC	4	0	4	□	□		Interna				0	
2	AMOUNT	Amount			0AMOUNT	CURR	17	2	23	□	□		Interna		CURRENCY		0	
3	CURRENCY				0CURRENCY	CUKY	5	0	5	□	□		Interna				0	
4	CALMONTH	Calendar Yea			0CALMONTH	NUMC	6	0	7	□	□	PERI6	Externa				0	

After adding the Calendar Month, the data from the PSA would look like as below, with no data being populated for Calendar Month. Calendar Month is added as a place holder in the data source, which can be used in the Start routine. For the same scenario, if the decision was to use End Routine, then we would require to change the DSO and add an additional field Calendar Year. Since we do not have access to Source Package in the End Routine, we need to have the Year value in order to achieve our goal of splitting the Budget Value into 12 equal month of the year.



Status	DataPacket	Data Rec.	Calendar Y	Amount	CURRENCY	Calendar Y
🟩	1	1	2007	100,000.00	USD	
🟩	1	2	2008	200,000.00	USD	
🟩	1	3	2009	240,000.00	USD	
🟩	1	4	2010	360,000.00	USD	

For this approach, the transformations would be simple one to one mapping as below



Transformation Change

Transformation: RSDS DEMO_DS ZFLATFILE -> ODSO DEMO_DSO
Source: Datasource for Demo (DEMO_DS)
Target: DSO for Demo (DEMO_DSO)
Version: Active — Saved
Active Version: Executable — Edited Version

Datasource for Demo (DEMO_DS)

Pos	Ke	Field	Descript.
1		CALYEAR	Calendar Year
2		AMOUNT	Amount
3		CURRENCY	
4		CALMONTH	Calendar Year/Month

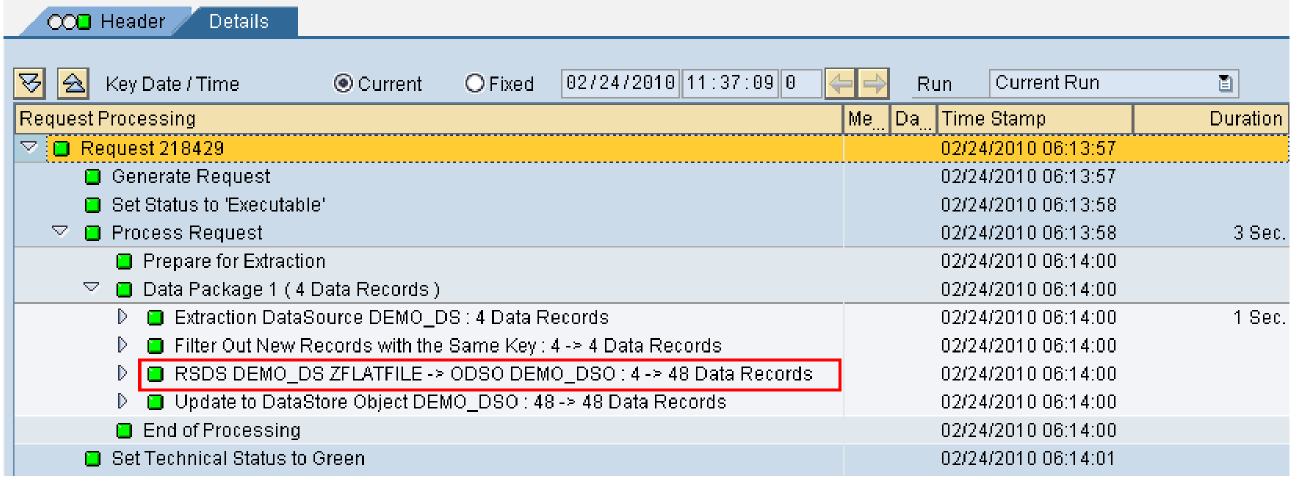
Rule Group: Standard Group

Rul	Rule Name	Pos	Key	InfoObject	Icon	Descript.	Inte
	0CALMONTH	1		0CALMONTH		Calendar Year/Month	□
=	0AMOUNT	3		0AMOUNT		Budget Amount	
=	0CURRENCY	4		0CURRENCY		Currency key	□

We add the following ABAP code in the Start Routine to split the data

```
data: it_sp type standard table of _ty_s_SC_1,
      wa_sp type _ty_s_SC_1.

data: tmp_amt type /bi0/oiamount,
      tmp_indx(2),
      tmp_ctr type i.

loop at SOURCE_PACKAGE assigning <source_fields>.
  clear wa_sp.
  clear tmp_indx.
  move <source_fields> to wa_sp.
  tmp_amt = wa_sp-amount.
  tmp_ctr = 1.
  do 12 times.
    tmp_indx = tmp_ctr.
    if tmp_ctr LE 9.
      concatenate wa_sp-calyear '0' tmp_indx into wa_sp-calmonth.
    else.
      concatenate wa_sp-calyear tmp_indx into wa_sp-calmonth.
    endif.
    wa_sp-amount = tmp_amt / 12.
    append wa_sp to it_sp.
    tmp_ctr = tmp_ctr + 1.
  enddo.
endloop.

refresh SOURCE_PACKAGE.

move it_sp to SOURCE_PACKAGE.
```

The log request of the DTP loaded is as below, which shows that 4 records are transferred and 48 records are added. Same as the First Approach we have taken.



