Informatica PC Training

Day-6

Agenda:

- Normalizer Transformation
- Transaction Control Transformation
- Reusable Transformation
- Mapplet
- Parameters and Variables
- Demo

Debadatta Mohanty

Housekeeping Tips

- Please mute your phone during the presentation.
- If there is too much noise, participants will be put on auto-mute.
- We shall open up the table for Q&A at the end of the session.
- Please feel free to post your questions over Chat as well.
- This session will be recorded and an email will be sent with links to the recordings after the session.
- At the end of the course, TEX will request you to provide feedback on the training.

Normalizer Transformation

Normalizer Transformations

- The Normalizer transformation receives a row that contains multiple-occurring columns and returns a row for each instance of the multiple-occurring data.
- The transformation processes multiple-occurring columns or multiple-occurring groups of columns in each source row.

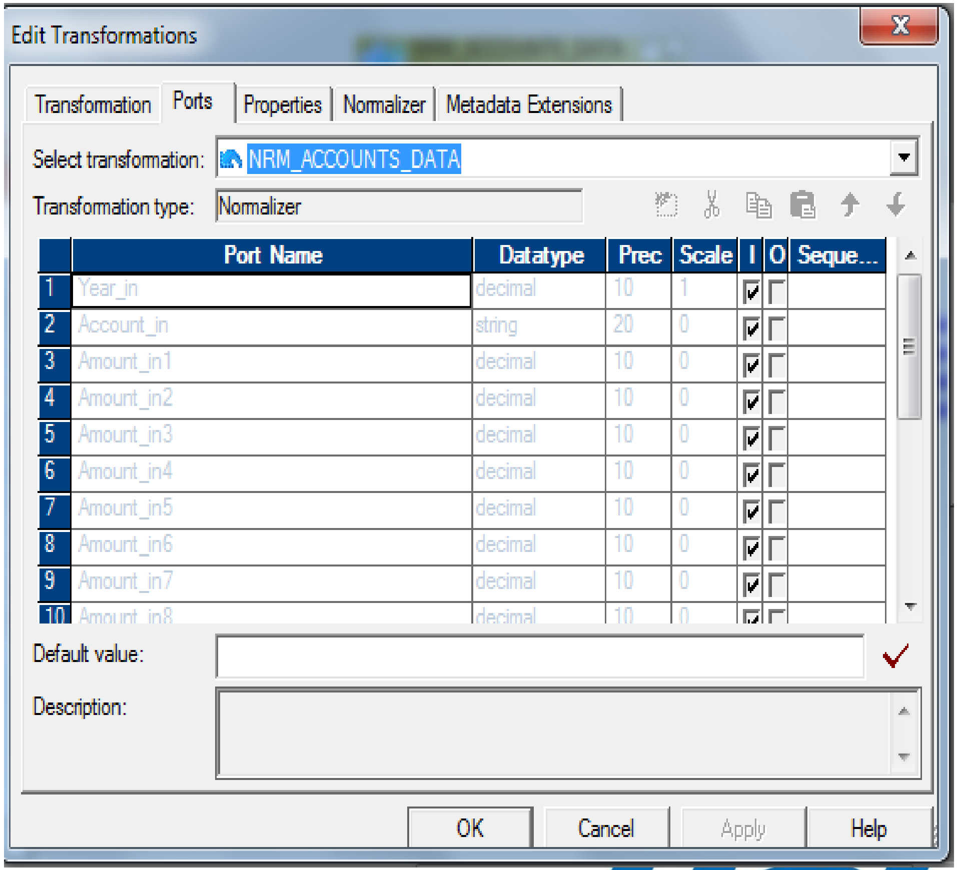
Active Transformation
Connected

Ports

- Input / output or output

Usage

- Required for VSAM Source definitions
- Normalize flat file or relational source definitions
- Generate multiple records from one record



