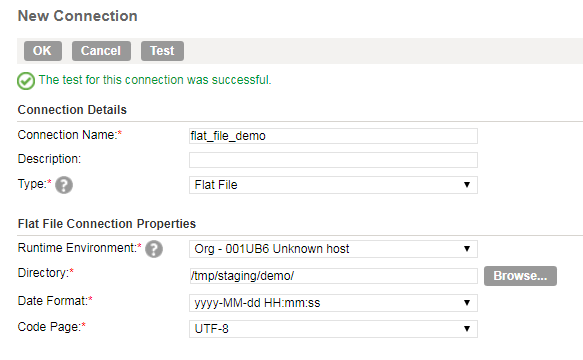
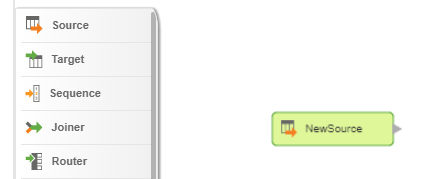
**INFORMATICA CLOUD**

1. **How to create flat file connection**
2. Go to the Configure tab 🡪 Connections
3. Add parameters & click OK



1. How to configure Source
2. Flat File

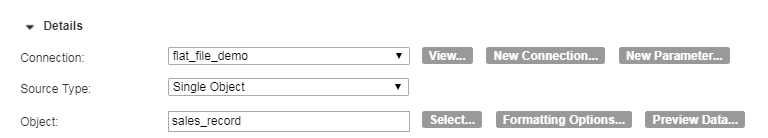
* Drag the Source stage into the mapping canvas.

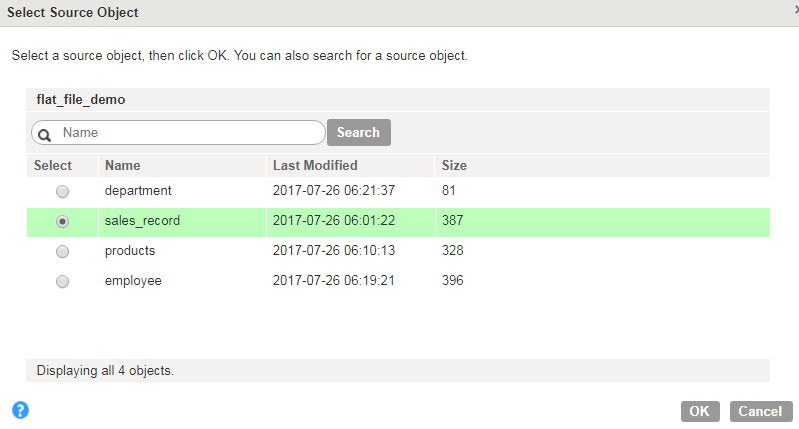


* Rename the stage under General



* Add connection. To read a specific flat file, choose Single Object as Source Type.
* Select Source Object

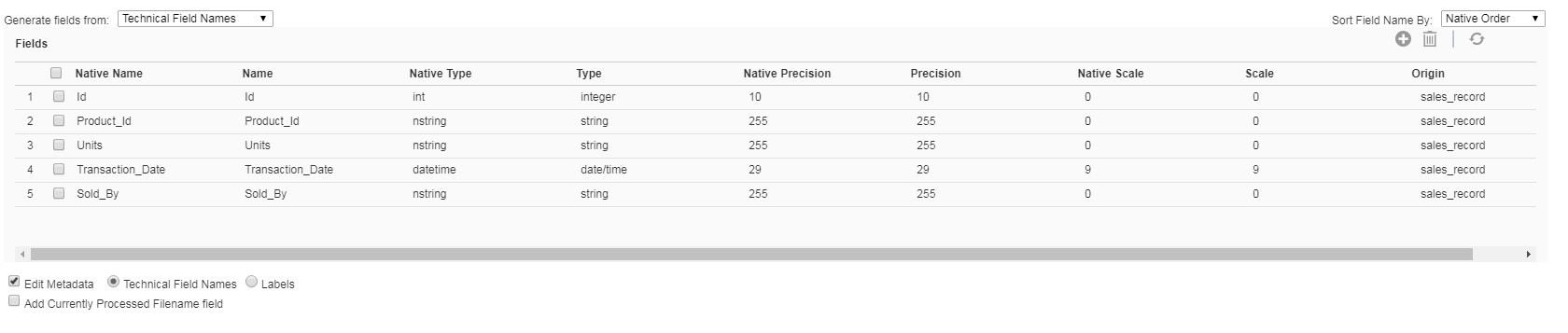




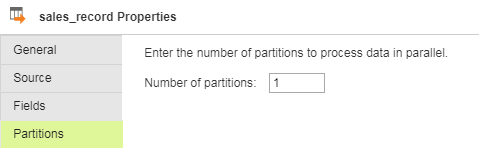
Under Advanced, we can set Tracing Level (Terse, Normal, Verbose Initialization, Verbose Data). This will determine what information goes to the session log file.



* Need to edit metadata. When you read from flat file, default data type is set to string



* We can set the number of partitions to process data in parallel.



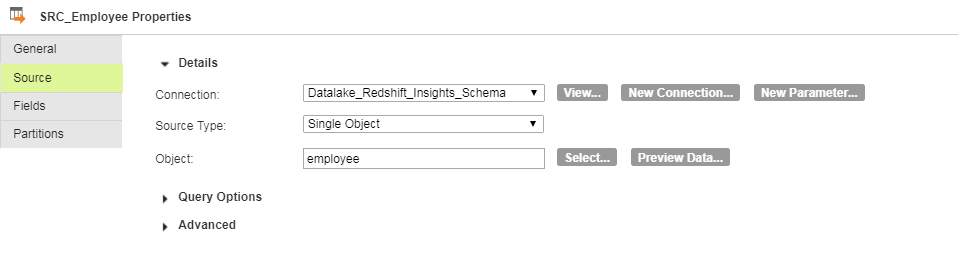
1. Database

Redshift

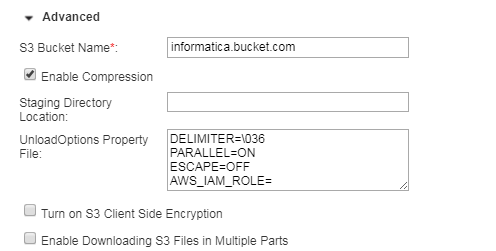
1. Name the Source stage in the General tab



