DataStage Performance Tuning

Performance Tuning - Basics

BasicsParallelism Parallelism in DataStage Jobs should be optimized rather than maximized. The degree of parallelism of a DataStage Job is determined by the number of nodes that is defined in the Configuration File, for example, four-node, eight –node etc. A configuration file with a larger number of nodes will generate a larger number of processes and will in turn add to the processing overheads as compared to a configuration file with a smaller number of nodes. Therefore, while choosing the configuration file one must weigh the benefits of increased parallelism against the losses in processing efficiency (increased processing overheads and slow start up time).Ideally , if the amount of data to be processed is small , configuration files with less number of nodes should be used while if data volume is more , configuration files with larger number of nodes should be used.

Partioning :

Proper partitioning of data is another aspect of DataStage Job design, which significantly improves overall job performance. Partitioning should be set in such a way so as to have balanced data flow i.e. nearly equal partitioning of data should occur and data skew should be minimized.

Memory :

In DataStage Jobs where high volume of data is processed, virtual memory settings for the job should be optimised. Jobs often abort in cases where a single lookup has multiple reference links. This happens due to low temp memory space. In such jobs $APT_BUFFER_MAXIMUM_MEMORY, $APT_MONITOR_SIZE and $APT_MONITOR_TIME should be set to sufficiently large values.

Performance Analysis of Various stages in DataStag

Sequential File Stage -

The sequential file Stage is a file Stage. It is the most common I/O Stage used in a DataStage Job. It is used to read data from or write data to one or more flat Files. It can have only one input link or one Output link .It can also have one reject link. While handling huge volumes of data, this Stage can itself become one of the major bottlenecks as reading and writing from this Stage is slow.Sequential files should be used in following conditionsWhen we are reading a flat file (fixed width or delimited) from UNIX environment which is FTPed from some external systemsWhen some UNIX operations has to be done on the file Don’t use sequential file for intermediate storage between jobs. It causes performance overhead, as it needs to do data conversion before writing and reading from a UNIX file.In order to have faster reading from the Stage the number of readers per node can be increased (default value is one).

Data Set Stage :
The Data Set is a file Stage, which allows reading data from or writing data to a dataset. This Stage can have a single input link or single Output link. It does not support a reject link. It can be configured to operate in sequential mode or parallel mode. DataStage parallel extender jobs use Dataset to store data being operated on in a persistent form.Datasets are operating system files which by convention has the suffix .dsDatasets are much faster compared to sequential files.Data is spread across multiple nodes and is referred by a control file.Datasets are not UNIX files and no UNIX operation can be performed on them.Usage of Dataset results in a good performance in a set of linked jobs.They help in achieving end-to-end parallelism by writing data in partitioned form and maintaining the sort order.

Lookup Stage –
A Look up Stage is an Active Stage. It is used to perform a lookup on any parallel job Stage that can output data. The lookup Stage can have a reference link, single input link, single output link and single reject link.Look up Stage is faster when the data volume is less.It can have multiple reference links (if it is a sparse lookup it can have only one reference link)The optional reject link carries source records that do not have a corresponding input lookup tables.Lookup Stage and type of lookup should be chosen depending on the functionality and volume of data.Sparse lookup type should be chosen only if primary input data volume is small.If the reference data volume is more, usage of Lookup Stage should be avoided as all reference data is pulled in to local memory

Join Stage :
Join Stage performs a join operation on two or more datasets input to the join Stage and produces one output dataset. It can have multiple input links and one Output link.There can be 3 types of join operations Inner Join, Left/Right outer Join, Full outer join. Join should be used when the data volume is high. It is a good alternative to the lookup stage and should be used when handling huge volumes of data.Join uses the paging method for the data matching.

Merge Stage :
The Merge Stage is an active Stage. It can have multiple input links, a single output link, and it supports as many reject links as input links. The Merge Stage takes sorted input. It combines a sorted master data set with one or more sorted update data sets. The columns from the records in the master and update data sets are merged so that the output record contains all the columns from the master record plus any additional columns from each update record. A master record and an update record are merged only if both of them have the same values for the merge key column(s) that you specify. Merge key columns are one or more columns that exist in both the master and update records. Merge keys can be more

than one column. For a Merge Stage to work properly master dataset and update dataset should contain unique records. Merge Stage is generally used to combine datasets or files.

Sort Stage :
The Sort Stage is an active Stage. The Sort Stage is used to sort input dataset either in Ascending or Descending order. The Sort Stage offers a variety of options of retaining first or last records when removing duplicate records, Stable sorting, can specify the algorithm used for sorting to improve performance, etc. Even though data can be sorted on a link, Sort Stage is used when the data to be sorted is huge.When we sort data on link (sort / unique option) once the data size is beyond the fixed memory limit , I/O to disk takes place, which incurs an overhead. Therefore, if the volume of data is large explicit sort stage should be used instead of sort on link.Sort Stage gives an option on increasing the buffer memory used for sorting this would mean lower I/O and better performance.

Transformer Stage :
The Transformer Stage is an active Stage, which can have a single input link and multiple output links. It is a very robust Stage with lot of inbuilt functionality. Transformer Stage always generates C-code, which is then compiled to a parallel component. So the overheads for using a transformer Stage are high. Therefore, in any job, it is imperative that the use of a transformer is kept to a minimum and instead other Stages are used, such as:Copy Stage can be used for mapping input links with multiple output links without any transformations. Filter Stage can be used for filtering out data based on certain criteria. Switch Stage can be used to map single input link with multiple output links based on the value of a selector field. It is also advisable to reduce the number of transformers in a Job by combining the logic into a single transformer rather than having multiple transformers .

Funnel Stage –
Funnel Stage is used to combine multiple inputs into a single output stream. But presence of a Funnel Stage reduces the performance of a job. It would increase the time taken by job by 30% (observations). When a Funnel Stage is to be used in a large job it is better to isolate itself to one job. Write the output to Datasets and funnel them in new job. Funnel Stage should be run in “continuous” mode, without hindrance.

Overall Job Design :
While designing DataStage Jobs care should be taken that a single job is not overloaded with Stages. Each extra Stage put in a Job corresponds to lesser number of resources available for every Stage, which directly affects the Jobs Performance. If possible, big jobs having large number of Stages should be logically split into smaller units. Also if a particular Stage has been identified to be taking lot of time in a job, like a transformer Stage having complex functionality with a lot of Stage variables and

transformations, then the design of jobs could be done in such a way that this Stage is put in a separate job all together (more resources for the transformer Stage!!!). Also while designing jobs, care must be taken that unnecessary column propagation is not done. Columns, which are not needed in the job flow, should not be propagated from one Stage to another and from one job to the next. As far as possible, RCP (Runtime Column Propagation) should be disabled in the jobs. Sorting in a job should be taken care try to minimise number sorts in a job. Design a job in such a way as to combine operations around same sort keys, if possible maintain same hash keys. Most often neglected option is "don't sort if previously sorted" in sort Stage, set this option to "true". This improves the Sort Stage performance a great deal. In Transformer Stage "Preserve Sort Order" can be used to maintain sort order of the data and reduce sorting in the job.In a transformer minimum of Stage variables should be used. More the no of Stage variable lower is the performance. An overloaded transformer can choke the data flow and lead to bad performance or even failure of job at some point. In order to minimise the load on transformer we can Avoid some unnecessary function calls. For example to convert a varchar field with date value can be type cast into Date type by simple formatting the input value. We need not use StringToDate function, which is used to convert a String to Date type. Implicit conversion of data types.
Reduce the number of Stage variables used. It was observed in our previous project by removing 5 Stage variables and 6 function calls, runtime for the job was reduced from 2 hours to 1 hour 10 min (approximately) with 100 million records input.Try to balance load on transformers by sharing the transformations across existing transformers. This would ensure smooth flow of data.If you require type casting, renaming of columns or addition of new columns, use Copy or Modify Stages to achieve thisWhenever you have to use Lookups on large tables, look at the options such as unloading the lookup tables to datasets and using, user defined join SQL to reduce the look up volume with the help of temp tables, etc.The Copy stage should be used instead of a Transformer for simple operations including:o Job Design placeholder between stages o Renaming Columnso Dropping Columnso Implicit (default) Type Conversions The "upsert" works well if the data is sorted on the primary key column of the table which is being loaded. Or Determine , if the record already exists or not to have "Insert" and "Update" separately.It is sometimes possible to re-arrange the order of business logic within a job flow to leverage the same sort order, partitioning, and groupings. Don't read from a Sequential File using SAME partitioning. Unless more than one source file is specified, this scenario will read the entire file into a single partition, making the entire downstream flow run sequentially (unless it is repartitioned)

Datastage Errors and Resolution

You may get many errors in datastage while compiling the jobs or running the jobs.

Some of the errors are as follows

a)Source file not found.
If you are trying to read the file, which was not there with that name.

b)Some times you may get Fatal Errors.

c) Data type mismatches.
This will occur when data type mismaches occurs in the jobs.

d) Field Size errors.

e) Meta data Mismach

f) Data type size between source and target different

g) Column Mismatch

i) Pricess time out.
If server is busy. This error will come some time.