Edit Transformations

Select transformation: NRM_ACCOUNTS_DATA

Transformation type: Normalizer

	Port Name	Datatype	Prec	Scale	I	O	Seque...
1	Year_in	decimal	10	1	☑	□	
2	Account_in	string	20	0	☑	□	
3	Amount_in1	decimal	10	0	☑	□	
4	Amount_in2	decimal	10	0	☑	□	
5	Amount_in3	decimal	10	0	☑	□	
6	Amount_in4	decimal	10	0	☑	□	
7	Amount_in5	decimal	10	0	☑	□	
8	Amount_in6	decimal	10	0	☑	□	
9	Amount_in7	decimal	10	0	☑	□	
10	Amount_in8	decimal	10	0	☑	□	

Default value:

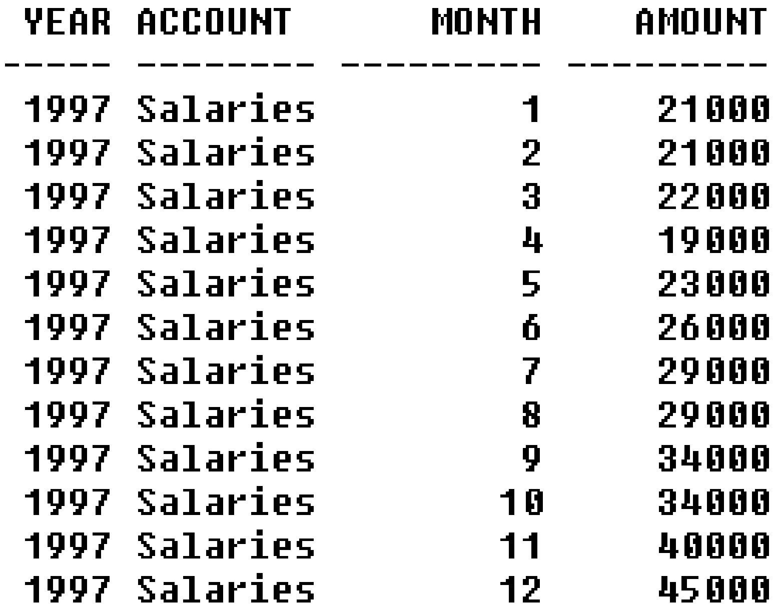
Description:

Normalizer Transformations

Turn one row

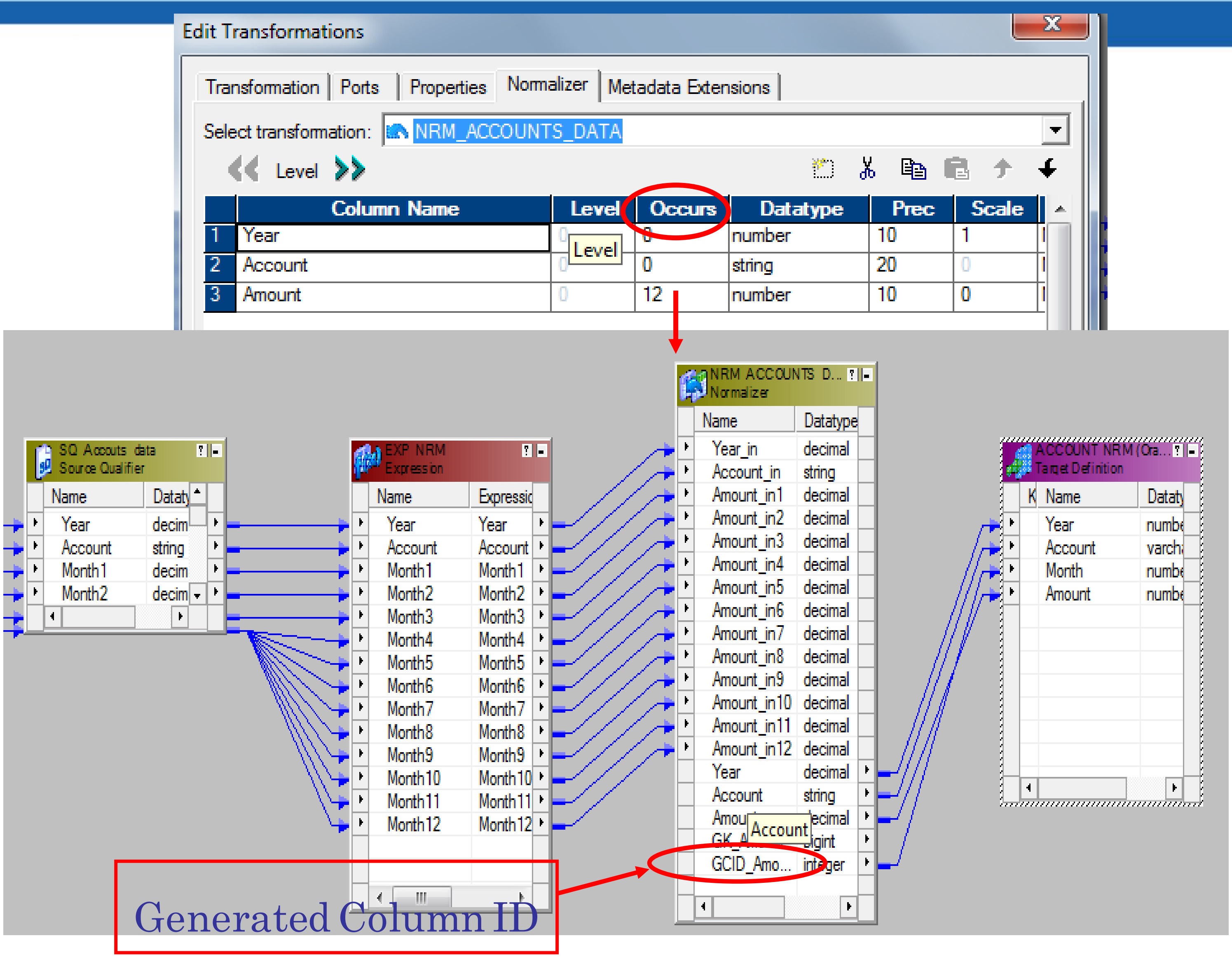
YEAR,ACCOUNT,MONTH1,MONTH2,MONTH3, … MONTH12
1997,Salaries,21000,21000,22000,19000,23000,26000,29000,29000,34000,34000,40000,45000
1997,Benefits,4200,4200,4400,3800,4600,5200,5800,5800,6800,6800,8000,9000
1997,Expenses,10500,4000,5000,6500,3000,7000,9000,4500,7500,8000,8500,8250

Into multiple rows



```
 YEAR ACCOUNT        MONTH     AMOUNT
----- --------- ---------- ----------
 1997 Salaries           1      21000
 1997 Salaries           2      21000
 1997 Salaries           3      22000
 1997 Salaries           4      19000
 1997 Salaries           5      23000
 1997 Salaries           6      26000
 1997 Salaries           7      29000
 1997 Salaries           8      29000
 1997 Salaries           9      34000
 1997 Salaries          10      34000
 1997 Salaries          11      40000
 1997 Salaries          12      45000
```

Normalizer Transformations



Edit Transformations

Transformation | Ports | Properties | Normalizer | Metadata Extensions

Select transformation: NRM_ACCOUNTS_DATA

Level

	Column Name	Level	Occurs	Datatype	Prec	Scale
1	Year	0	0	number	10	1
2	Account	0	0	string	20	0
3	Amount	0	12	number	10	0

SQ_Accouts_data
Source Qualifier

Name	Dataty
Year	decim
Account	string
Month1	decim
Month2	decim

EXP_NRM
Expression

Name	Expressio
Year	Year
Account	Account
Month1	Month1
Month2	Month2
Month3	Month3
Month4	Month4
Month5	Month5
Month6	Month6
Month7	Month7
Month8	Month8
Month9	Month9
Month10	Month10
Month11	Month11
Month12	Month12

NRM_ACCOUNTS_D...
Normalizer

Name	Datatype
Year_in	decimal
Account_in	string
Amount_in1	decimal
Amount_in2	decimal
Amount_in3	decimal
Amount_in4	decimal
Amount_in5	decimal
Amount_in6	decimal
Amount_in7	decimal
Amount_in8	decimal
Amount_in9	decimal
Amount_in10	decimal
Amount_in11	decimal
Amount_in12	decimal
Year	decimal
Account	string
Amou...	decimal
GK_A...	bigint
GCID_Amo...	integer

