

GENESYS

This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

iWD Data Mart Reference Guide

Customization Example

Customization Example

This example creates a new statistic transformation script, product_pending overdue.ktr, that calculates how many pending and overdue tasks there are for a particular product. You start by copying the process_pendingoverdue transformation script, which calculates similar statistics for the PROCESS dimension. This script references the TASK_CLASSIF_FACT subject area, which supports a join to the PRODUCT dimension.

- 1. In the aggregate_stats\stats subdirectory, copy the process_pending overdue.ktr transformation script and rename it as follows: product_pendingoverdue.ktr
- 2. Open this script in Kettle ETL Designer.
- 3. Within the Transformation window, right-click and select Transformation Settings from the context menu (shown below) to open the Transformation properties dialog box.



4. Rename the transformation appropriately and click OK. The figure below uses the name product_pendingoverdue.

Transformation properties						
Transformation Logging Date	es Dependencies Miscellaneo					
Transformation name :	product_pendingoverdue					
Description :						
Extended description:						

5. Double-click the get_stats step to open the Table input dialog box that is shown below. Next, you must update the logic for the calculation of this statistic.

•						
	🗧 Table input	🗾 🗖 🐱				
	Step name	get_stats				
select_params	Connection	GTL_DM 📃 Edit New				
1	squ	Get SQL select statement				
	select ? as TENANTID, ? as SOLUTIONID, ? as STATSERVICEID, P.PROCESS_RUNTIME_ID as DIMENSIONID, sum(C.TOTAL_OVERDUE_TASK_COUNT) as OVERDUE, sum(C.TOTAL_PENDING_TASK_COUNT) as OVERDUE, sum(C.TOTAL_PENDING_TASK_COUNT) as PENDING from PROCESS_CURRENT P laft.join LTABK_CLASSIF_FACT_15MIN C on C.PROCESS_KEY = P.PROCESS_KEY and C.INTERVAL_KEY = ? aroup by P_PROCESS_RUNTIME_ID					
	<	5				
	Replace variables in script7					
	Insert data from step	select_params				
	Execute for each row?					
:ompleted_null_to_0	Limit size	0				
		OK Preview Cancel				

6. Replace the SQL statement with the following and click OK:

SELECT ?	AS TENANTID ,
?	AS SOLUTIONID ,
?	AS STATSERVICEID,
P.PRODUCT_TYPE	AS DIMENSIONID ,
SUM(C.TOTAL OVERDUE TASK COUNT)	AS OVERDUE ,
SUM(C.TOTAL PENDING TASK COUNT)	AS PENDING
FROM PRODUCT P	
LEFT JOIN I_TASK_CLASSIF_FACT_15M	IN C
ON C.PRODUCT_KEY = P.PRODUCT_KEY	
AND C.INTERVAL_KEY = ?	
GROUP BY P.PRODUCT TYPE	

You use a left join on I_TASK_CLASSIF_FACT_15MIN, instead of an inner join because you want to retrieve data about all products, whether or not they have tasks associated with them.

- 7. Double-click the add_dimensiontype step to open the Add constant values dialog box.
- 8. Set the value of DIMENSIONTYPE to some string, and click OK. The figure below sets this string to PRD.

C Add constant values								
	Step name	add_dimensiontype						
Fields :								
🔺 Name 😽	Туре	Format	Length	Precision	Currency	Decimal	Group	Value
1 DIMENSIONTYPE	String							PRD
OK Cancel								

- 9. Double-click the add_dimensionid_prefix step to open the Script Values dialog box.
- 10. Change the script to use the dimension-type string that was assigned in Step 8, as shown in the figure below, and click 0K.

JS Script Values / Mo	d					
\searrow	Step name	add_dimensionid_p	prefix			
Java script functions :	Java script :					
🕀 Transform Scripts	🔞 Script 1 👷					
Transform Constants Transform Functions Transform Functions Output Fields Output Fields						
Position: 1, 25						
Fields						
Fieldname	Rename to 👘 Type	Length	Precision			
1						
OK Get variables Test script Cancel						

- 11. Close the Designer, saving all work.
- 12. Stop iWD Data Mart runtime node.
- 13. In the aggregate_stats\stats subdirectory, using an ASCII editor, edit the stats.properties file to enable the statistic. Add the last line shown in the example below, and then save and close the file:

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\department_activeheld.ktr

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\department_newcompleted.ktr

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\department_pendingoverdue.ktr

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\process_activeheld.ktr

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\process_pendingoverdue.ktr

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\solution_newcompleted.ktr

\${KETTLE_REPOS_DIR}\aggregate_stats\stats\product_pendingoverdue.ktr

This new product_pendingoverdue script is now ready for ETL to calculate overdue and pending tasks on its next run.