Some of the errors in detail:

ds_Trailer_Rec: When checking operator: When binding output schema variable "outRec": When binding output interface field "TrailerDetailRecCount" to field "TrailerDetailRecCount": Implicit conversion from source type "ustring" to result type "string[max=255]": Possible truncation of variable length ustring when converting to string using codepage ISO-8859-1.

Solution:I resolved changing the extended col under meta data of the transformer to unicode

When checking operator: A sequential operator cannot preserve the partitioning
of the parallel data set on input port 0.

Solution:I resolved by changing the preserve partioning to 'clear' under transformer properties

Syntax error: Error in "group" operator: Error in output redirection: Error in output parameters: Error in modify adapter: Error in binding: Could not find type: "subrec", line 35

Solution:Its the issue of level number of those columns which were being added in transformer. Their level number was blank and the columns that were being taken from cff file had it as 02. Added the level number and job worked.

Out_Trailer: When checking operator: When binding output schema variable "outRec": When binding output interface field "STDCA_TRLR_REC_CNT" to field "STDCA_TRLR_REC_CNT": Implicit conversion from source type "dfloat" to result type "decimal[10,0]": Possible range/precision limitation.

CE_Trailer: When checking operator: When binding output interface field "Data" to field "Data": Implicit conversion from source type "string" to result type "string[max=500]": Possible truncation of variable length string.

Implicit conversion from source type "dfloat" to result type "decimal[10,0]": Possible range/precision limitation.

Solution: Used to transformer function'DFloatToDecimal'. As target field is Decimal. By default the output from aggregator output is double, getting the above by using above function able to resolve the warning.

When binding output schema variable "outputData": When binding output interface field "RecordCount" to field "RecordCount": Implicit conversion from source type "string[max=255]" to result type "int16": Converting string to number.

Problem(Abstract)
Jobs that process a large amount of data in a column can abort with this error:
the record is too big to fit in a block; the length requested is: xxxx, the max block length is: xxxx.

Resolving the problem

To fix this error you need to increase the block size to accommodate the record size:

1. Log into Designer and open the job.
2. Open the job properties--> parameters-->add environment variable and select:
APT_DEFAULT_TRANSPORT_BLOCK_SIZE
3. You can set this up to 256MB but you really shouldn't need to go over 1MB.
NOTE: value is in KB

For example to set the value to 1MB:
APT_DEFAULT_TRANSPORT_BLOCK_SIZE=1048576

The default for this value is 128kb.

When setting APT_DEFAULT_TRANSPORT_BLOCK_SIZE you want to use the smallest possible value since this value will be used for all links in the job.

For example if your job fails with APT_DEFAULT_TRANSPORT_BLOCK_SIZE set to 1 MB and succeeds at 4 MB you would want to do further testing to see what it the smallest value between 1 MB and 4 MB that will allow the job to run and use that value. Using 4 MB could cause the job to use more memory than needed since all the links would use a 4 MB transport block size.

NOTE: If this error appears for a dataset use APT_PHYSICAL_DATASET_BLOCK_SIZE.

. While connecting “Remote Desktop”, Terminal server has been exceeded maximum number of allowed connections

SOL: In Command Prompt, type mstsc /v: ip address of server /admin

OR mstsc /v: ip address /console

2. SQL20521N. Error occurred processing a conditional compilation directive near string. Reason code=rc. Following link has issue description:

http://pic.dhe.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=%2Fcom.ibm.db2.luw.messages.sql.doc%2Fdoc%2Fmsql20521n.html

3. SK_RETAILER_GROUP_BRDIGE,1: runLocally() did not reach EOF on its input data set 0.

SOL: Warning will be disappeared by regenerating SK File.

4. While connecting to Datastage client, there is no response, and while restarting websphere services, following errors occurred

```
[root@poluloro01 bin]# ./stopServer.sh  server1 -user wasadmin -password Wasadmin0708

ADMU0116I: Tool information is being logged in file

         /opt/ibm/WebSphere/AppServer/profiles/default/logs/server1/stopServer.log

ADMU0128I: Starting tool with the default profile

ADMU3100I: Reading configuration for server: server1

ADMU0111E: Program exiting with error: javax.management.JMRuntimeException:

          ADMN0022E: Access is denied for the stop operation on Server MBean

          because of insufficient or empty credentials.

ADMU4113E: Verify that username and password information is on the command line

          (-username and -password) or in the <conntype>.client.props file.

ADMU1211I: To obtain a full trace of the failure, use the -trace option.

ADMU0211I: Error details may be seen in the file:

         /opt/ibm/WebSphere/AppServer/profiles/default/logs/server1/stopServer.log
```

SOL: Wasadmin and XMeta passwords needs to be reset and commands are below..

```
          [root@poluloro01 bin]# cd /opt/ibm/InformationServer/ASBServer/bin/

[root@poluloro01 bin]# ./AppServerAdmin.sh -was -user wasadmin

-password Wasadmin0708

Info WAS instance /Node:poluloro01/Server:server1/ updated with new user information

Info MetadataServer daemon script updated with new user information

[root@poluloro01 bin]# ./AppServerAdmin.sh -was -user xmeta -password Xmeta0708

Info WAS instance /Node:poluloro01/Server:server1/ updated with new user information

Info MetadataServer daemon script updated with new user information
```

5. “The specified field doesn’t exist in view adapted schema”

SOL: Most of the time "The specified field: XXXXXX does not exist in the view adapted schema" occurred when we missed a field to map. Every stage has got an output tab if used in the between of the job. Make sure you have mapped every single field required for the next stage.

Sometime even after mapping the fields this error can be occurred and one of the reason could be that the view adapter has not linked the input and output fields. Hence in this case the required field mapping should be dropped and recreated.

Just to give an insight on this, the view adapter is an operator which is responsible for mapping the input and output fields. Hence DataStage creates an instance of APT_ViewAdapter which translate the components of the operator input interface schema to matching components of the interface schema. So if the interface schema is not having the same columns as operator input interface schema then this error will be reported.

1)When we use same partitioning in datastage transformer stage we get the following warning in 7.5.2 version.

TFCP000043   2   3   input_tfm: Input dataset 0 has a partitioning method other than entire specified; disabling memory sharing.

This is known issue and you can safely demote that warning into informational by adding this warning to Project specific message handler.

2) Warning: A sequential operator cannot preserve the partitioning of input data set on input port 0

Resolution: Clear the preserve partition flag before Sequential file stages.

3)DataStage parallel job fails with fork() failed, Resource temporarily unavailable

On aix execute following command to check maxuproc setting and increase it if you plan to run multiple jobs at the same time.

```
lsattr -E -l sys0 | grep maxuproc
maxuproc        1024            Maximum number of PROCESSES allowed per user      True
```

4)TFIP000000   3   Agg_stg: When checking operator: When binding input interface field “CUST_ACT_NBR” to field “CUST_ACT_NBR”: Implicit conversion from source type “string[5]” to result type “dfloat”: Converting string to number.

Resolution: use the Modify stage explicitly convert the data type before sending to aggregator stage.

5)Warning: A user defined sort operator does not satisfy the requirements.

Resolution:check the order of sorting columns and make sure use the same order when use join stage after sort to joing two inputs.

6)TFTM000000   2   3   Stg_tfm_header,1: Conversion error calling conversion routine timestamp_from_string data may have been lost

TFTM000000   1   xfmJournals,1: Conversion error calling conversion routine decimal_from_string data may have been lost

Resolution:check for the correct date format or decimal format and also null values in the date or decimal fields before passing to datastage StringToDate, DateToString,DecimalToString or StringToDecimal functions.

7)TOSO000119   2   3   Join_sort: When checking operator: Data claims to already be sorted on the specified keys the ‘sorted’ option can be used to confirm this. Data will be resorted as necessary. Performance may improve if this sort is removed from the flow

Resolution: Sort the data before sending to join stage and check for the order of sorting keys and join keys and make sure both are in the same order.

8)TFOR000000   2   1   Join_Outer: When checking operator: Dropping component “CUST_NBR” because of a prior component with the same name.

Resolution:If you are using join,diff,merge or comp stages make sure both links have the differnt column names other than key columns

9)TFIP000022 1 oci_oracle_source: When checking operator: When binding output interface field “MEMBER_NAME” to field “MEMBER_NAME”: Converting a nullable source to a non-nullable result;

Resolution:If you are reading from oracle database or in any processing stage where incoming column is defined as nullable and if you define metadata in datastage as non-nullable then you will get above issue.if you want to convert a nullable field to non nullable make sure you apply available null functions in datastage or in the extract query.

DATASTAGE COMMON ERRORS/WARNINGS AND SOLUTIONS – 2

1. No jobs or logs showing in IBM DataStage Director Client, however jobs are still accessible from the Designer Client.

SOL: SyncProject cmd that is installed with DataStage 8.5 can be run to analyze and recover projects

SyncProject -ISFile islogin -project dstage3 dstage5 –Fix

2. CASHOUT_DTL: Invalid property value /Connection/Database (CC_StringProperty::getValue, file CC_StringProperty.cpp, line 104)

SOL: Change the Data Connection properties manually in the produced

DB2 Connector stage.