1. Choose Connection, Source Type and Object

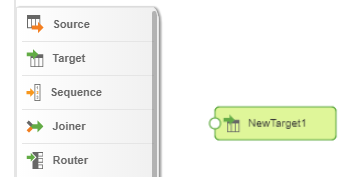


1. Redshift connector needs s3 bucket name defined in the advanced settings.



1. Salesforce
2. How to configure Target
3. Flat File

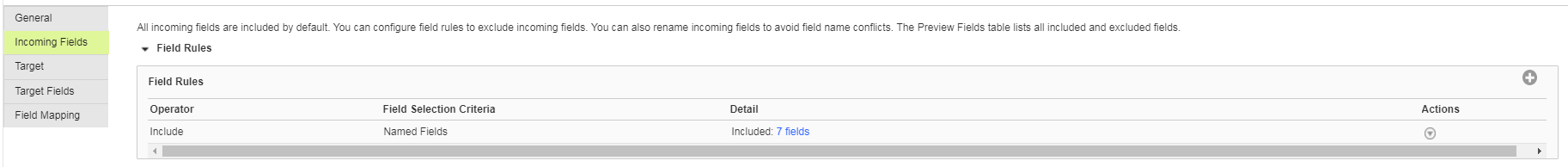
* Drag the Target stage into the mapping canvas

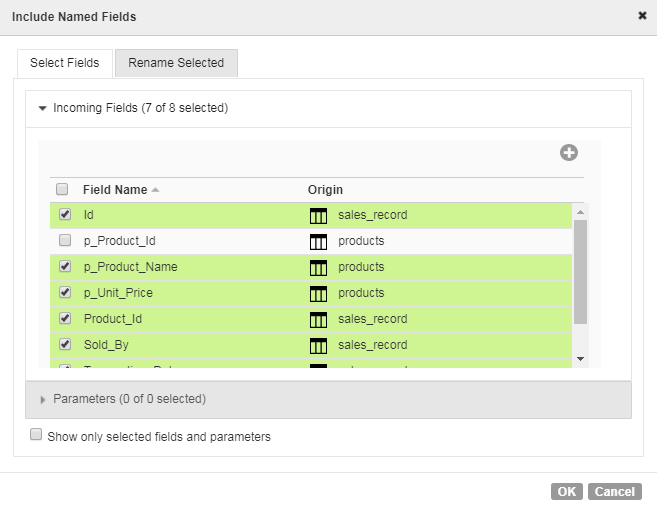


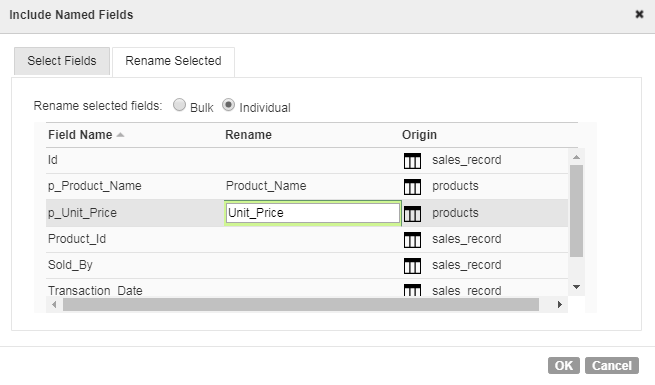
* Rename the stage



* Choose which fields to include. We can rename any included field individually.



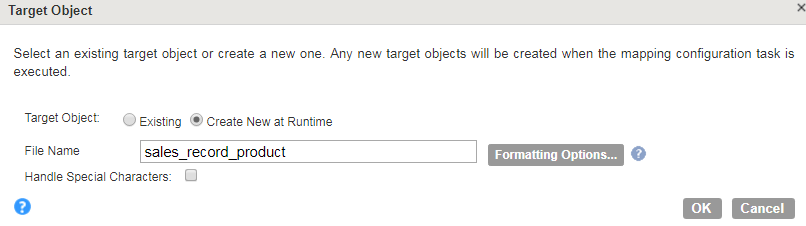




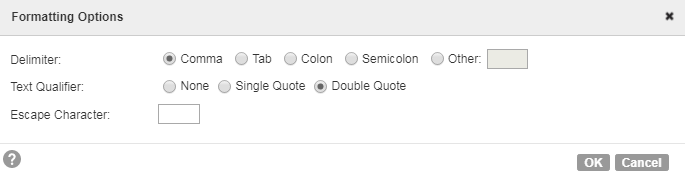
* Configure Target



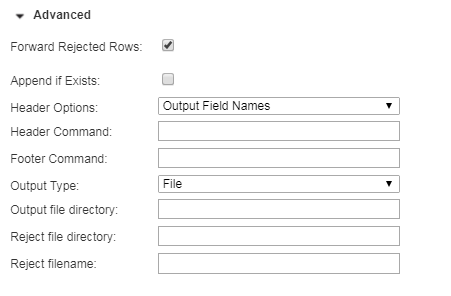
We can create a new file at runtime (click the select button in the object section)



We can change the format options



* We can also configure advanced section.



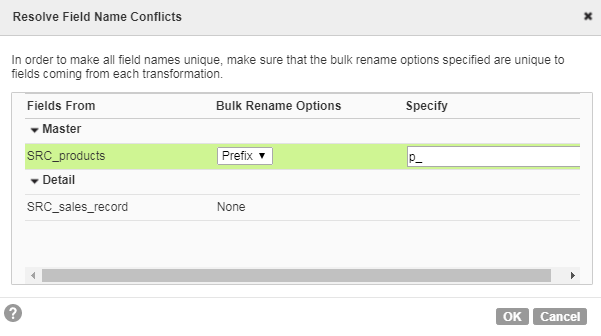
1. Database
2. Salesforce

**INFORMATICA CLOUD TRANSFORMATION**

1. **Joiner Transformation**

In this example, we add product name and price to the sales record.

1. Go to the Design tab 🡪 Select Mappings 🡪 Select New Mapping
2. When two tables have the same column name, we need to rename one of them by Bulk Rename.



1. Choose Join Type

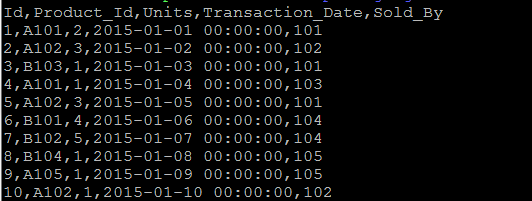
Master Outer keeps all rows of data from the detail source and matching rows from the master source. It discards the unmatched rows from the master source.

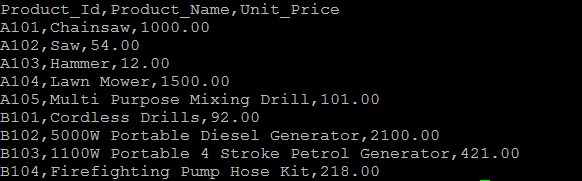
Detail Outer keeps all rows of data from the master source and the matching rows from the detail source. It discards the unmatched rows from the detail source.

