Informatica PC Training

Day-5

Agenda:

- Lookup Transformation
- Update Strategy Transformation
- SCD1
- SCD2
- SCD3
- Demo

Debadatta Mohanty

HCL

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.**

Look Up Transformation

Look Up Transformation

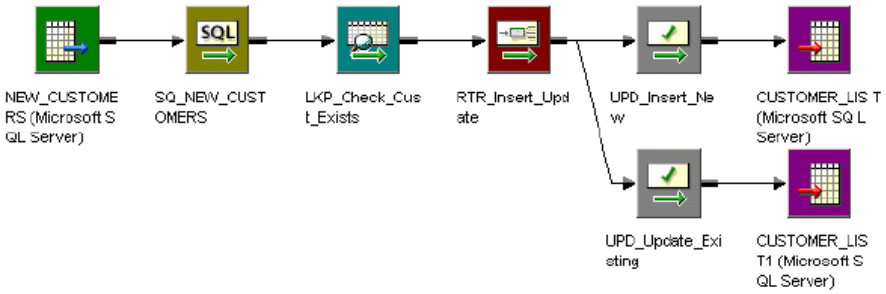
- Its created when data is to be looked up in a relational table, view, synonym or Flat File.
- The Integration Service queries the lookup table based on the lookup ports in the transformation.
- It compares Lookup transformation port values to lookup table column values based on the lookup condition.
- It's active transformation.
- Can use the Lookup transformation to perform many tasks, including:
  - Get a related value
  - Perform a calculation
  - Update slowly changing dimension tables
- Lookup can be configured as
  - Connected or unconnected
  - Relational or flat file lookup
  - Cached or un-cached
- Types Of Lookup's
  - Connected
  - Unconnected

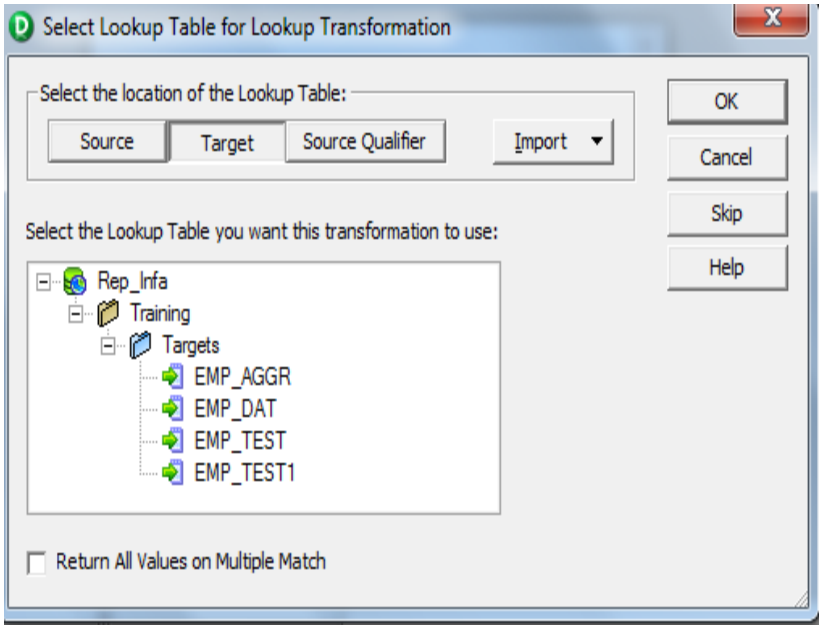
HCL

Connected Look Up Transformation

Connected Lookup Transformation

- Receives input values directly from another transformation in the pipeline
- For each input row, the Integration Service queries the lookup table or cache based on the lookup ports and the condition in the transformation
- Passes return values from the query to the next transformation
- It's uses Static Cache and Dynamic Cache.
- It will have Input , Output and Look up ports only.