A patch fix is available for this issue JR35643

3. Import .dsx file from command line

SOL: DSXImportService -ISFile dataconnection –DSProject dstage –DSXFile c:\export\oldproject.dsx

4. Generate Surrogate Key without Surrogate Key Stage

SOL: @PARTITIONNUM + (@NUMPARTITIONS * (@INROWNUM – 1)) + 1

Use above Formula in Transformer stage to generate a surrogate key.

5. Failed to authenticate the current user against the selected Domain: Could not connect to server.

RC: Client has invalid entry in host file

Server listening port might be blocked by a firewall

Server is down

SOL: Update the host file on client system so that the server hostname can be resolved from client.

Make sure the WebSphere TCP/IP ports are opened by the firewall.

Make sure the WebSphere application server is running. (OR)

Restart Websphere services.

6. The connection was refused or the RPC daemon is not running (81016)

RC: The dsprcd process must be running in order to be able to login to DataStage.

If you restart DataStage, but the socket used by the dsrpcd (default is 31538) was busy, the dsrpcd will fail to start. The socket may be held by dsapi_slave processes that were still running or recently killed when DataStage was restarted.

SOL: Run “ps -ef | grep dsrpcd” to confirm the dsrpcd process is not running.

Run “ps -ef | grep dsapi_slave” to check if any dsapi_slave processes exist. If so, kill them.

Run “netstat -a | grep dsprc” to see if any processes have sockets that are ESTABLISHED, FIN_WAIT, or CLOSE_WAIT. These will prevent the dsprcd from starting. The sockets with status FIN_WAIT or CLOSE_WAIT will eventually time out and disappear, allowing you to restart DataStage.

Then Restart DSEngine.   (if above doesn’t work) Needs to reboot the system.

7. To save Datastage logs in notepad or readable format

SOL: a) /opt/ibm/InformationServer/server/DSEngine (go to this directory)

./bin/dsjob -logdetail project_name job_name >/home/dsadm/log.txt

b) In director client, Project tab à Print à select print to file option and save it in local directory.

8. “Run time error ’457′. This Key is already associated with an element of this collection.”

SOL: Needs to rebuild repository objects.

a) Login to the Administrator client

b) Select the project

c) Click on Command

d) Issue the command ds.tools

e) Select option ‘2’

f) Keep clicking next until it finishes.

g) All objects will be updated.

9. To stop the datastage jobs in linux level

SOL: ps –ef | grep dsadm

To Check process id and phantom jobs

Kill -9 process_id

10. To run datastage jobs from command line

SOL: cd /opt/ibm/InformationServer/server/DSEngine

./dsjob -server $server_nm   -user $user_nm   -password $pwd -run $project_nm $job_nm

11. Failed to connect to JobMonApp on port 13401.

SOL: needs to restart jobmoninit script (in /opt/ibm/InformationServer/Server/PXEngine/Java)

Type   sh jobmoninit start $APT_ORCHHOME

Add 127.0.0.1 local host in /etc/hosts file

(Without local entry, Job monitor will be unable to use the ports correctly)

12. SQL0752N. Connect to a database is not permitted within logical unit of work CONNECT type 1 settings is in use.

SOL: COMMIT or ROLLBACK statement before requesting connection to another database.

1. While running ./NodeAgents.sh start command… getting the following error: “LoggingAgent.sh process stopped unexpectedly”

SOL: needs to kill LoggingAgentSocketImpl

Ps –ef | grep LoggingAgentSocketImpl (OR)

PS –ef | grep Agent (to check the process id of the above)

2. Warning: A sequential operator cannot preserve the partitioning of input data set on input port 0

SOL: Clear the preserve partition flag before Sequential file stages.

3. Warning: A user defined sort operator does not satisfy the requirements.

SOL: Check the order of sorting columns and make sure use the same order when use join stage after sort to joing two inputs.

4. Conversion error calling conversion routine timestamp_from_string data may have been lost. xfmJournals,1: Conversion error calling conversion routine decimal_from_string data may have been lost

SOL: check for the correct date format or decimal format and also null values in the date or decimal fields before passing to datastage StringToDate, DateToString,DecimalToString or StringToDecimal functions.

5. To display all the jobs in command line

SOL:

cd /opt/ibm/InformationServer/Server/DSEngine/bin

./dsjob -ljobs <project_name>

6. "Error trying to query dsadm[]. There might be an issue in database server"

SOL: Check XMETA connectivity.

db2 connect to xmeta (A connection to or activation of database "xmeta" cannot be made because of BACKUP pending)

7. "DSR_ADMIN: Unable to find the new project location"

SOL: Template.ini file might be missing in /opt/ibm/InformationServer/Server.

Copy the file from another severs.

8. "Designer LOCKS UP while trying to open any stage"

SOL: Double click on the stage that locks up datastage

Press ALT+SPACE

Windows menu will popup and select Restore

It will show your properties window now

Click on "X" to close this window.

Now, double click again and try whether properties window appears.

9. "Error Setting up internal communications (fifo RT_SCTEMP/job_name.fifo)

SOL: Remove the locks and try to run (OR)

Restart DSEngine and try to run (OR)

Go to /opt/ibm/InformationServer/server/Projects/proj_name/

ls RT_SCT* then

rm –f RT_SCTEMP

then try to restart it.

10. While attempting to compile job, "failed to invoke GenRunTime using Phantom process helper"

RC: /tmp space might be full

Job status is incorrect

Format problems with projects uvodbc.config file

SOL: a) clean up /tmp directory

b) DS Director à JOB à clear status file

c) confirm uvodbc.config has the following entry/format:

```
[ODBC SOURCES]
<local uv>
DBMSTYPE = UNIVERSE
Network  = TCP/IP
Service =  uvserver
Host = 127.0.0.1
```

ERROR:Phantom error in jobs

Resolution – Datastage Services have to be started

So follow the following steps.

Login to server through putty using dsadm user.

Check whether active or stale sessions are there.

ps –ef|grep slave

Ask the application team to close the active or stale sessions running from application's user.

If they have closed the sessions, but sessions are still there, then kill those sessions.

Make sure no jobs are running

If any, ask the application team to stop the job

ps –ef|grep dsd.run

Check for output for below command before stopping Datastage services.

netstat –a|grep dsrpc

If any processes are in established, check any job or stale or active or osh sessions are not running.

If any processes are in close_wait, then wait for some time, those processes

will not be visible.

Stop the Datastage services.

cd $DSHOME

./dsenv

cd $DSHOME/bin

./uv –admin –stop

Check whether Datastage services are stopped.

netstat –a|grep dsrpc

No output should come for above command.

Wait for 10 to 15 min for shared memory to be released by process holding them.

Start the Datastage services.

./uv –admin –start

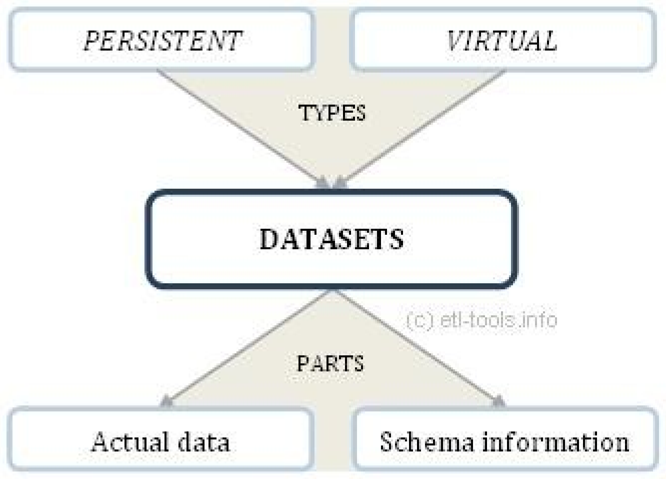
If asking for dsadm password while firing the command , then enable impersonation.through root user

${DSHOME}/scripts/DSEnable_impersonation.sh

Datasets

Managing and operating on large data volumes is usually an extremely complex process. To facilitate that, Orchestrate might extend relational database management system support in Datastage by implementing Datasets.
By and large, datasets might be interpreted as uniform sets of rows within the internal representation of Framework. Commonly, the two types of datasets might be distinguished:

Types of Datasets in Datastage:



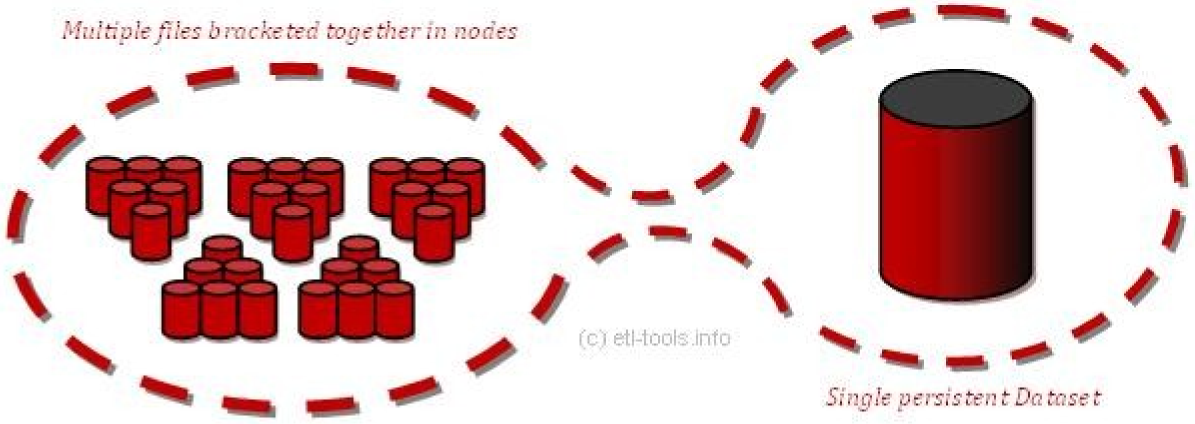
According to the scheme above, there are the two groups of Datasets - **persistent and virtual**.
The first type, persistent Datasets are marked with ***.ds** extensions, while for second type, virtual datasets ***.v** extension is reserved. (It's important to mention, that no *.v files might be visible in the Unix file system, as long as they exist only virtually, while inhabiting RAM memory. Extesion *.v itself is characteristic strictly for OSH - the Orchestrate language of scripting).
Further differences are much more significant. Primarily, **persistent Datasets are being stored in Unix files using internal Datastage EE format**, while **virtual Datasets are never stored on disk** - they do exist within links, and in EE format, but in RAM memory. Finally, persistent **Datasets are readable and rewriteable with the DataSet Stage**, and **virtual Datasets - might be passed through in memory**.

Accurateness demands mentioning the third type of Datasets - they're called **filesets**, while their storage places are diverse Unix files and they're human-readable. Filesets are generally marked with the ***.fs** extension.

There are a few features specific for Datasets that have to be mentioned to complete the Datasets policy. Firstly, as a single Dataset contains multiple records, it is obvious that all of them must undergo the same processes and modifications. In a word, all of them must go through the same successive stage.
Secondly, it should be expected that different Datasets usually have different schemas, therefore they cannot be treated commonly.

More accurately, typically Orchestrate Datasets are a single-file equivalent for the whole sequence of records. With datasets, they might be shown as a one object. Datasets themselves help ignoring the fact that demanded data really are compounded of multiple and diverse files remaining spread across different sectors of processors and disks of parallel computers. Along that, complexity of programming might be significantly reduced, as shown in the example below.

Datasets parallel architecture in Datastage:


Multiple files bracketed together in nodes
(c) etl-tools.info
Single persistent Dataset

Primary multiple files - shown on the left side of the scheme - have been bracketed together, what resulted in five nodes. While using datasets, all the files, all the nodes might be boiled down to the only one, single Dataset - shown on the right side of the scheme. Thereupon, you might program only one file and get results on all the input files. That significantly shorten time needed for modifying the whole group of separate files, and reduce the possibility of engendering accidental errors. What are its measurable profits? Mainly, significantly increasing speed of applications basing on large data volumes.

All in all, is it worth-while? Surely. Especially, while talking about technological advance and succeeding data-dependence. They do coerce using larger and larger volumes of data, and - as a consequence - systems able to rapid cooperate on them became a necessity.

Datastage Scenarios and solutions

Field mapping using Transformer stage:

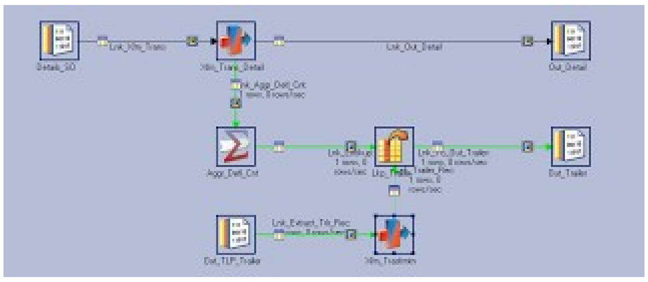
Requirement:
field will be right justified zero filled, Take last 18 characters

Solution:
Right("0000000000":Trim(Lnk_Xfm_Trans.link),18)

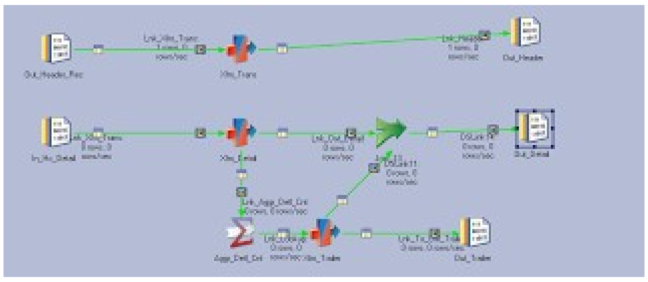
Scenario 1:

We have two datasets with 4 cols each with different names. We should create a dataset with 4 cols 3 from one dataset and one col with the record count of one dataset.

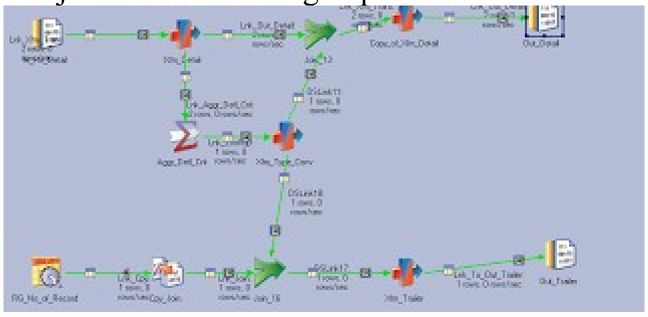
We can use aggregator with a dummy column and get the count from one dataset and do a look up from other dataset and map it to the 3 rd dataset
Something similar to the below design:



Scenario 2:
Following is the existing job design. But requirement got changed to: Head and trailer datasets should populate even if detail records is not present in the source file. Below job don't do that job.



Hence changed the above job to this following requirement:



Used row generator with a copy stage. Given default value(zero) for col(count) coming in from row generator. If no detail records it will pick the record count from row generator.

We have a source which is a sequential file with header and footer. How to remove the header and footer while reading this file using sequential file stage of Datastage?
Sol:Type command in putty: sed '1d;$d' file_name>new_file_name (type this in job before job subroutine then use new file in seq stage)

IF I HAVE SOURCE LIKE COL1 A A B AND TARGET LIKE COL1 COL2 A 1 A 2 B1. HOW TO ACHIEVE THIS OUTPUT USING STAGE VARIABLE IN TRANSFORMER STAGE?

If keyChange =1 Then 1 Else stagevaraible+1

Suppose that 4 job control by the sequencer like (job 1, job 2, job 3, job 4)if job 1 have 10,000 row ,after run the job only 5000 data has been loaded in target table remaining are not loaded and your job going to be aborted then.. How can short out the problem.Suppose job

sequencer synchronies or control 4 job but job 1 have problem, in this condition should go director and check it what type of problem showing either data type problem, warning massage, job fail or job aborted, If job fail means data type problem or missing column action .So u should go Run window ->Click-> Tracing->Performance or In your target table ->general -> action-> select this option here two option
(i) On Fail -- commit , Continue
(ii) On Skip -- Commit, Continue.
First u check how many data already load after then select on skip option then continue and what remaining position data not loaded then select On Fail , Continue ...... Again Run the job defiantly u get successful massage
Question: I want to process 3 files in sequentially one by one how can i do that. while processing the files it should fetch files automatically .
Ans:If the metadata for all the files r same then create a job having file name as parameter then use same job in routine and call the job with different file name...or u can create sequencer to use the job..

Parameterize the file name.
Build the job using that parameter
Build job sequencer which will call this job and will accept the parameter for file name.
Write a UNIX shell script which will call the job sequencer three times by passing different file each time.
RE: What Happens if RCP is disable ?

In such case Osh has to perform Import and export every time when the job runs and the processing time job is also increased...
Runtime column propagation (RCP): If RCP is enabled for any job and specifically for those stages whose output connects to the shared container input then meta data will be propagated at run time so there is no need to map it at design time.
If RCP is disabled for the job in such case OSH has to perform Import and export every time when the job runs and the processing time job is also increased.
Then you have to manually enter all the column description in each stage.RCP- Runtime column propagation

Question:
Source: Target

Eno	Ename	Eno	Ename
1	a,b	1	a
2	c,d	2	b
3	e,f	3	c

source has 2 fields like

COMPANY	LOCATION
IBM	HYD
TCS	BAN
IBM	CHE
HCL	HYD

```
TCS              CHE
IBM              BAN
HCL              BAN
HCL              CHE
```

LIKE THIS.......