Normal discards all rows of data from the master and detail source that do not match, based on the condition

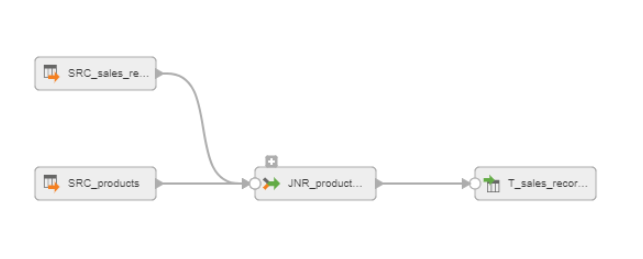
Full Outer keeps all rows of data from both the master and detail sources.

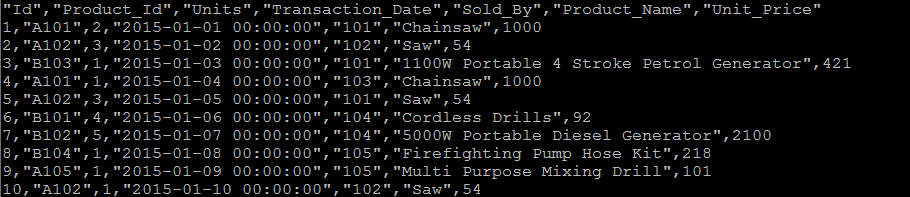
In this case, Sales\_Record is in Detail and Product is in Master. As not all products are in the sales\_record, we use master outer, which will essentially work as left joining product on sales\_record.





See configure source and target



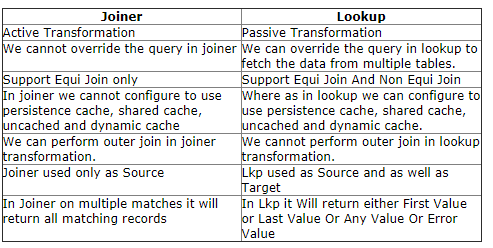


1. Lookup Transformation

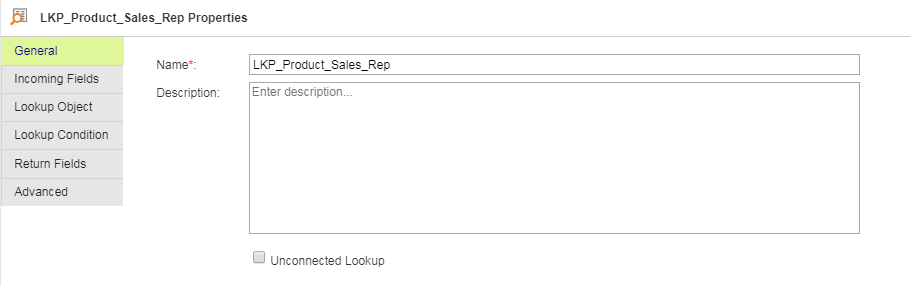
In this example, we add product name and price to the sales record by looking up the product table.

<http://www.javaorator.com/informatica/interview/difference-between-lookup-and-joiner-transformation-47.code>

<https://informaticareference.wordpress.com/2011/12/07/difference-between-joiner-transformation-and-lookup-transformation/>

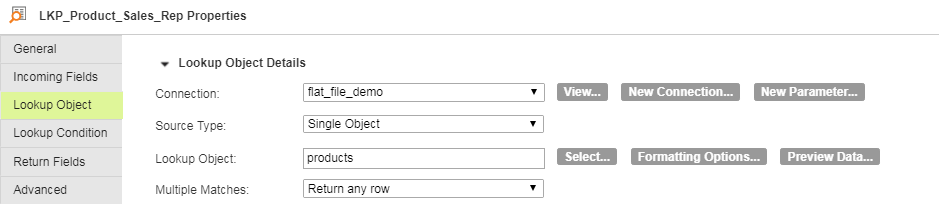


1. Rename the Lookup stage

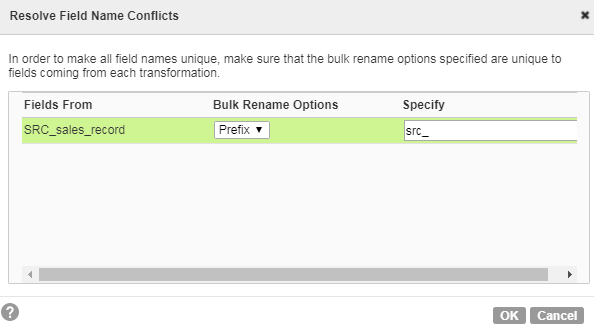


1. Choose Lookup object

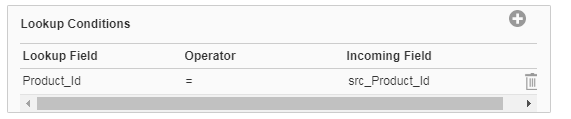
In the lookup stage, we can have different options for multiple matches (Return any row, Return first row, Return last row, Return all rows, Report error).