ACCOUNT_NRM (Ora...
Target Definition

K	Name	Dataty
	Year	numbe
	Account	varch
	Month	numbe
	Amount	numbe

Generated Column ID

Transaction Control Transformation

Transaction Control Transformation

- PowerCenter allows you to control commit and rollback transactions based on a set of rows that pass through a Transaction Control transformation.
- A transaction is the set of rows bound by commit or rollback rows.
- you define transaction control at two levels
  - ✓ Within a mapping
  - ✓ Within a session

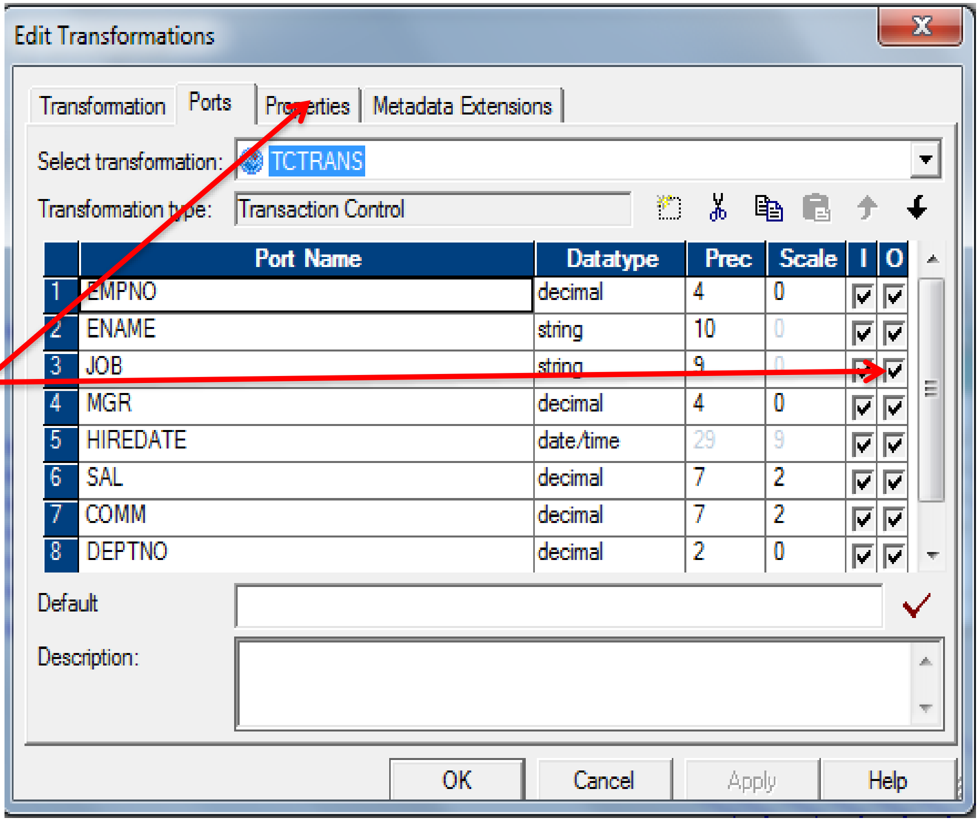
Active Transformation
Connected Mode Only

Ports

- Input and Output

Properties

- TC_CONTINUE_TRANSACTION
- TC_COMMIT_BEFORE
- TC_COMMIT_AFTER
- TC_ROLLBACK_BEFORE
- TC_ROLLBACK_AFTER



Select transformation: TCTRANS

Transformation type: Transaction Control

	Port Name	Datatype	Prec	Scale	I	O
1	EMPNO	decimal	4	0	☑	☑
2	ENAME	string	10	0	☑	☑
3	JOB	string	9	0	☑	☑
4	MGR	decimal	4	0	☑	☑
5	HIREDATE	date/time	29	9	☑	☑
6	SAL	decimal	7	2	☑	☑
7	COMM	decimal	7	2	☑	☑
8	DEPTNO	decimal	2	0	☑	☑