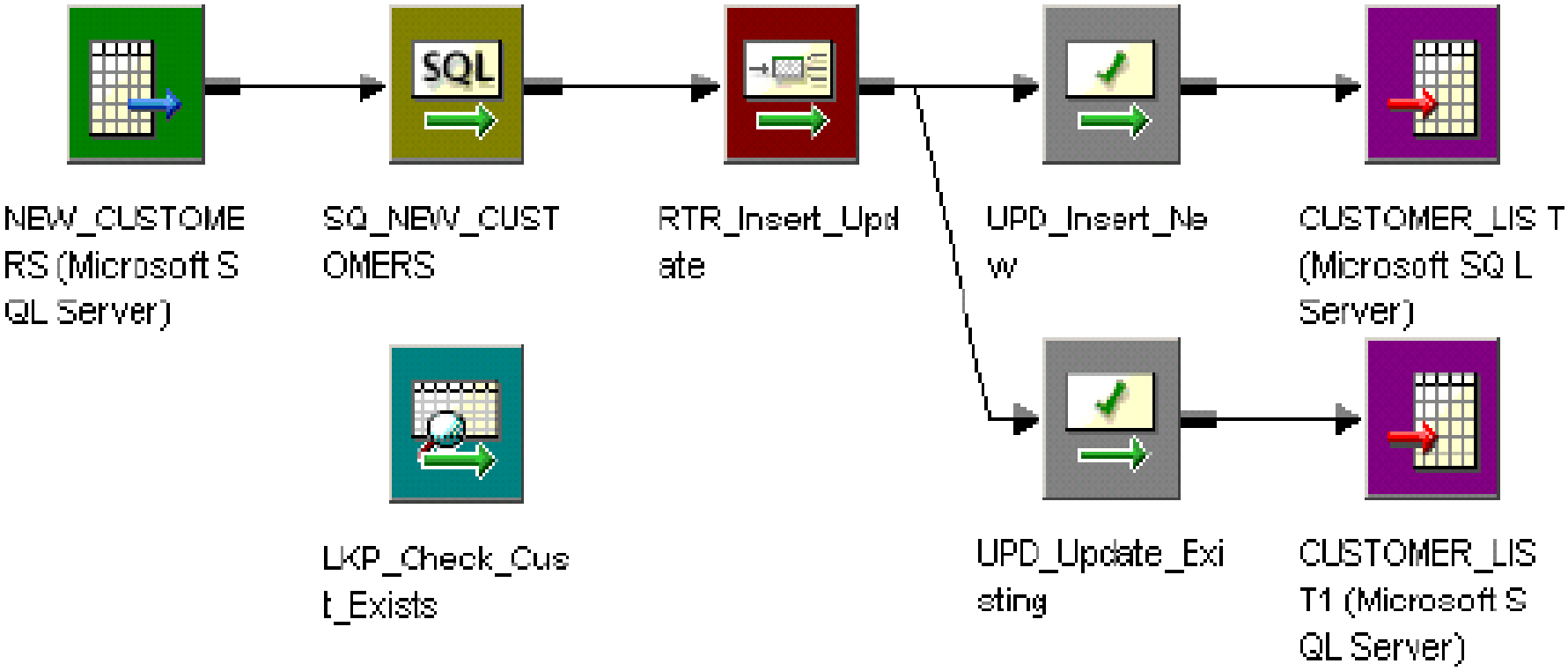
Un Connected Look Up Transformation

Unconnected Lookup Transformation:

- Receives input values from an expression using the :LKP (:LKP.lookup_transformation_name(argument, argument,...)) reference qualifier to call the lookup and returns one value.
- Some common uses for unconnected lookups include,
  - Testing the results of a lookup in an expression,
  - Filtering records based on the lookup results,
  - Marking records for update based on the result of a lookup (for example, updating slowly changing dimension tables),
  - Calling the same lookup multiple times in one mapping.
- With unconnected Lookups, you can pass multiple input values into the transformation, but only one column of data out of the transformation
- Use the return port to specify the return value in an unconnected lookup transformation
- It will have Lookup , Input, Output & Return Port .
- It does not support user-defined default value for return port. By default, return value is NULL
- It uses only Static Cache