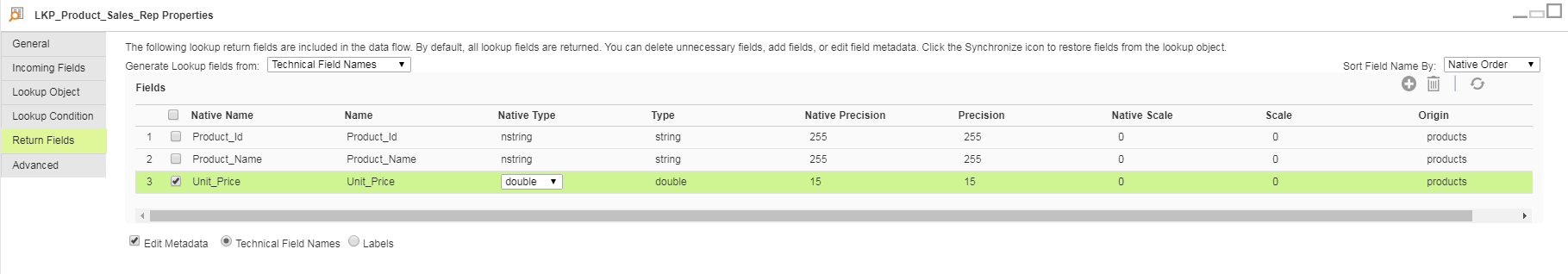
1. We can resolve field name conflicts



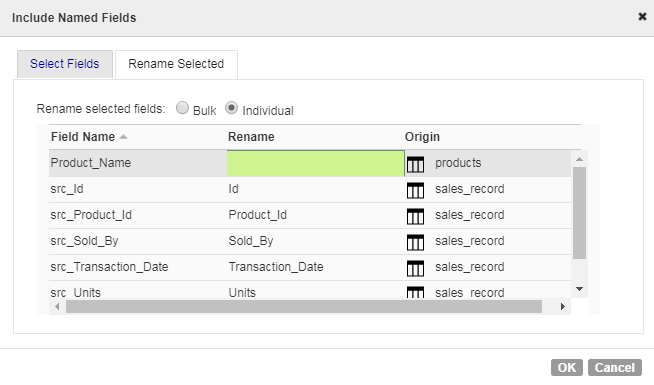
1. Add the lookup condition



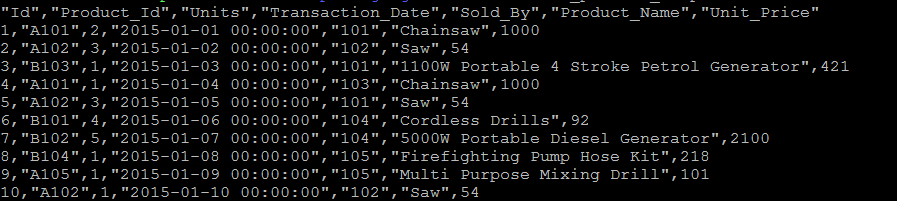
1. Edit the data type in the lookup table in Return Fields



1. Choose the columns to be included in the output file and rename columns in the target stage.



Output



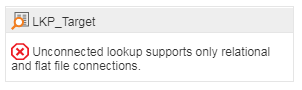


**Slowly Changing Dimension (Type 1 with Upsert)**

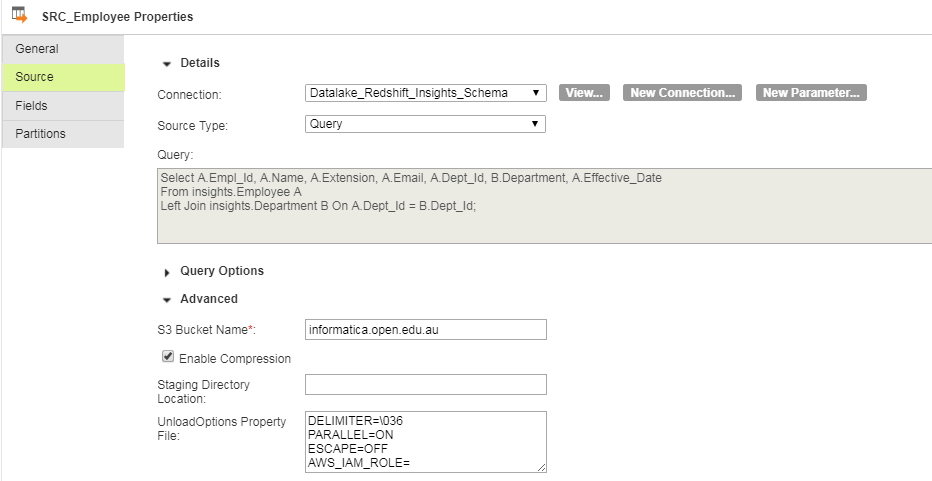
**Source: Employee**

Unconnected lookup does not work on Redshift. It only works on flat file or relational connections. The following connection types are relational, ODBC, MySql, Oracle, SqlServer and MS\_ACCESS.

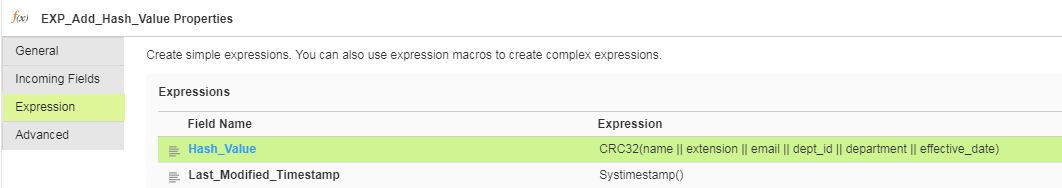
With slowly changing dimension, we could use unconnected lookup.



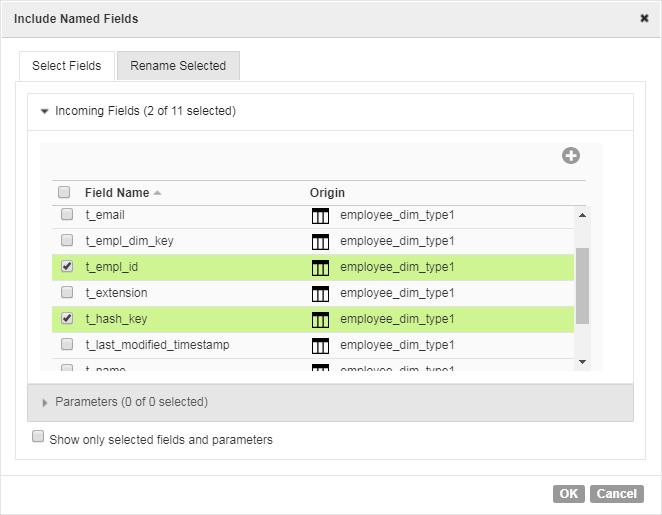
1. Use Query as Source Type and join Employee and Department in order to add Department name.

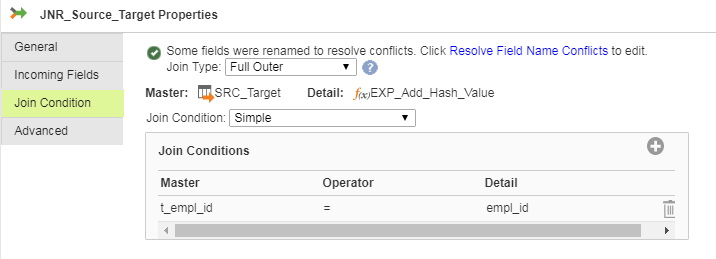


1. Add Hash\_Value to the non-key columns of the source table

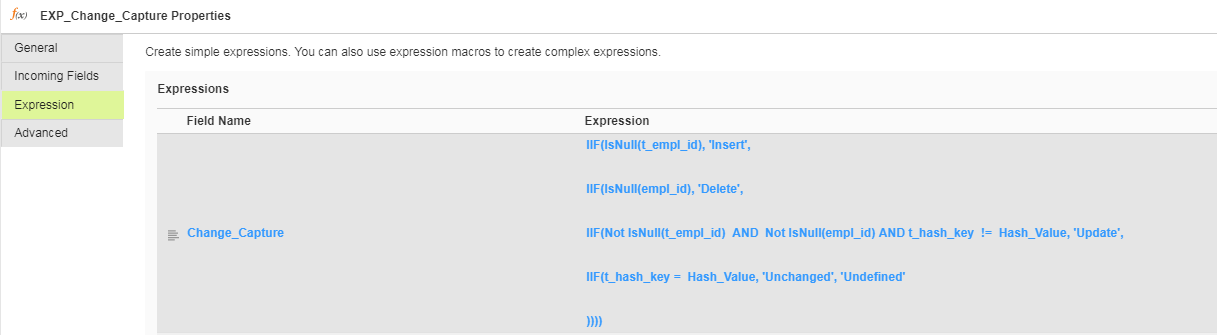


1. Full outer join on empl\_id (source primary key) with Target and Source table. Bring hash value from the target table for comparison.