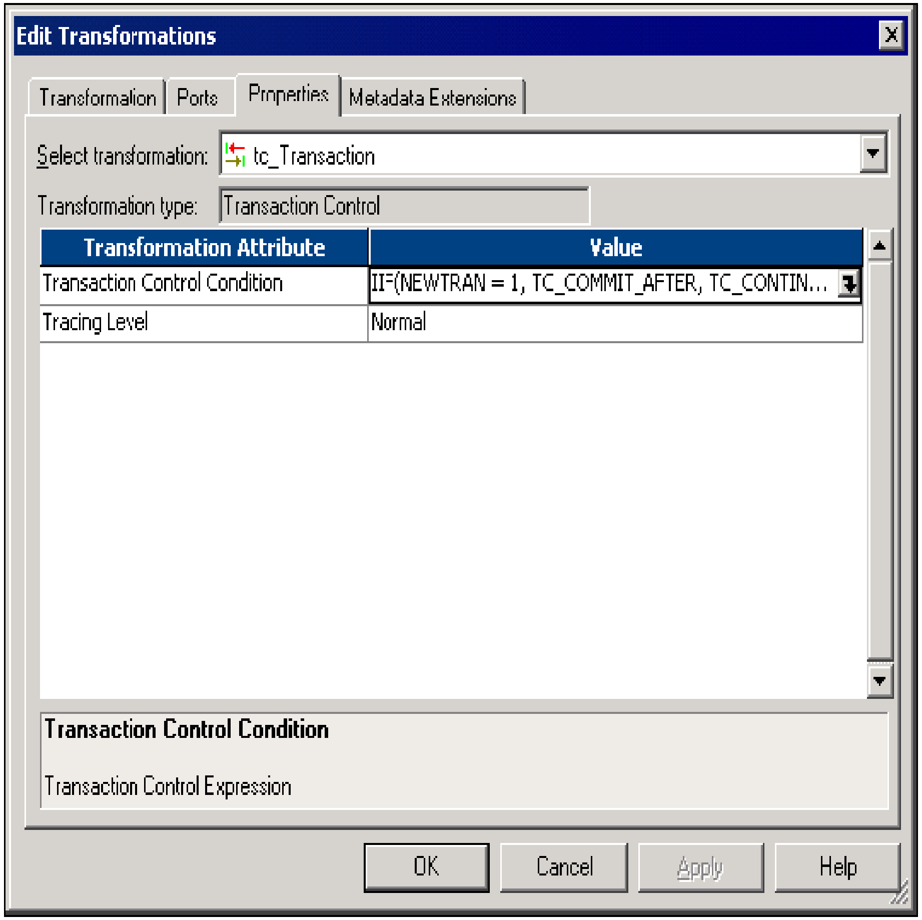
Default

Description:

Transaction Control Transformation

Properties

- You can define the transaction control expression, which flags transactions for commit, rollback, or no action.
- Enter the transaction control expression in the Transaction Control Condition field. The transaction control expression uses the IIF function to test each row against the condition.
- Use the following syntax for the expression: IIF (condition, value1, value2)
- The expression contains values that represent actions the PowerCenter Server performs based on the return value of the condition. The PowerCenter Server evaluates the condition on a row-by-row basis.
- The return value determines whether the PowerCenter Server commits, rolls back, or makes no transaction changes to the row. When the PowerCenter Server issues a commit or rollback based on the return value of the expression, it begins a new transaction.