HCL

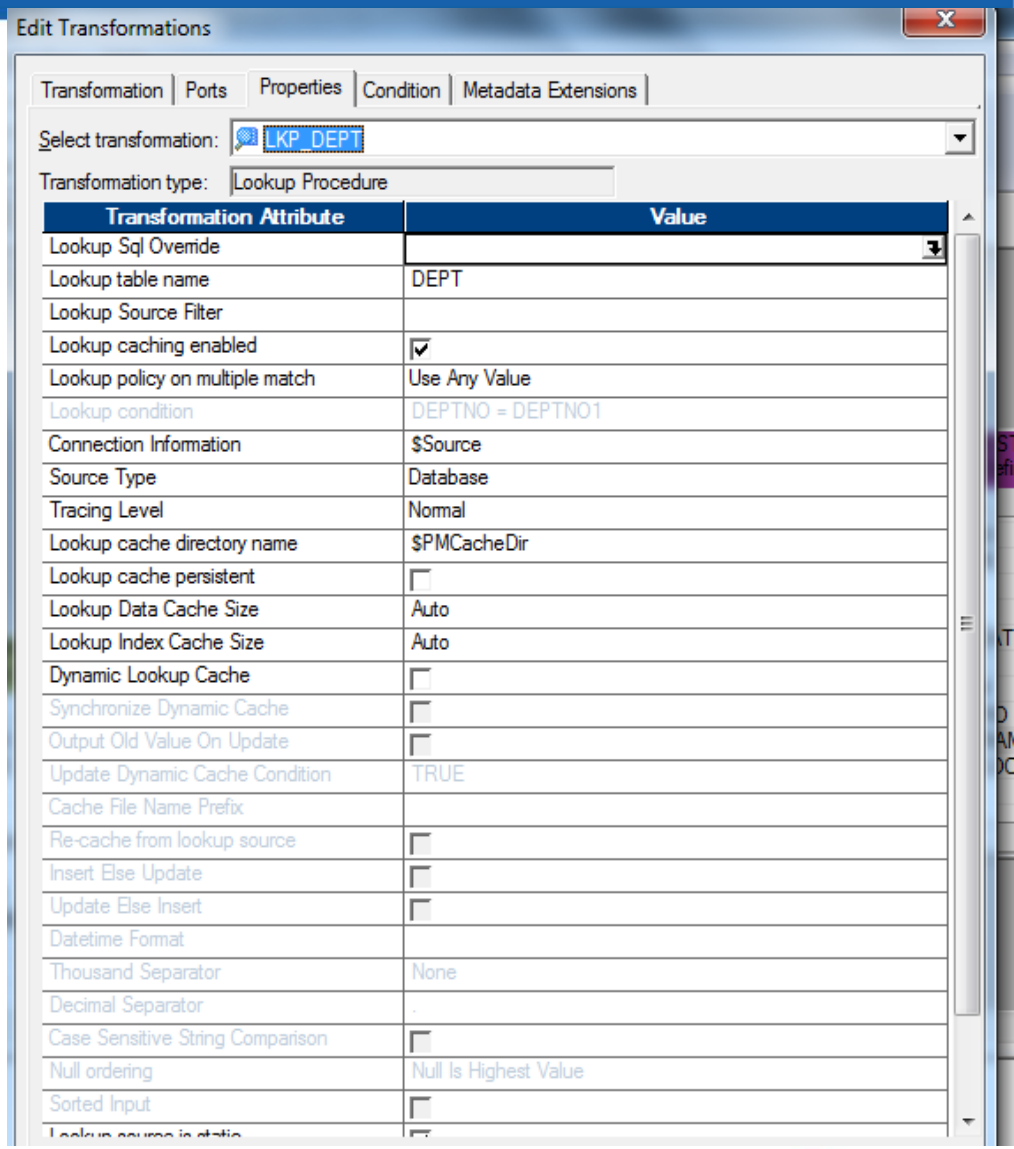
Un Connected Look Up Transformation



Look Up Transformation – Properties

Properties

- Lookup SQL Override
- Lookup Table Name
- Lookup Source Filter
- Lookup Caching Enabled
- Lookup Policy on Multiple Match
- Lookup Condition
- Connection Information
- Source Type
- Tracing Level
- Lookup Cache Directory Name
- Lookup Cache Persistent
- Dynamic Lookup Cache
- Recache From Lookup Source
- Lookup Source is Static
- Pre-build Lookup Cache



Look Up To Cache or not to Cache?