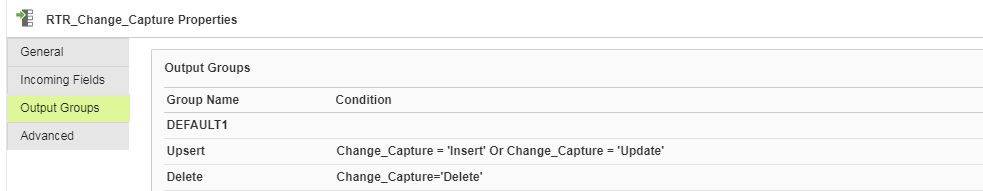




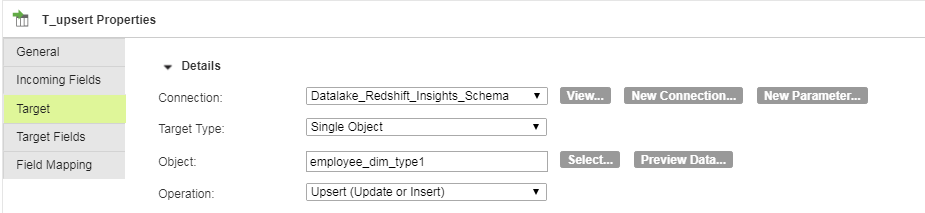
1. Add change capture value

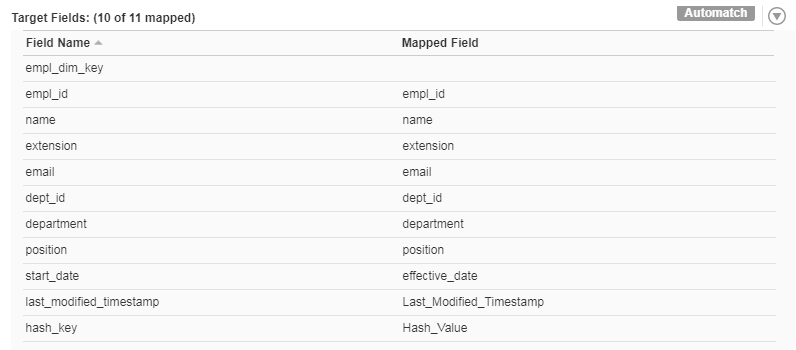


1. Use Router Stage to Split records into delete and upsert.

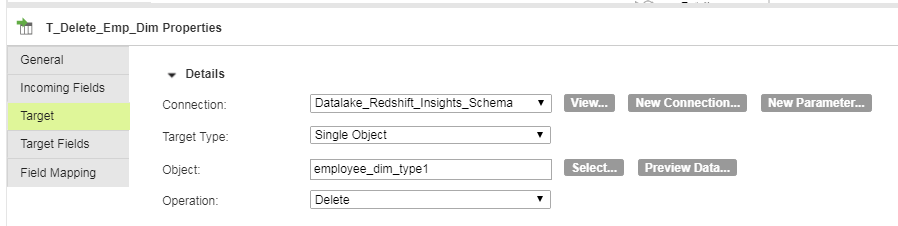


1. For upsert, include all the relevant field in the mapping. Empl\_dim\_key is defined as an identity column in Redshift table definition and mapping is not required.



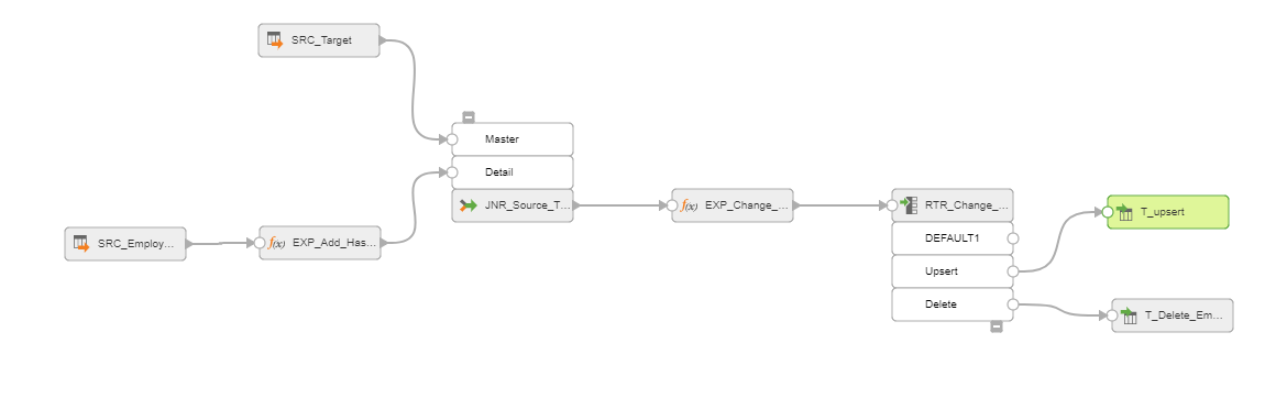


For delete, only map the primary key.

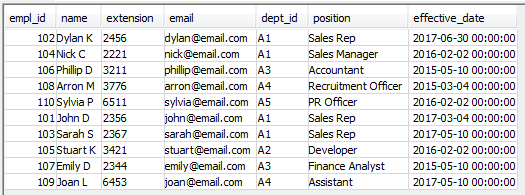


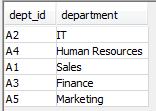


Outcome

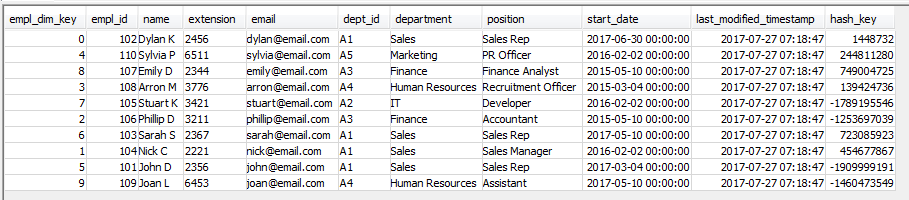


Source Table





Dimension Table



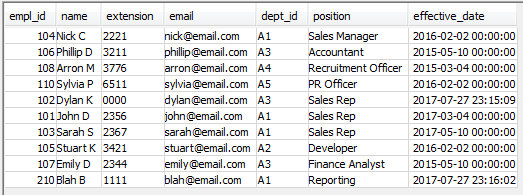
Update, Insert, Delete on Employee

Update insights.Employee Set extension = '0000', dept\_id = 'A3', effective\_date = GetDate() Where empl\_id = 102;