THEN THE OUTPUT LOOKS LIKE THIS....

```
Company loc count

TCS  HYD  3
     BAN
     CHE
IBM  HYD  3
     BAN
     CHE
HCL  HYD  3
     BAN
     CHE
```

2)input is like this:

```
no,char
1,a
2,b
3,a
4,b
5,a
6,a
7,b
8,a
```

But the output is in this form with row numbering of Duplicate occurence

output:

```
no,char,Count
"1","a","1"
"6","a","2"
"5","a","3"
"8","a","4"
"3","a","5"
"2","b","1"
"7","b","2"
"4","b","3"
```

3)Input is like this:

```
file1
10
```

```
20
10
10
20
30
```

Output is like:

```
file2          file3(duplicates)
10             10
20             10
30             20
```

4)Input is like:

```
file1
10
20
10
10
20
30
```

Output is like Multiple occurrences in one file and single occurrences in one file:

```
file2    file3
10       30
10
10
20
20
```

5)Input is like this:

```
file1
10
20
10
10
20
30
```

Output is like:

```
file2          file3
10             30
20
```

6)Input is like this:

```
file1
1
2
3
4
5
6
7
```

8
9
10

Output is like:

file2(odd)	file3(even)
1	2
3	4
5	6
7	8
9	10

7)How to calculate Sum(sal), Avg(sal), Min(sal), Max(sal) with out using Aggregator stage..

8)How to find out First sal, Last sal in each dept with out using aggregator stage

9)How many ways are there to perform remove duplicates function with out using Remove duplicate stage..

Scenario:

source has 2 fields like

COMPANY	LOCATION
IBM	HYD
TCS	BAN
IBM	CHE
HCL	HYD
TCS	CHE
IBM	BAN
HCL	BAN
HCL	CHE

LIKE THIS.......

THEN THE OUTPUT LOOKS LIKE THIS....

```
Company loc count

TCS  HYD  3
       BAN
       CHE
IBM  HYD  3
       BAN
       CHE
HCL  HYD  3
       BAN
       CHE
```

Solution:

Seqfile......>Sort......>Trans......>RemoveDuplicates..........Dataset

```
Sort                                    Trans:
Key=Company                               create stage variable as Company1
Sort order=Asc              Company1=If(in.keychange=1) then in.Location Else
Company1:',':in.Location
create keykchange=True                         Drag and Drop in derivation
                                               Company     ...................Company
                                              Company1.......................Location
```

RemoveDup:
Key=Company
Duplicates To Retain=Last

11)The input is
Shirt|red|blue|green
Pant|pink|red|blue

Output should be,

Shirt:red
Shirt:blue
Shirt:green
pant:pink
pant:red
pant:blue

Solution:
it is reverse to pivote stage
use
    seq------sort------tr----rd-----tr----tg
in the sort stage use create key change column is true
in trans create stage variable=if colu=1 then key c.value else key v::colum
rd stage use duplicates retain last
tran stage use field function superate columns

similar **Scenario:** :
source
col1 col3
1 samsung
1 nokia
1 ercisson
2 iphone
2 motrolla
3 lava
3 blackberry
3 reliance

Expected Output

```
col 1 col2       col3   col4
1     samsung nokia ercission
2      iphone   motrolla
3       lava     blackberry reliance
```

You can get it by using Sort stage --- Transformer stage --- RemoveDuplicates --- Transformer --tgt

Ok

First Read and Load the data into your source file(For Example Sequential File)

And in Sort stage select key change column = True (To Generate Group ids)

Go to Transformer stage

Create one stage variable.

You can do this by right click in stage variable go to properties and name it as your wish (For example temp)

and in expression write as below

if keychange column =1 then column name else temp:',':column name

This column name is the one you want in the required column with delimited commas.

On remove duplicates stage key is col1 and set option duplicates retain to--> Last.
in transformer drop col3 and define 3 columns like col2,col3,col4
in col1 derivation give Field(InputColumn,",",1) and
in col1 derivation give Field(InputColumn,",",2) and
in col1 derivation give Field(InputColumn,",",3)

Scenario:

12)Consider the following employees data as source?

```
employee_id, salary
-------------------
10,         1000
20,         2000
30,         3000
40,         5000
```

Create a job to find the sum of salaries of all employees and this sum should repeat for all the rows.

The output should look like as

```
employee_id, salary, salary_sum
-------------------------------
10,         1000,   11000
20,         2000,   11000
30,         3000,   11000
40,         5000,   11000
```

Scenario:

I have two source tables/files numbered 1 and 2.
In the the target, there are three output tables/files, numbered 3,4 and 5.

The scenario is that,

to the out put 4 -> the records which are common to both 1 and 2 should go.

to the output 3 -> the records which are only in 1 but not in 2 should go

to the output 5 -> the records which are only in 2 but not in 1 should go.

```
sltn:src1----->copy1------>---------------------------------->output_1(only left table)
                              Join(inner type)----> ouput_1
src2----->copy2------>---------------------------------->output_3(only right table)
```

Consider the following employees data as source?

```
employee_id, salary
-------------------
10,        1000
20,        2000
30,        3000
40,        5000
```

Scenario:

Create a job to find the sum of salaries of all employees and this sum should repeat for all the rows.

The output should look like as

```
employee_id, salary, salary_sum
-------------------------------
10,        1000,  11000
20,        2000,  11000
30,        3000,  11000
40,        5000,  11000
```

sltn:

Take Source --->Transformer(Add new Column on both the output links and assign a value as 1)------------------------>   1) Aggregator (Do group by using that new column)
                                    2)lookup/join(join on that new column)-------->tgt.

Scenario:

```
sno,sname,mark1,mark2,mark3
1,rajesh,70,68,79
2,mamatha,39,45,78
3,anjali,67,39,78
4,pavani,89,56,45
5,indu,56,67,78
```

out put is

```
sno,snmae,mark1,mark2,mark3,delimetercount
1,rajesh,70,68,79,4
2,mamatha,39,45,78,4
3,anjali,67,39,78,4
4,pavani,89,56,45,4
5,indu,56,67,78,4
```

seq--->trans--->seq

create one stage variable as **delimiter**..
and put derivation on stage as DSLink4.sno : "," : DSLink4.sname : "," : DSLink4.mark1 : "," :DSLink4.mark2 : "," : DSLink4.mark3

and do mapping and create one more column count as integer type.

and put derivation on count column as Count(**delimter**, ",")

scenario:

sname	total_vowels_count
Allen	2
Scott	1
Ward	1

Under Transformer Stage Description:

total_Vowels_Count=Count(DSLink3.last_name,"a")+Count(DSLink3.last_name,"e")+Count(DSLink3.last_name,"i")+Count(DSLink3.last_name,"o")+Count(DSLink3.last_name,"u").

Scenario:

1)On daily we r getting some huge files data so all files metadata is same we have to load in to target table how we can load?
Use File Pattern in sequential file

2) One column having 10 records at run time we have to send 5th and 6th record to target at run time how we can send?
Can get through,by using UNIX command in sequential file filter option

How can we get 18 months date data in transformer stage?
Use transformer stage after input seq file and try this one as constraint in transformer stage :

DaysSinceFromDate(CurrentDate(), DSLink3.date_18)<=548 OR
DaysSinceFromDate(CurrentDate(), DSLink3.date_18)<=546

where date_18 column is the column having that date which needs to be less or equal to 18 months and 548 is no. of days for 18 months and for leap year it is 546(these numbers you need to check).

What is differences between Force Compile and Compile ?

Diff b/w Compile and Validate?

Compile option only checks for all mandatory requirements like link requirements, stage options and all. But it will not check if the database connections are valid.
Validate is equivalent to Running a job except for extraction/loading of data. That is, validate option will test database connectivity by making connections to databases.

How to FInd Out Duplicate Values Using Transformer?

You can capture the duplicate records based on keys using Transformer stage variables.

1. Sort and partition the input data of the transformer on the key(s) which defines the duplicate.
2. Define two stage variables, let's say StgVarPrevKeyCol(data type same as KeyCol) and StgVarCntr as Integer with default value 0
where KeyCol is your input column which defines the duplicate.

Expression for StgVarCntr(1st stg var-- maintain order):

If DSLinknn.KeyCol = StgVarPrevKeyCol Then StgVarCntr + 1 Else 1

Expression for StgVarPrevKeyCol(2nd stg var):

DSLinknn.KeyCol

3. Now in constrain, if you filter rows where StgVarCntr = 1 will give you the unique records and if you filter StgVarCntr > 1 will give you duplicate records.

My source is Like
Sr_no, Name
10,a
10,b
20,c
30,d
30,e
40,f

My target Should Like:

Target 1:(Only unique means which records r only once)
20,c
40,f

Target 2:(Records which r having more than 1 time)
10,a
10,b
30,d
30,e

How to do this in DataStage....
****

use aggregator and transformer stages

source-->aggregator-->transformat-->target

perform count in aggregator, and take two op links in trasformer, filter data count>1 for one llink and put count=1 for second link.

Scenario:
in my i/p source i have N no.of records

In output i have 3 targets

i want o/p like 1st rec goes to 1st targt and

2nd rec goes to 2nd target and

3rd rec goes to 3rd target again

4th rec goes to 1st taget .... like this

do this ""without using partition techniques "" remember it.