- ***Cached***
  - ✓ Lookup table data is cached locally on the Server
  - ✓ Mapping rows are looked up against the cache
  - ✓ Only one SQL SELECT is needed
- ***Uncached***
  - ✓ Each Mapping row needs one SQL SELECT
- ***Rule Of Thumb***: Cache if the number (and size) of records in the Lookup table is small relative to the number of mapping rows requiring lookup.
- The Integration Service builds a cache in memory when it processes the first row of data in a cached Lookup transformation
- The Integration Service stores Condition Values in the Index Cache and Output Values in Data Cache
- Cache files are created in $PMCacheDir by default. These files are deleted, and cache memory is released on completion of session, exception being Persistent Caches
- ***Lookup cache*** can be
  - ✓ Persistent Cache
  - ✓ Static Cache
  - ✓ Dynamic Cache
  - ✓ Shared Cache

HCL

Look Up Cache

- ***Persistent Cache*** – With this option, lookup cache files are created by Integration Service the first time and can be reused the next time Integration Service processes Lookup transformation, thus eliminating the time required to read the Lookup table. Lookup can be configured to use persistent cache when table does not change between sessions.

- ***Static Cache*** – By default, PowerCenter Server creates static or read-only cache for any lookup table. The Integration Service does not update the cache while it processes the Lookup Transformation.

- ***Dynamic Cache*** – Usual scenario where dynamic cache is used is when Lookup table is also a target table. The Integration Service dynamically inserts or updates data in Lookup cache

- ***Shared Cache*** – Lookup cache can be shared between multiple transformations. Unnamed cache can be shared between transformations of single mapping. Named cache can be shared between transformations of single mapping or across mappings

HCL

Look Up Cache

Connected Lookup	Unconnected Lookup
1) Receives input values directly from the pipeline.	1) Receives input values from the result of a :LKP expression in another transformation.
2) Use a dynamic or static cache.	2) Use a static cache.
3) Cache includes the lookup source columns in the lookup condition and the lookup source columns that are output ports.	3) Cache includes all lookup and output ports in the lookup condition and the lookup/return port.
4) Returns multiple columns from the same row or insert into the dynamic lookup cache.	4) Returns one column from each row to a return port.
5) If there is no match for the lookup condition, the Integration Service returns the default value for all output ports. If you configure dynamic caching, the Integration Service inserts rows into the cache or leaves it unchanged.	5) If there is no match for the lookup condition, the Integration Service returns NULL.
6) If there is a match for the lookup condition, the Integration Service returns the result of the lookup condition for all lookup/output ports. If you configure dynamic caching, the Integration Service either updates the row the in the cache or leaves the row unchanged.	6) If a match occurs for the lookup condition, the Integration Service returns the result of the lookup condition to the return port.
7) Passes multiple output values to another transformation. Link lookup/output ports to another transformation.	7) Returns one output value to another transformation. The Lookup transformation return port passes the value to the port that contains the :LKP expression in the other transformation.
8) Supports user-defined default values.	8) Does not support user-defined default values.

Update Strategy Transformation

Update Strategy

- ***An Active transformation***
- As part of the target table design, you need to determine whether to maintain all the historic data or just the most recent changes.
  - For example, you might have a target table, T_CUSTOMERS, that contains customer data. When a customer address changes, you may want to save the original address in the table instead of updating that portion of the customer row. In this case, you would create a new row containing the updated address and preserve the original row with the old customer address. This shows how you might store historical information in a target table. However, if you want the T_CUSTOMERS table to be a snapshot of current customer data, you will update the existing customer row and lose the original address.
- ***In PowerCenter, you set the update strategy at two different levels:***
  - ***Session level***– Within Session, you can instruct the Integration Service to either treat all rows in the same way (for example, treat all rows as inserts), or use instructions coded into the session mapping to flag rows for different database operations.
  - ***Mapping level***– Within a mapping, you use the Update Strategy transformation to flag rows for insert, delete, update, or reject.

HCL

Update Strategy

- ***Flagging Rows Within a Mapping :*** *The most important feature of this transformation is its update strategy expression, used to flag individual rows for insert, delete, update, or reject.*

Operation	Constant	Numeric Value
Insert	DD_INSERT	0
Update	DD_UPDATE	1
Delete	DD_DELETE	2
Reject	DD_REJECT	3

- ***Forwarding Rejected Rows :***
- *You can configure the Update Strategy transformation to either pass rejected rows to the next transformation or drop them. By default, the Integration Service forwards rejected rows to the next transformation. The Integration Service flags the rows for reject and writes them to the session reject file. If you do not select Forward Rejected Rows, the Integration Service drops rejected rows and writes them to the session log file.*
- *If you enable row error handling, the Integration Service writes the rejected rows and the dropped rows to the row error logs. It does not generate a reject file. If you want to write the dropped rows to the session log in addition to the row error logs, you can enable verbose data tracing.*