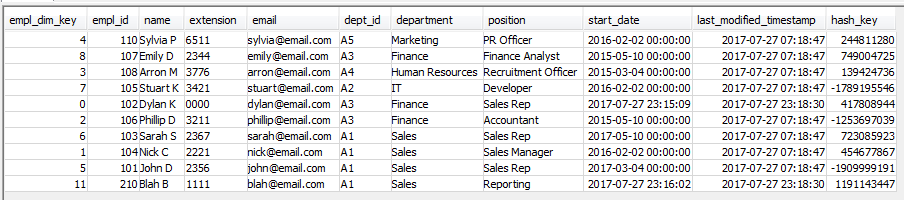
Delete From insights.Employee where empl\_id = 109;

Insert Into insights.Employee Values

(210, 'Blah B', '1111', 'blah@email.com', 'A1', 'Reporting', GetDate());

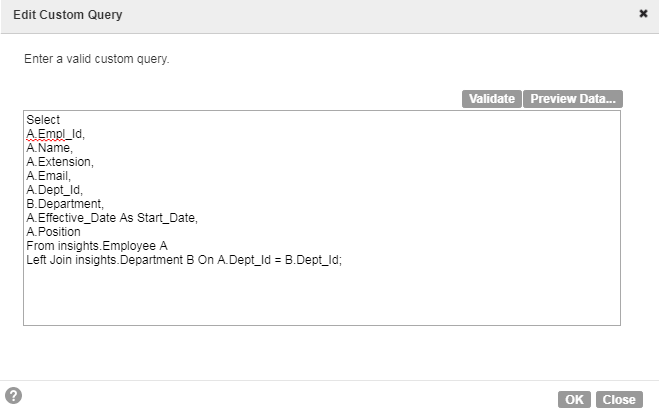


Dimension table with changed values

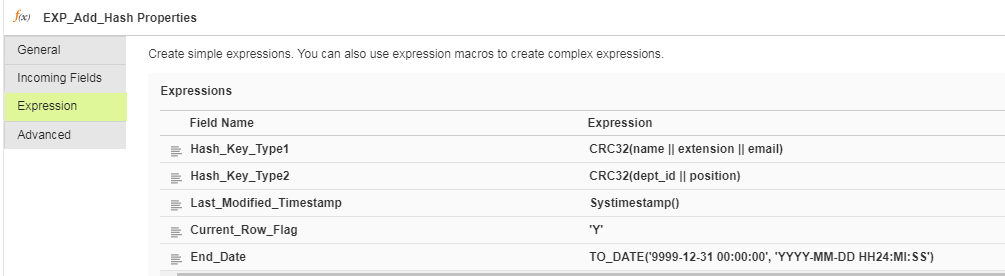


**Slowly Changing Dimension (Type 2 by using Change Capture Strategy)**

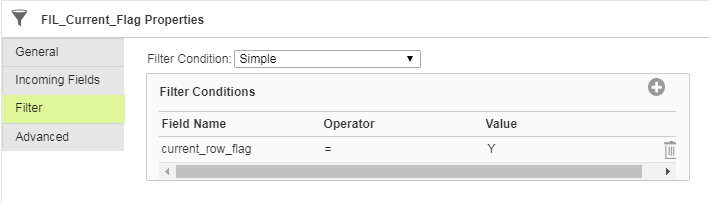
1. Join Source table in the Source stage



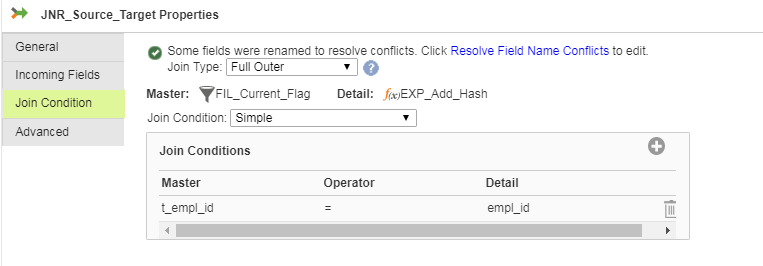
1. Add dimension columns with Expression stage from the Source table.



1. Bring the dimension table with only Current\_Row\_Flag is set to ‘Y’



1. Full Outer Join on the Source and Dimension Table on the Business Key



1. Create a Change Capture Columns

IIF(IsNull(empl\_id), 'Delete',

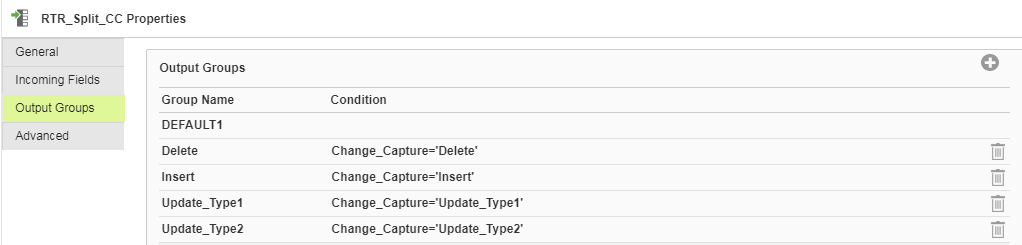
IIF(IsNull(t\_empl\_id), 'Insert',

IIF (Not IsNull(empl\_id) And Not IsNull(t\_empl\_id) And Hash\_Key\_Type1 != t\_hash\_key\_type1 And Hash\_Key\_Type2 = t\_hash\_key\_type2, 'Update\_Type1',

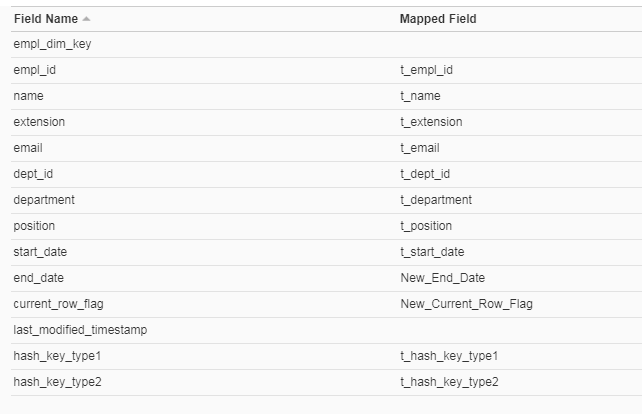
IIF (Not IsNull(empl\_id) And Not IsNull(t\_empl\_id) And Hash\_Key\_Type2 != t\_hash\_key\_type2, 'Update\_Type2' ))))