Select transformation: tc_Transaction

Transformation type: Transaction Control

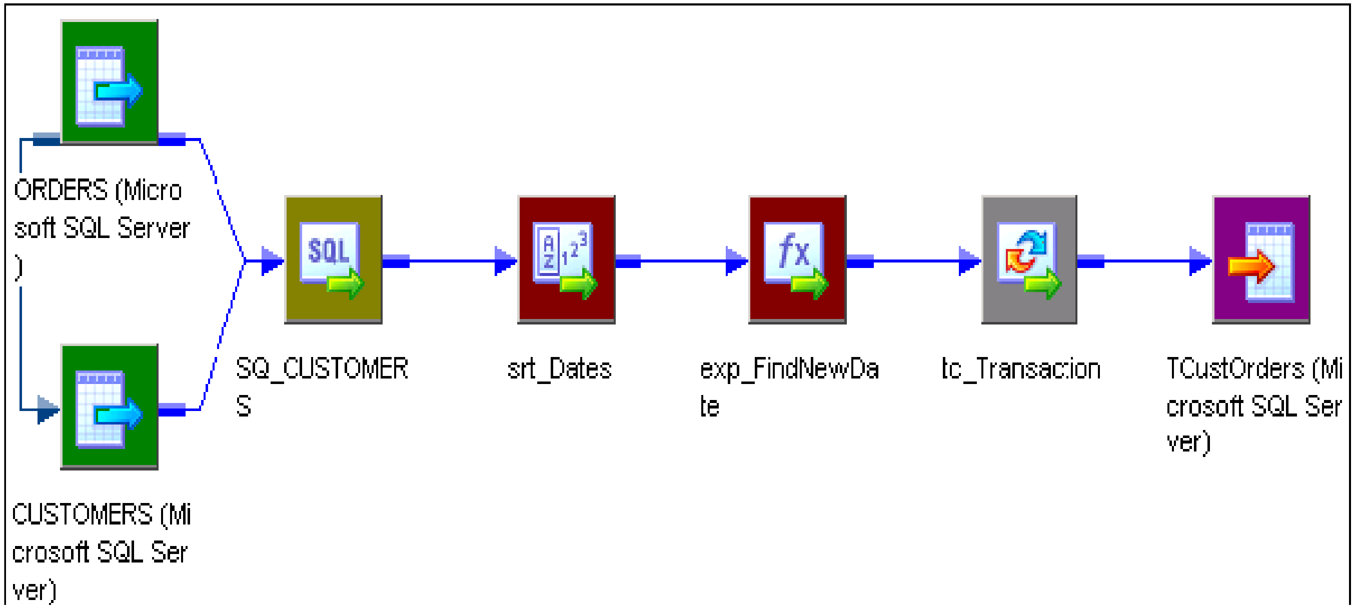
Transformation Attribute	Value
Transaction Control Condition	IIF(NEWTRAN = 1, TC_COMMIT_AFTER, TC_CONTIN...
Tracing Level	Normal

Transaction Control Condition

Transaction Control Expression

Transaction Control Transformation

IIF(NEW_DATE = 1, TC_COMMIT_BEFORE, TC_CONTINUE_TRANSACTION)


ORDERS (Micro soft SQL Server)
CUSTOMERS (Mi crosoft SQL Ser ver)
SQL
SQ_CUSTOMER S
srt_Dates
exp_FindNewDa te
tc_Transacion
TCustOrders (Mi crosoft SQL Ser ver)