HCL

Update Strategy

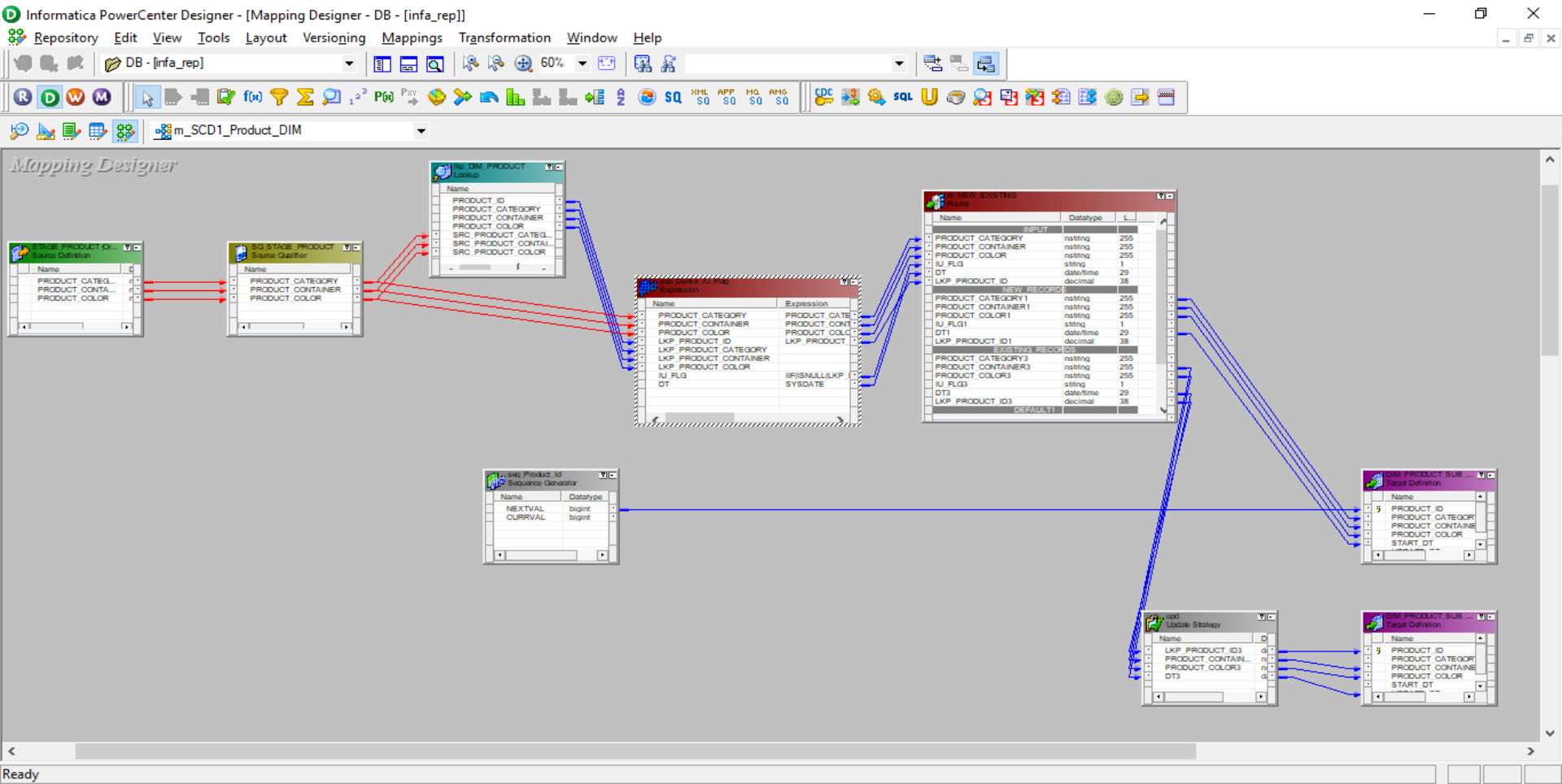
- ***Update Strategy Expressions***

- *Frequently, the update strategy expression uses the IIF or DECODE function from the transformation language to test each row to see if it meets a particular condition. If it does, you can then assign each row a numeric code to flag it for a particular database operation. For example, the following IIF statement flags a row for reject if the entry date is after the apply date. Otherwise, it flags the row for update:*