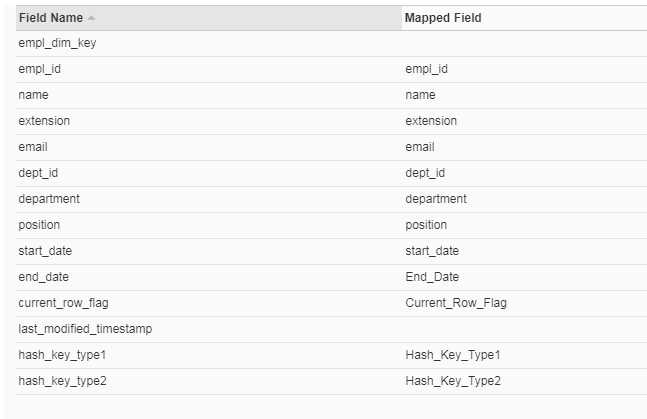
1. Filter by change capture



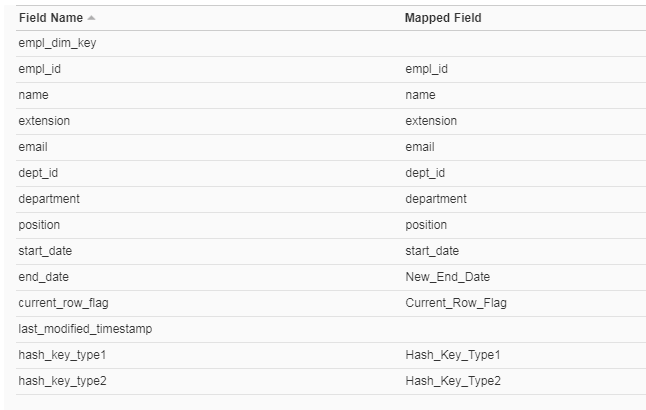
1. **Upsert in Database**
2. **Deleted records**



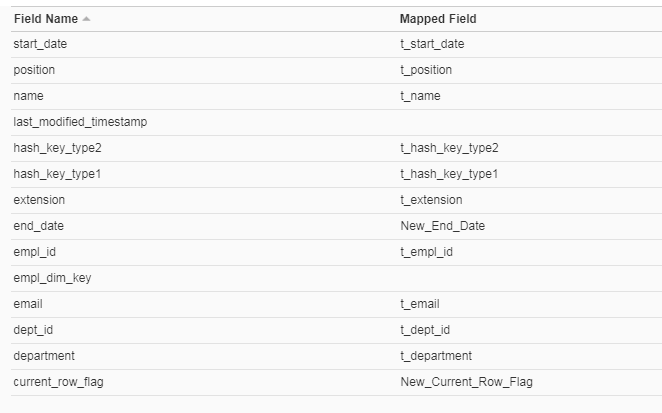
1. **Inserting new records**



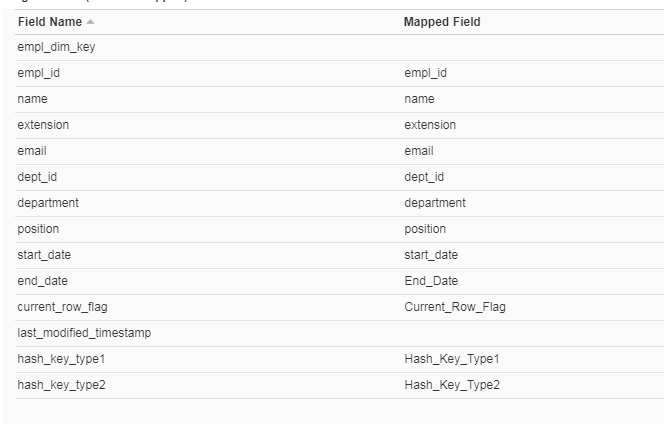
1. **Type 1 Record update**



1. **Type 2 Record update**

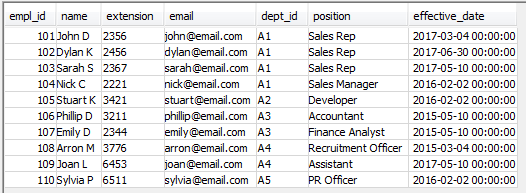


1. **Type 2 record insert**

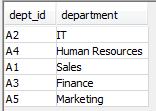


**Outcome**

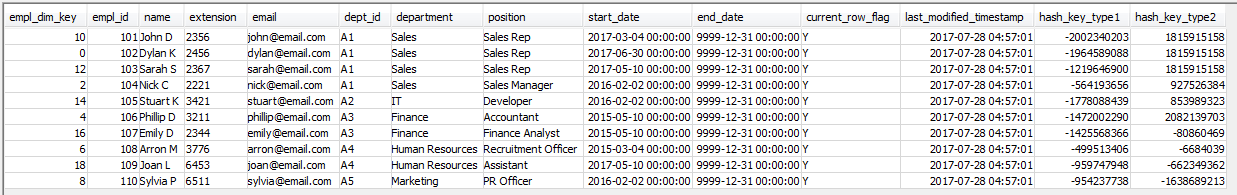
Employee



Department



Dim type2 Initial Load



Update, Insert, Delete

Update insights.Employee Set extension = '0000', dept\_id = 'A3', effective\_date = GetDate() Where empl\_id = 102;

Update insights.Employee Set extension = '5675' Where empl\_id = 104;

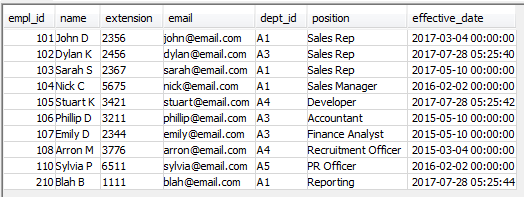
Update insights.Employee Set extension = '5675', dept\_id = 'A4' Where empl\_id = 105;

Delete From insights.Employee where empl\_id = 109;