****
source--->trans---->target
in trans use conditions on constraints
mod(empno,3)=1
mod(empno,3)=2
mod(empno,3)=0
Scenario:
im having i/p as
col A
a_b_c
x_F_I
DE_GH_IF

we hav to mak it as

col1 col 2 col3
a b c
x f i
de gh if

****

Transformer
create 3 columns with derivation
col1 Field(colA,'_',1)
col2 Field(colA,'_',2)
col3 Field(colA,'_',3)

****
Field function divides the column based on the delimeter,
if the data in the col is like A,B,C
then

Field(col,',',1) gives A
Field(col,',',2) gives B
Field(col,',',3) gives C

Unix Shell Scripting

Script for passwordless sftp unix:

```
cd $target_directory
sftp $sftp_user@$sftp_machine <<EOF
cd $source_directory
bin
get $source_file_name
exit
EOF
```

DataSet in DataStage

Inside a InfoSphere DataStage parallel job, data is moved around in data sets. These carry meta data with them, both column definitions and information about **the configuration** that was in effect when the data set was created. If for example, you have a stage which limits execution to a subset of available nodes, and the data set was created by a stage using all nodes, InfoSphere DataStage can detect that the data will need repartitioning.

If required, data sets can be landed as persistent data sets, represented by a Data Set stage .This is the most efficient way of moving data between linked jobs. Persistent data sets are stored in a series of files linked by a control file (note that you should not attempt to manipulate these files using UNIX tools such as RM or MV. Always use the tools provided with InfoSphere DataStage).
there are the two groups of Datasets - **persistent and virtual**.

The first type, persistent Datasets are marked with ***.ds** extensions, while for second type, virtual datasets ***.v** extension is reserved. (It's important to mention, that no *.v files might be visible in the Unix file system, as long as they exist only virtually, while inhabiting RAM memory. Extesion *.v itself is characteristic strictly for OSH - the Orchestrate language of scripting).

Further differences are much more significant. Primarily, **persistent Datasets are being stored in Unix files using internal Datastage EE format**, while **virtual Datasets are never stored on disk** - they do exist within links, and in EE format, but in RAM memory. Finally, persistent **Datasets are readable and rewriteable with the DataSet Stage**, and **virtual Datasets - might be passed through in memory**.

A data set comprises a descriptor file and a number of other files that are added as the data set grows. These files are stored on multiple disks in your system. A data set is organized in terms of **partitions** and **segments.**

Each partition of a data set is stored on a single processing node. Each data segment contains all the records written by a single job. So a segment can contain files from many partitions, and a partition has files from many segments.

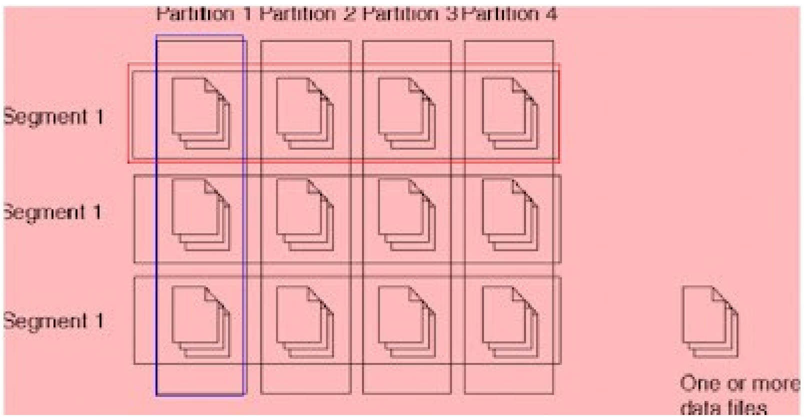
Firstly, as a single Dataset contains multiple records, it is obvious that all of them must undergo the same processes and modifications. In a word, all of them must go through the same successive stage.
Secondly, it should be expected that different Datasets usually have different schemas, therefore they cannot be treated commonly.

Alias names of Datasets are

1) Orchestrate File
2) Operating System file

And Dataset is multiple files. They are
a) Descriptor File
b) Data File
c) Control file
d) Header Files

In Descriptor File, we can see the Schema details and address of data.
In Data File, we can see the data in Native format.
And Control and Header files reside in Operating System.



Starting a Dataset Manager

Choose Tools ► Data Set Management, a Browse Files dialog box appears:

1. Navigate to the directory containing the data set you want to manage. By convention, data set files have the suffix .ds.

2. Select the data set you want to manage and click OK. The Data Set Viewer appears. From here you can copy or delete the chosen data set. You can also view its schema (column definitions) or the data it contains.

3. SCD Type 2

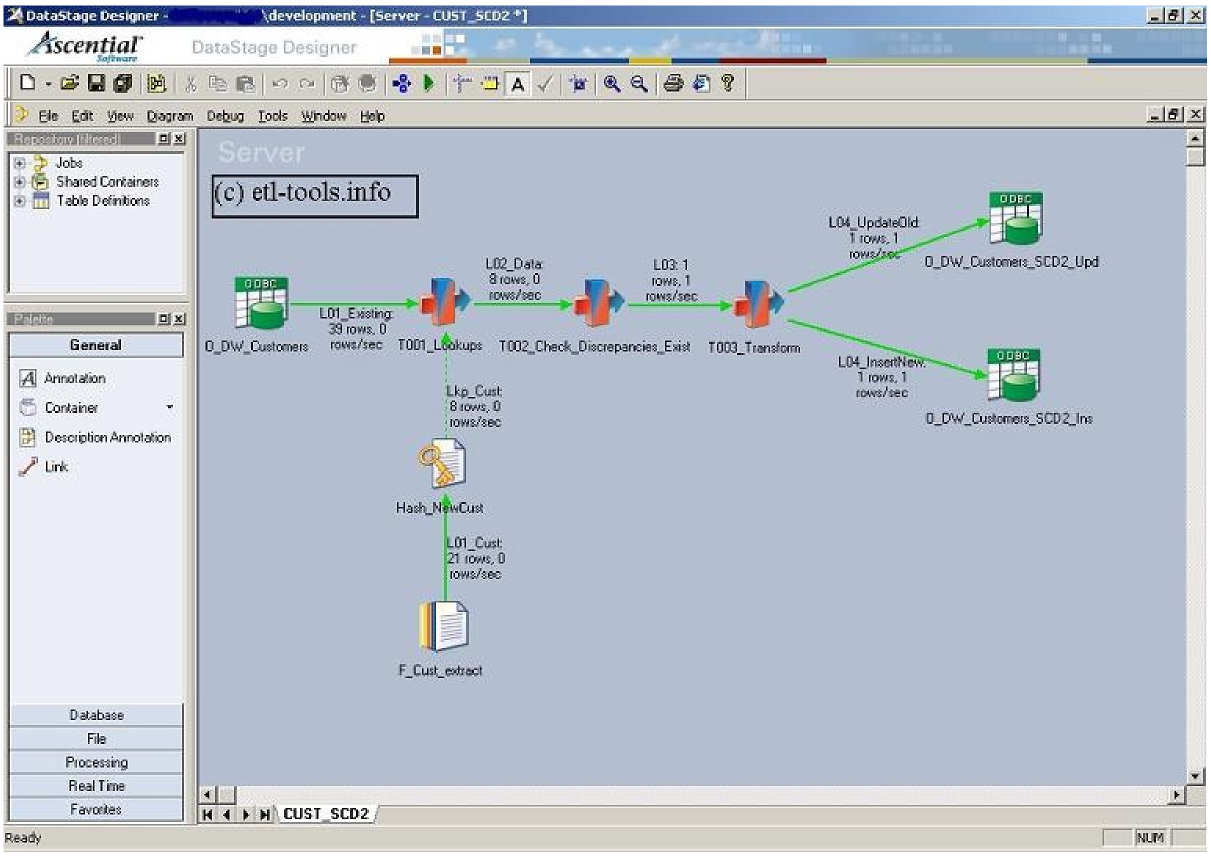
4. **Slowly changing dimension Type 2** is a model where the whole history is stored in the database. An additional dimension record is created and the segmenting between the old record values and the new (current) value is easy to extract and the history is clear.
   The fields 'effective date' and 'current indicator' are very often used in that dimension and the fact table usually stores dimension key and version number.