Transaction Control T/F – Guidelines

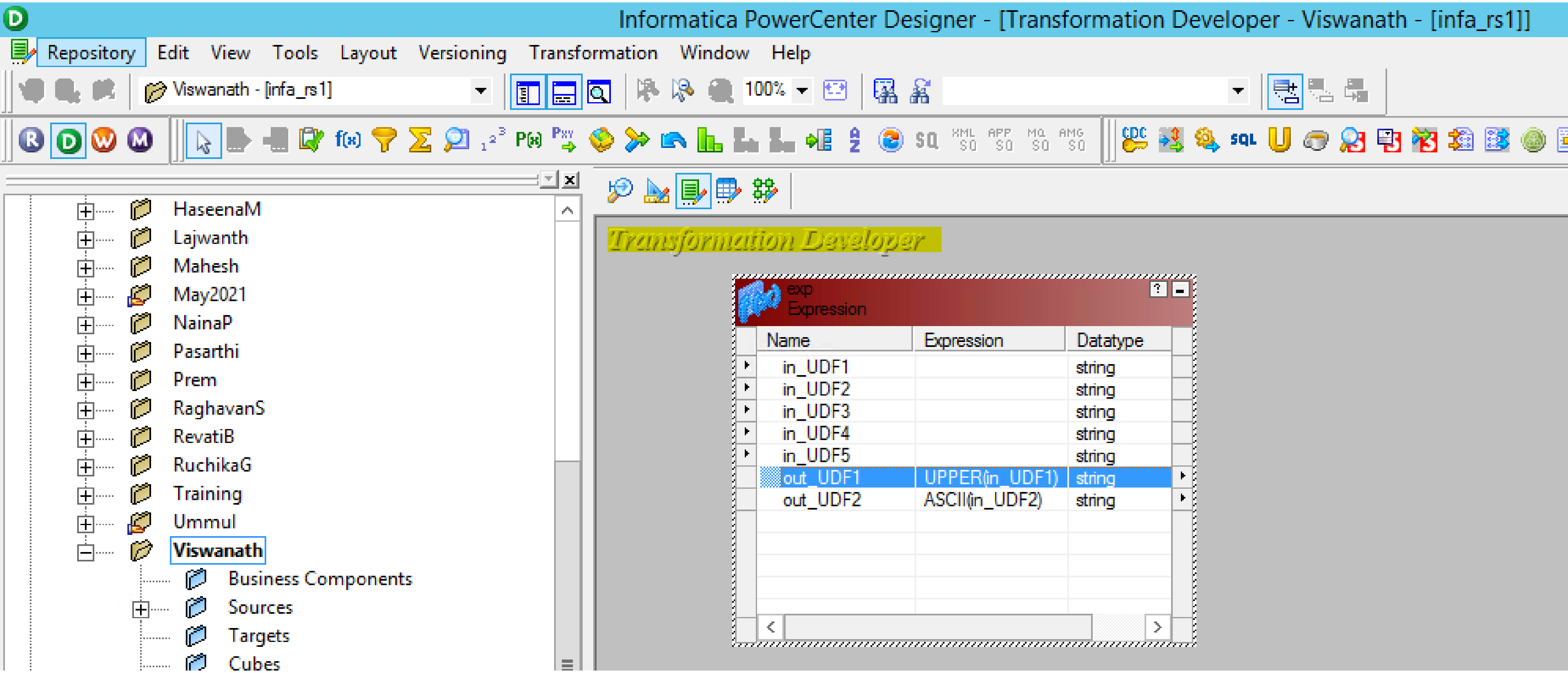
Use the following rules and guidelines when you create a mapping with a Transaction Control transformation:

- If the mapping includes an XML target, and you choose to append or create a new document on commit, the input groups must receive data from the same transaction control point.
- You must connect each target instance to a Transaction Control transformation.
- You can connect multiple targets to a single Transaction Control transformation.
- You can connect only one effective Transaction Control transformation to a target.
- You cannot place a Transaction Control transformation in a pipeline branch that starts with a Sequence Generator transformation.
- If you use a dynamic Lookup transformation and a Transaction Control transformation in the same mapping, a rolled-back transaction might result in unsynchronized target data.
- A Transaction Control transformation may be effective for one target and ineffective for another target. If each target is connected to an effective Transaction Control transformation, the mapping is valid.
- Either all targets or none of the targets in the mapping should be connected to an effective Transaction Control transformation

Reusable Transformation

Use Transformation Developer: Create reusable transformations.

- When a transformation is created under Transformation Developer, it becomes reusable across multiple different Mapplets or Mappings.



Informatica PowerCenter Designer - [Transformation Developer - Viswanath - [infa_rs1]]

- HaseenaM
- Lajwanth
- Mahesh
- May2021
- NainaP
- Pasarthi
- Prem
- RaghavanS
- RevatiB
- RuchikaG
- Training
- Ummul
- Viswanath
  - Business Components
  - Sources
  - Targets
  - Cubes