- *IIF((ENTRY_DATE > APPLY_DATE), DD_REJECT, DD_UPDATE)*

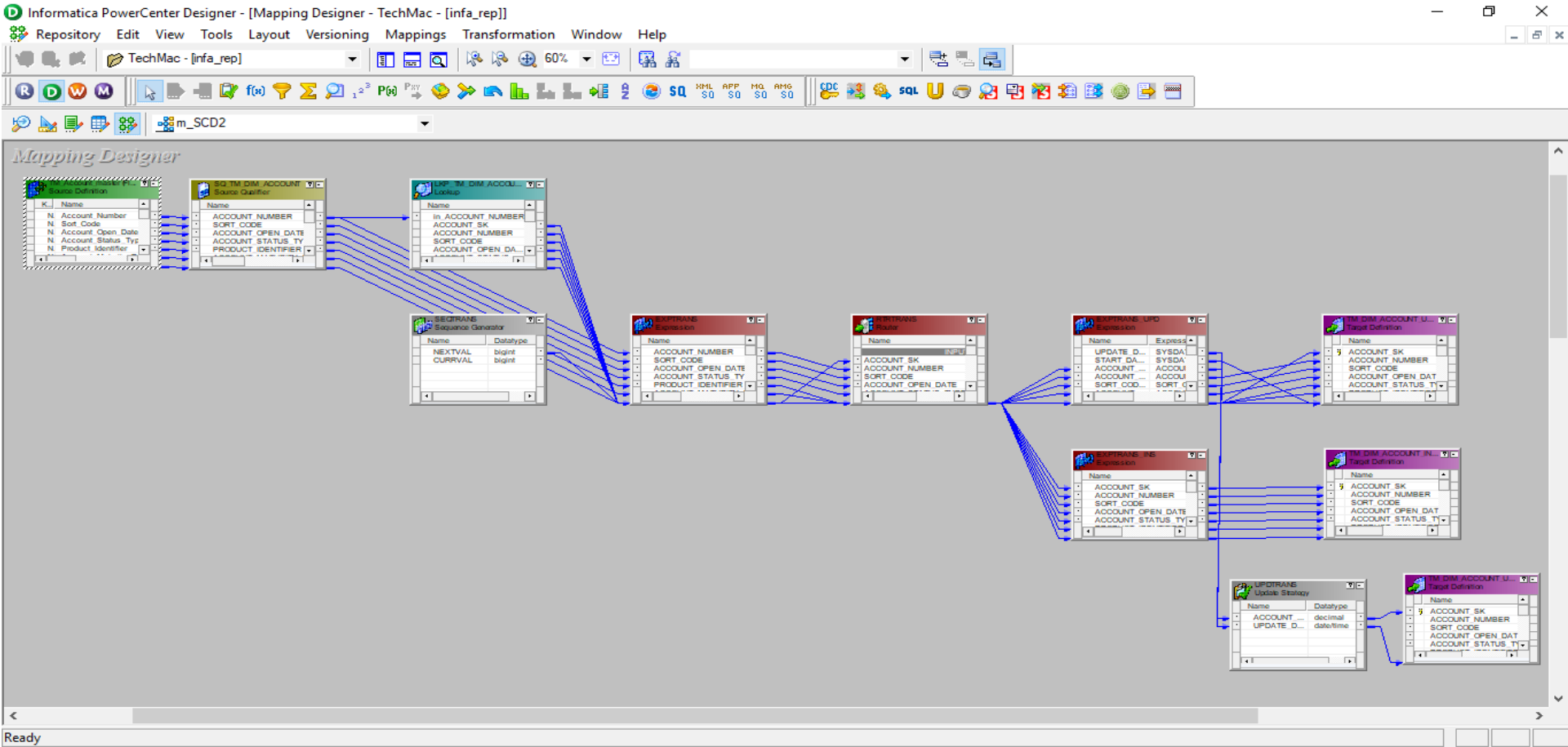
SCD1

- *SCD1 mapping filters source rows based on user-defined comparisons and inserts only those found to be new dimensions to the target. Rows containing changes to existing dimensions are updated in the target by overwriting the existing dimension. In SCD1 mapping, all rows contain current dimension data.*