5. SCD 2 implementation in Datastage

6. The job described and depicted below shows how to implement SCD Type 2 in Datastage. It is one of many possible designs which can implement this dimension. For this example, we will use a table with customers data (it's name is D_CUSTOMER_SCD2) which has the following structure and data:

D_CUSTOMER dimension table before loading							
CUST_ID	**CUST_**	**CUST_**	**CUST_**	**CUST_**	**REC_**	**REC_**	**REC_**
	NAME	**GROUP_ID**	**TYPE_ID**	**COUNTRY_ID**	**VERSION**	**EFF DT**	**CURRENT_IND**
DRBOUA7	Dream Basket	EL	S	PL	1	01-10-2006	Y
ETIMAA5	**ETL tools info**	**BI**	**C**	**FI**	**1**	**29-09-2006**	**Y**
FAMMFA0	Fajatso	FD	S	CD	1	27-09-2006	Y
FICILA0	First Pactonic	FD	C	IT	1	25-09-2006	Y
FRDXXA2	Frasir	EL	C	SK	1	23-09-2006	Y
GAMOPA9	Ganpa LTD.	FD	C	US	1	21-09-2006	Y
GGMOPA9	GG electroni	EL	S	RU	1	19-09-	Y

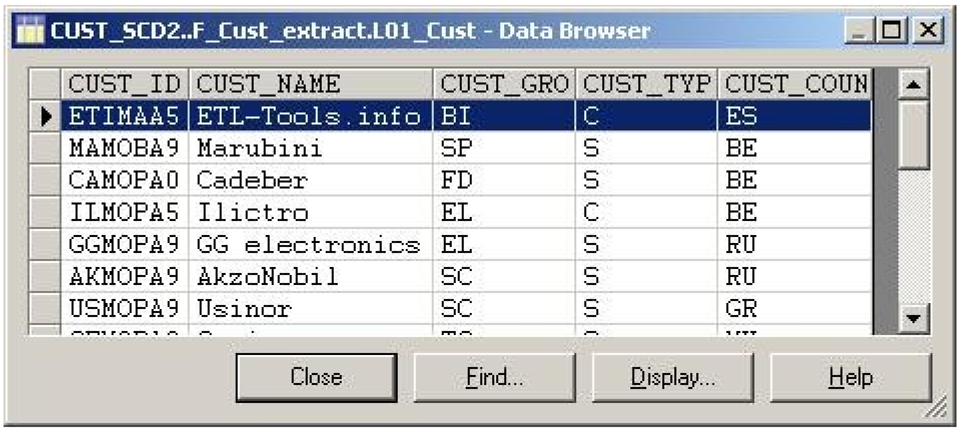
	cs					2006	
GLMFIA6	Glasithklini	FD	S	PL	1	17-09-2006	Y
GLMPEA9	Globitele co	TC	S	FI	1	15-09-2006	Y
GONDWA5	Goli Airlines	BN	S	GB	1	13-09-2006	Y



The most important facts and stages of the CUST_SCD2 job processing:

- The dimension table with customers is refreshed daily and one of the data sources is a text file. For the purpose of this example the CUST_ID=ETIMAA5 differs from the one stored in the database and it is the only record with changed data. It has the following structure and data:

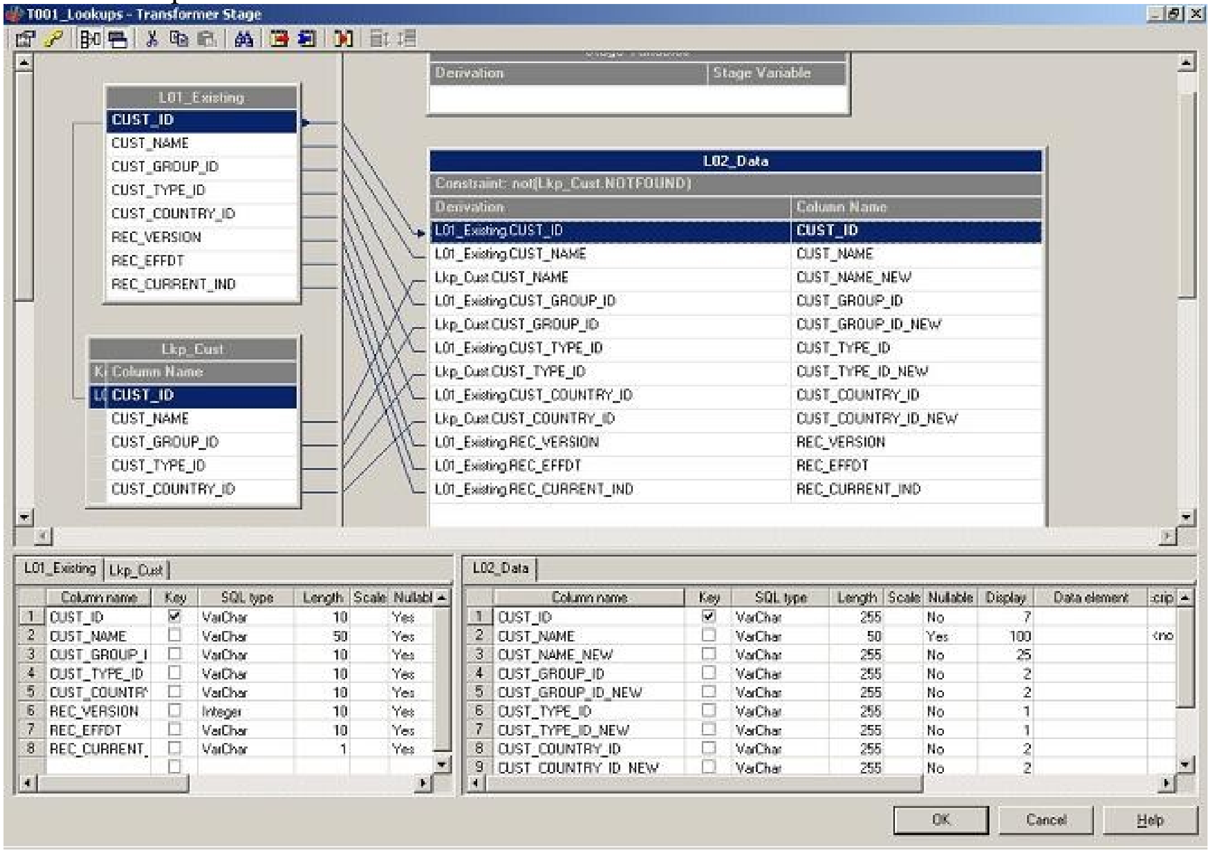
SCD 2 - Customers file extract:


CUST_SCD2..F_Cust_extract.L01_Cust - Data Browser

CUST_ID	CUST_NAME	CUST_GRO	CUST_TYP	CUST_COUN
ETIMAA5	ETL-Tools.info	BI	C	ES
MAMOBA9	Marubini	SP	S	BE
CAMOPA0	Cadeber	FD	S	BE
ILMOPA5	Ilictro	EL	C	BE
GGMOPA9	GG electronics	EL	S	RU
AKMOPA9	AkzoNobil	SC	S	RU
USMOPA9	Usinor	SC	S	GR

- There is a hashed file (Hash_NewCust) which handles a lookup of the new data coming from the text file.
- A T001_Lookups transformer does a lookup into a hashed file and maps new and old values to separate columns.

SCD 2 lookup transformer


T001_Lookups - Transformer Stage

L01_Existing: CUST_ID, CUST_NAME, CUST_GROUP_ID, CUST_TYPE_ID, CUST_COUNTRY_ID, REC_VERSION, REC_EFFDT, REC_CURRENT_IND

Lkp_Cust: CUST_ID, CUST_NAME, CUST_GROUP_ID, CUST_TYPE_ID, CUST_COUNTRY_ID

L02_Data

Constraint: not(Lkp_Cust.NOTFOUND)

Derivation	Column Name
L01_Existing.CUST_ID	CUST_ID
L01_Existing.CUST_NAME	CUST_NAME
Lkp_Cust.CUST_NAME	CUST_NAME_NEW
L01_Existing.CUST_GROUP_ID	CUST_GROUP_ID
Lkp_Cust.CUST_GROUP_ID	CUST_GROUP_ID_NEW
L01_Existing.CUST_TYPE_ID	CUST_TYPE_ID
Lkp_Cust.CUST_TYPE_ID	CUST_TYPE_ID_NEW
L01_Existing.CUST_COUNTRY_ID	CUST_COUNTRY_ID
Lkp_Cust.CUST_COUNTRY_ID	CUST_COUNTRY_ID_NEW
L01_Existing.REC_VERSION	REC_VERSION
L01_Existing.REC_EFFDT	REC_EFFDT
L01_Existing.REC_CURRENT_IND	REC_CURRENT_IND