Transformation Developer

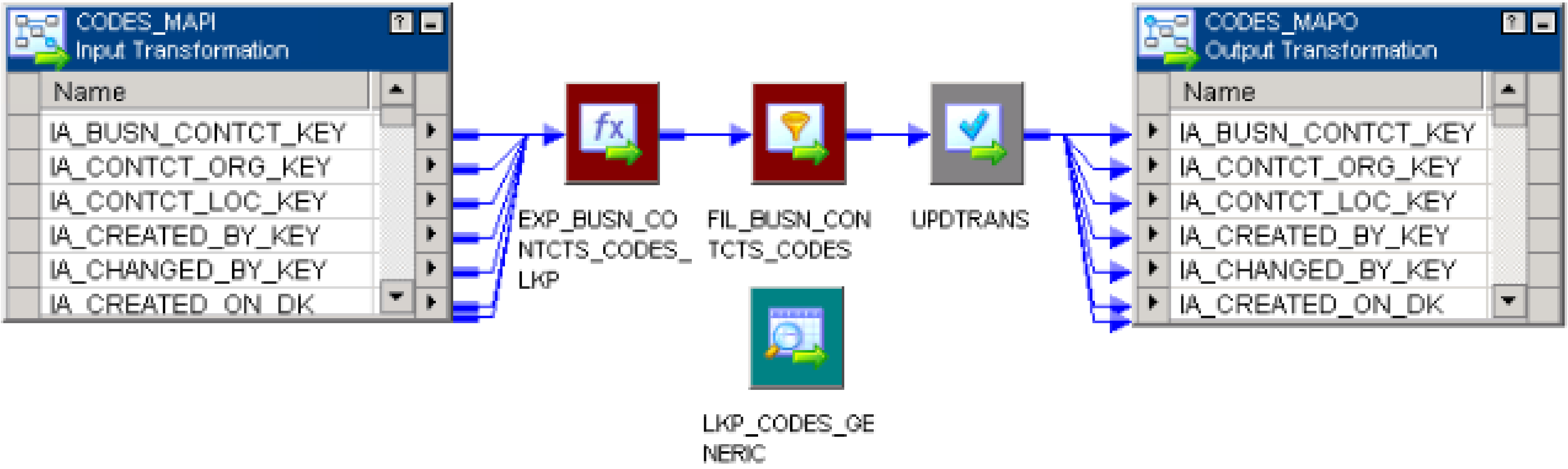
exp
Expression

Name	Expression	Datatype
in_UDF1		string
in_UDF2		string
in_UDF3		string
in_UDF4		string
in_UDF5		string
out_UDF1	UPPER(in_UDF1)	string
out_UDF2	ASCII(in_UDF2)	string

Mapplets

- A mapplet is a reusable object that you create in the Mapplet Designer. It contains a set of transformations and lets you reuse the transformation logic in multiple mappings.
- For example, if you have several fact tables that require a series of dimension keys, you can create a mapplet containing a series of Lookup transformations to find each dimension key. You can then use the mapplet in each fact table mapping, rather than recreate the same lookup logic in each mapping.

Mapplet Designer



Parameters and Variables

- If you declare mapping parameters and variables in a mapping, you can reuse a mapping by altering the parameter or variable values of the mapping in the session. Reuse a mapping to reduce the overhead of creating multiple mappings when only certain attributes of a mapping need to be changed.
- Use : You first declare the mapping parameter or variable for use in each mapplet or mapping. Then, you define a value for the mapping parameter or variable before you run the session.
- You can declare mapping parameters with the same name for use in a mapplet and a mapping. If the mapping parameter in a mapplet does not have a value, the parameter takes the value of the mapping parameter with the same name in the mapping.
- Use mapping parameters and variables in a mapping to incrementally extract data. Use mapping parameters or variables in the source filter of a Source Qualifier transformation to determine the beginning time stamp and end time stamp for incrementally extracting data.
- For example, you can create a user-defined mapping variable $$LastUpdateDateTime to save the time stamp of the last row that the Integration Service read in the previous session. In the source filter, use $$LastUpdateDateTime for the beginning time stamp and the built-in variable $$$SessStartTime for the end time stamp. Use the following filter to incrementally extract data based on the SALES.sales_datetime column in the source:
- SALES.sales_datetime > TO_DATE (‘$$LastUpdateDateTime’) AND SALES.sales_datetime < TO_DATE (‘$$$SessStartTime’)
- **Mapping Parameters :** A mapping parameter represents a constant value that you can define before running a session. A mapping parameter retains the same value throughout the entire session.
- **Mapping Variables :** Unlike a mapping parameter, a mapping variable represents a value that can change through the session. The Integration Service saves the value of a mapping variable to the repository at the end of each successful session run and uses that value the next time you run the session.

Demo and Q&A

- Demo
- Mappings covering the following areas
  - Normalizer Transformation
  - Transaction Control Transformation
  - Reusable Transformation
  - Mapplet
  - Parameters and Variables

Thank You