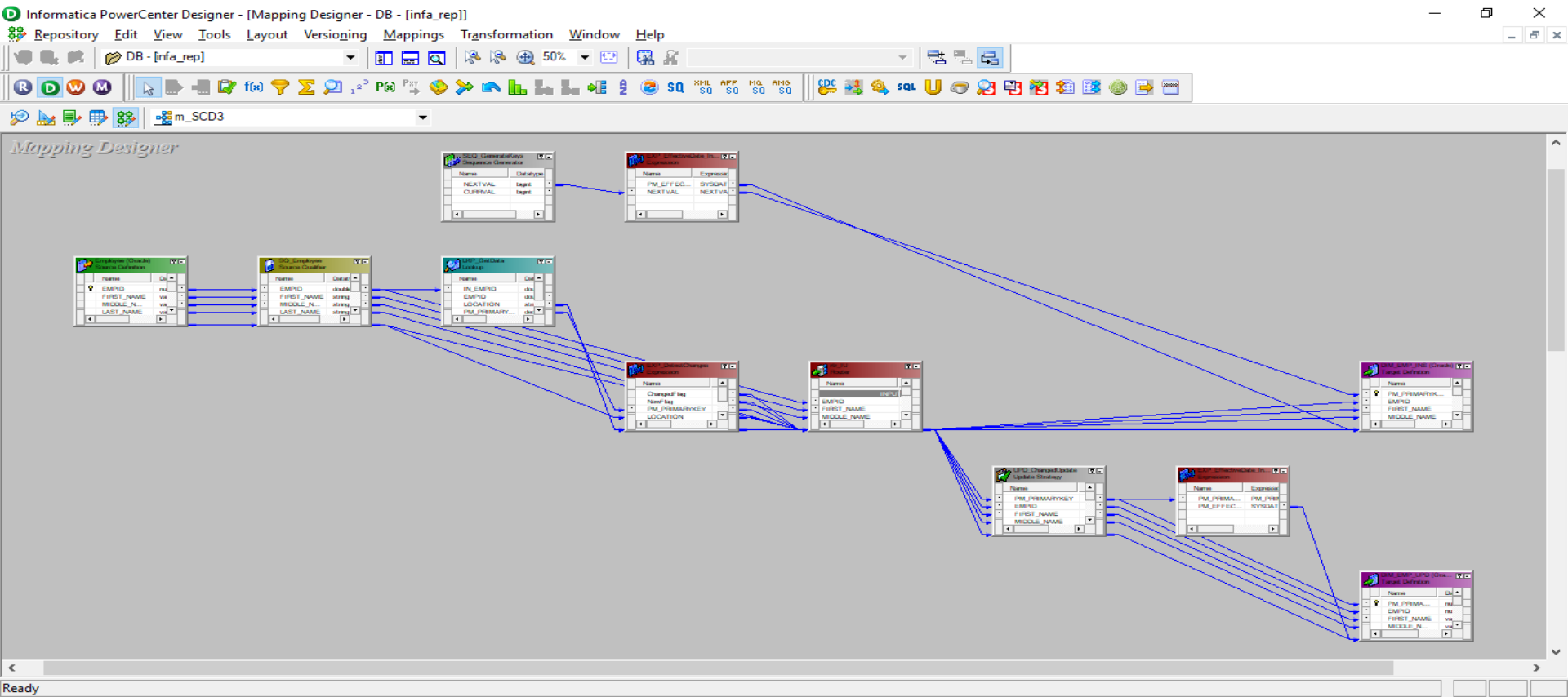
SCD2

- *The SCD2/Effective Date Range mapping filters source rows based on user-defined comparisons and inserts both new and changed dimensions into the target. Changes are tracked in the target table by maintaining an effective date range for each version of each dimension in the target. In the SCD2/Effective Date Range target, the current version of a dimension has a begin date with no corresponding end date.*



SCD3

- *SCD3 mapping filters source rows based on user-defined comparisons and inserts only those found to be new dimensions to the target. Rows containing changes to existing dimensions are updated in the target. When updating an existing dimension, IS saves existing data in different columns of the same row and replaces the existing data with the updates. IS optionally enters the system date as a timestamp for each row it inserts or updates. In the Type 3 Dimension target, each dimension contains current dimension data.*



Demo and Q&A

- Demo
- Mappings covering the following areas
  - Rank Transformation
  - Update Strategy Transformation
  - SCD1
  - SCD2
  - SCD3

Thank You