Request Processing	Me...	Da...	Time Stamp	Duration
🟩 Request 218429			02/24/2010 06:13:57	
🟩 Generate Request			02/24/2010 06:13:57	
🟩 Set Status to 'Executable'			02/24/2010 06:13:58	
🟩 Process Request			02/24/2010 06:13:58	3 Sec.
🟩 Prepare for Extraction			02/24/2010 06:14:00	
🟩 Data Package 1 (4 Data Records)			02/24/2010 06:14:00	
🟩 Extraction DataSource DEMO_DS : 4 Data Records			02/24/2010 06:14:00	1 Sec.
🟩 Filter Out New Records with the Same Key : 4 -> 4 Data Records			02/24/2010 06:14:00	
🟩 RSDS DEMO_DS ZFLATFILE -> ODSO DEMO_DSO : 4 -> 48 Data Records			02/24/2010 06:14:00	
🟩 Update to DataStore Object DEMO_DSO : 48 -> 48 Data Records			02/24/2010 06:14:00	
🟩 End of Processing			02/24/2010 06:14:00	
🟩 Set Technical Status to Green			02/24/2010 06:14:01	

Below is the screen shot from the Target DSO, after loading using the second approach.

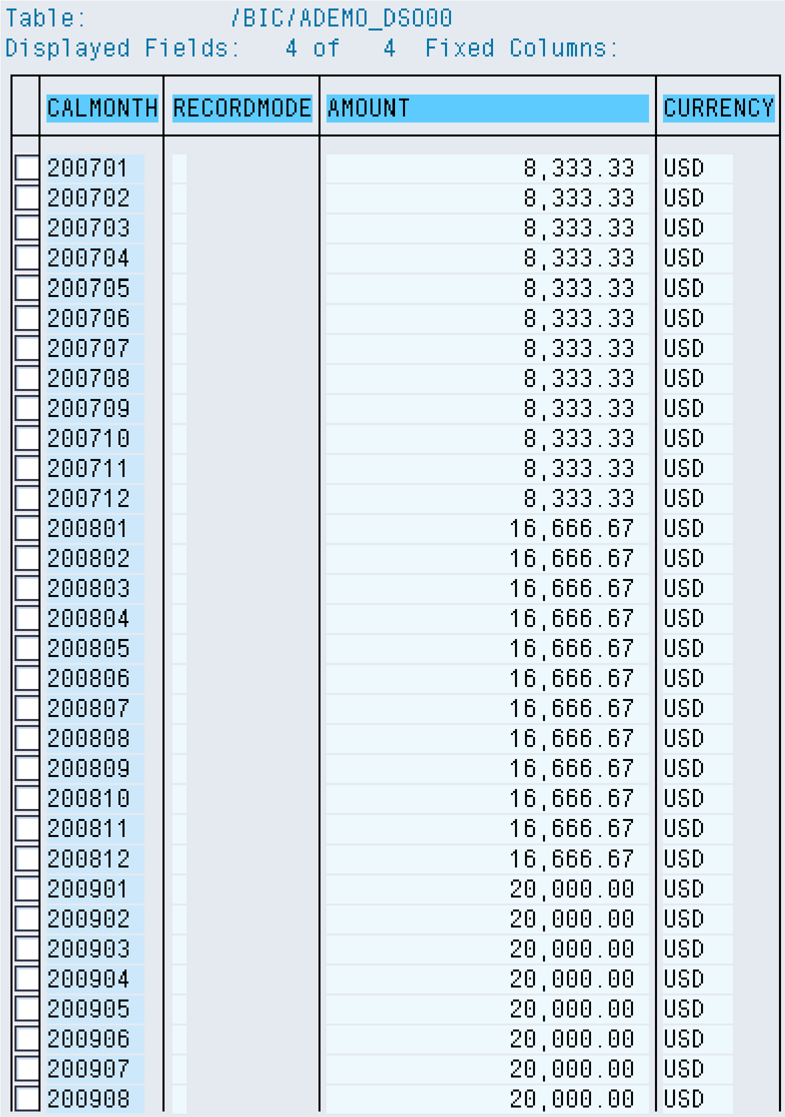


Table: /BIC/ADEMO_DSO00
Displayed Fields: 4 of 4 Fixed Columns:

	CALMONTH	RECORDMODE	AMOUNT	CURRENCY
□	200701		8,333.33	USD
□	200702		8,333.33	USD
□	200703		8,333.33	USD
□	200704		8,333.33	USD
□	200705		8,333.33	USD
□	200706		8,333.33	USD
□	200707		8,333.33	USD
□	200708		8,333.33	USD
□	200709		8,333.33	USD
□	200710		8,333.33	USD
□	200711		8,333.33	USD
□	200712		8,333.33	USD
□	200801		16,666.67	USD
□	200802		16,666.67	USD
□	200803		16,666.67	USD
□	200804		16,666.67	USD
□	200805		16,666.67	USD
□	200806		16,666.67	USD
□	200807		16,666.67	USD
□	200808		16,666.67	USD
□	200809		16,666.67	USD
□	200810		16,666.67	USD
□	200811		16,666.67	USD
□	200812		16,666.67	USD
□	200901		20,000.00	USD
□	200902		20,000.00	USD
□	200903		20,000.00	USD
□	200904		20,000.00	USD
□	200905		20,000.00	USD
□	200906		20,000.00	USD
□	200907		20,000.00	USD
□	200908		20,000.00	USD

Conclusion

Using both the approaches discussed in this paper, it is possible to accomplish the task of splitting data at a higher level in source to a detail level in the target. But the approach we chose would vary from situation to situation. First approach would require us to create several Rule Groups, with ABAP routines for each rule group separately. Where as the second approach would enable us to have a simple one to one mapping in the transformations and have all the logic done at the Start routine.

Related Content

http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/c07161ee-afa6-2c10-06bd-f1c8643558a5

http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/20b50a75-0330-2c10-bf86-a2e4b7ab64ba

For more information, visit the Business Intelligence homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.