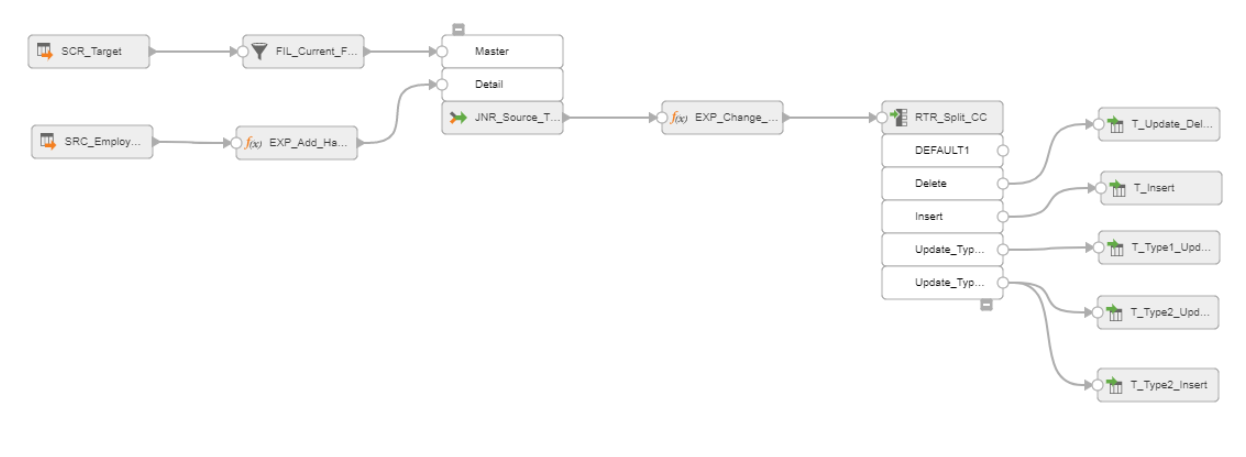
Insert Into insights.Employee Values

(210, 'Blah B', '1111', 'blah@email.com', 'A1', 'Reporting', GetDate());

Employee table- updated

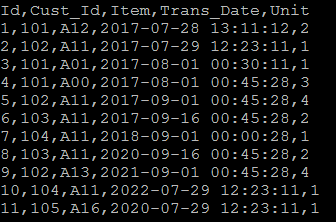


Mapping

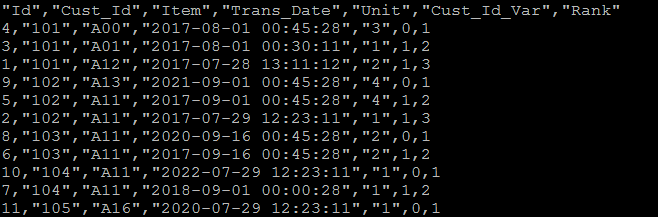


**Rank Transformation with Sorter and Expression**

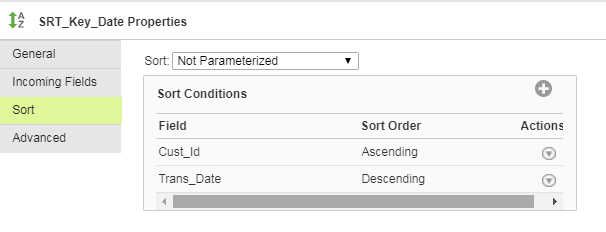
**Input**



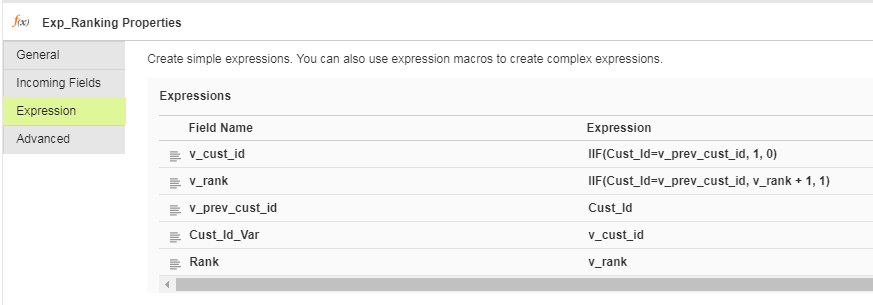
**Output**

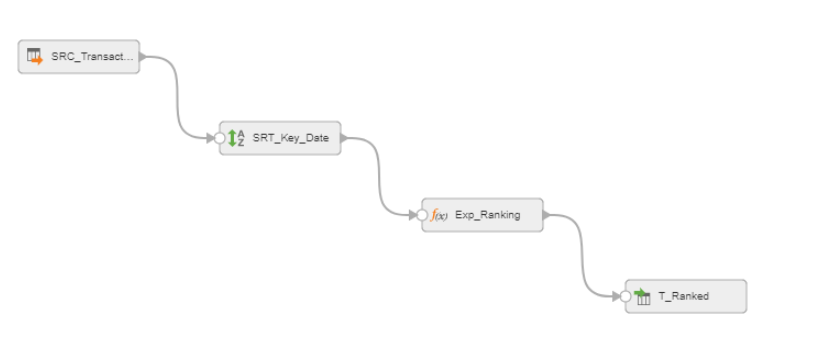


1. Sort the transaction record by Business Key (Cust\_Id) in an ascending order and by tran\_date in descending order.



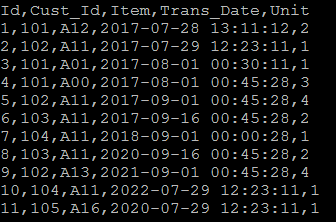
1. User variables in the expression stage to increment rank



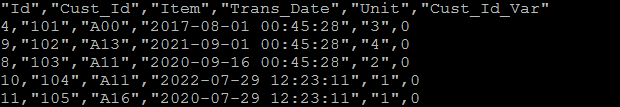


**Remove Duplicate Transformation with Sorter and Expression**

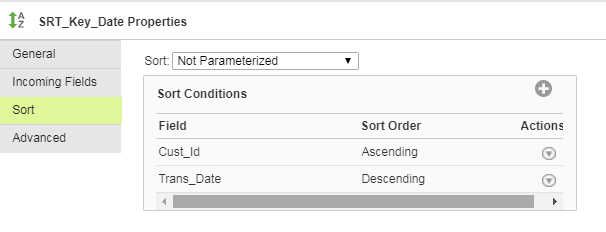
**Input**



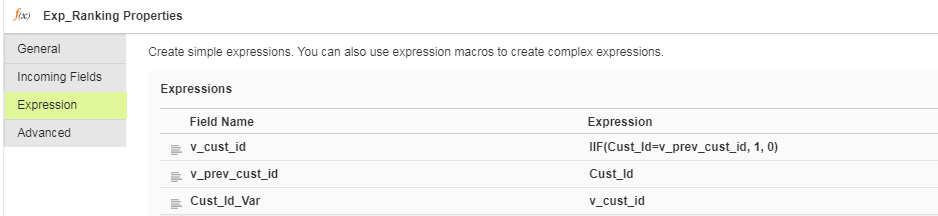
Output



1. Sort the transaction record by Business Key (Cust\_Id) in an ascending order and by tran\_date in descending order.



1. Expression stage



1. Filter only 0