L01_Existing | Lkp_Cust

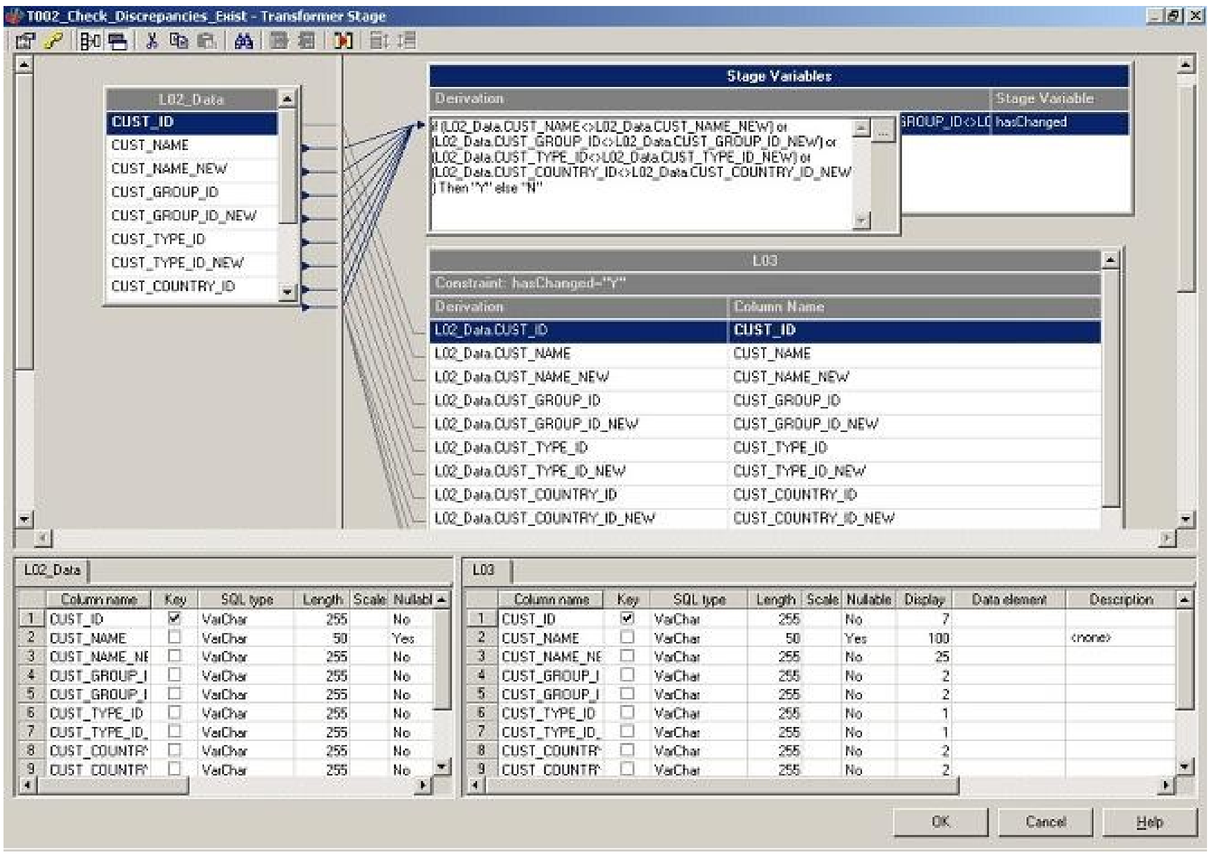
	Column name	Key	SQL type	Length	Scale	Nullabl
1	CUST_ID	✓	VarChar	10		Yes
2	CUST_NAME		VarChar	50		Yes
3	CUST_GROUP_I		VarChar	10		Yes
4	CUST_TYPE_ID		VarChar	10		Yes
5	CUST_COUNTR\`		VarChar	10		Yes
6	REC_VERSION		Integer	10		Yes
7	REC_EFFDT		VarChar	10		Yes
8	REC_CURRENT_		VarChar	1		Yes

L02_Data

	Column name	Key	SQL type	Length	Scale	Nullable	Display	Data element	:crip
1	CUST_ID	✓	VarChar	255		No	7		
2	CUST_NAME		VarChar	50		Yes	100		<no
3	CUST_NAME_NEW		VarChar	255		No	25		
4	CUST_GROUP_ID		VarChar	255		No	2		
5	CUST_GROUP_ID_NEW		VarChar	255		No	2		
6	CUST_TYPE_ID		VarChar	255		No	1		
7	CUST_TYPE_ID_NEW		VarChar	255		No	1		
8	CUST_COUNTRY_ID		VarChar	255		No	2		
9	CUST_COUNTRY_ID_NEW		VarChar	255		No	2		

A T002_Check_Discrepacies_exist transformer compares old and new values of records and passes through only records that differ.

SCD 2 check discrepancies transformer



T002_Check_Discrepancies_Exist - Transformer Stage

L02_Data
CUST_ID
CUST_NAME
CUST_NAME_NEW
CUST_GROUP_ID
CUST_GROUP_ID_NEW
CUST_TYPE_ID
CUST_TYPE_ID_NEW
CUST_COUNTRY_ID

Stage Variables

Derivation	Stage Variable
If (L02_Data.CUST_NAME<>L02_Data.CUST_NAME_NEW) or (L02_Data.CUST_GROUP_ID<>L02_Data.CUST_GROUP_ID_NEW) or (L02_Data.CUST_TYPE_ID<>L02_Data.CUST_TYPE_ID_NEW) or (L02_Data.CUST_COUNTRY_ID<>L02_Data.CUST_COUNTRY_ID_NEW) Then "Y" else "N"	hasChanged

L03

Constraint: hasChanged="Y"

Derivation	Column Name
L02_Data.CUST_ID	CUST_ID
L02_Data.CUST_NAME	CUST_NAME
L02_Data.CUST_NAME_NEW	CUST_NAME_NEW
L02_Data.CUST_GROUP_ID	CUST_GROUP_ID
L02_Data.CUST_GROUP_ID_NEW	CUST_GROUP_ID_NEW
L02_Data.CUST_TYPE_ID	CUST_TYPE_ID
L02_Data.CUST_TYPE_ID_NEW	CUST_TYPE_ID_NEW
L02_Data.CUST_COUNTRY_ID	CUST_COUNTRY_ID
L02_Data.CUST_COUNTRY_ID_NEW	CUST_COUNTRY_ID_NEW

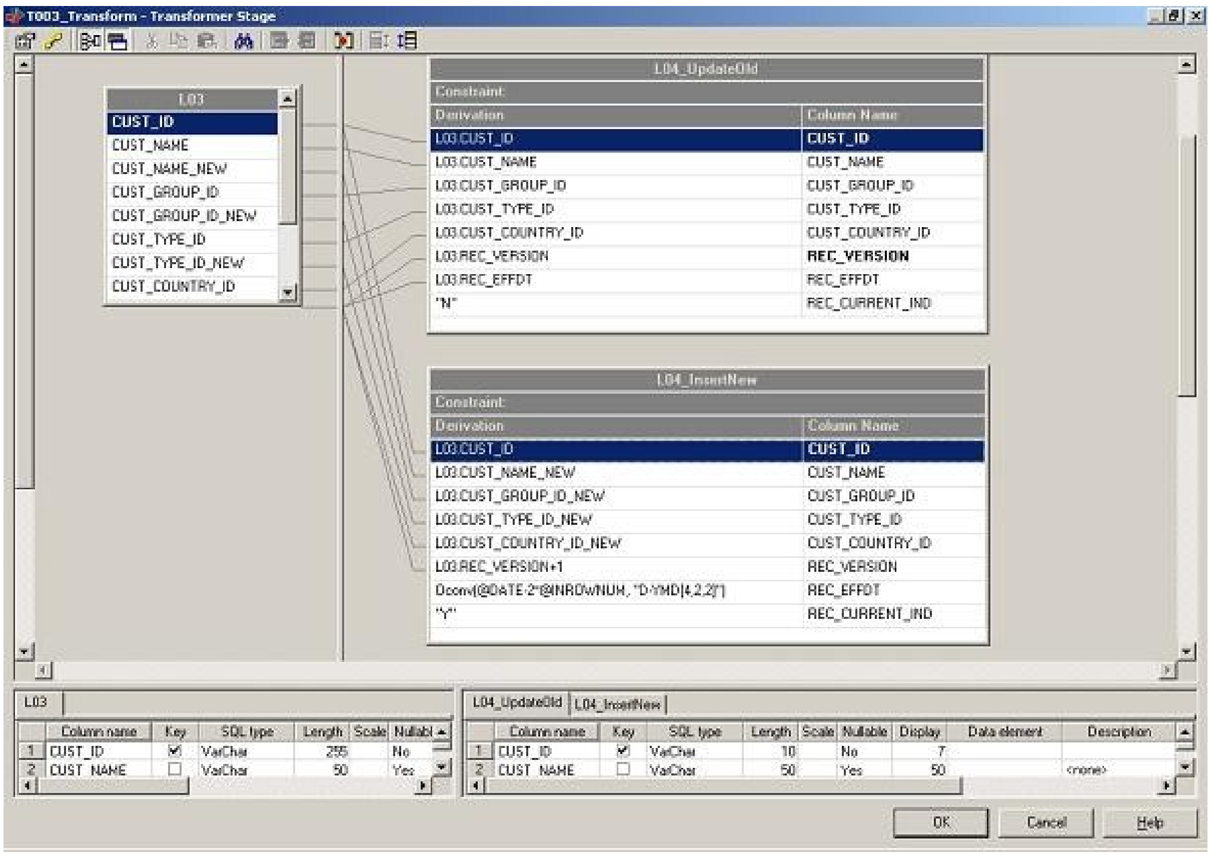
L02_Data

	Column name	Key	SQL type	Length	Scale	Nullabl
1	CUST_ID	✓	VarChar	255		No
2	CUST_NAME	☐	VarChar	50		Yes
3	CUST_NAME_NE	☐	VarChar	255		No
4	CUST_GROUP_I	☐	VarChar	255		No
5	CUST_GROUP_I	☐	VarChar	255		No
6	CUST_TYPE_ID	☐	VarChar	255		No
7	CUST_TYPE_ID_	☐	VarChar	255		No
8	CUST_COUNTR'	☐	VarChar	255		No
9	CUST_COUNTR'	☐	VarChar	255		No

L03

	Column name	Key	SQL type	Length	Scale	Nullable	Display	Data element	Description
1	CUST_ID	✓	VarChar	255		No	7		
2	CUST_NAME	☐	VarChar	50		Yes	100		<none>
3	CUST_NAME_NE	☐	VarChar	255		No	25		
4	CUST_GROUP_I	☐	VarChar	255		No	2		
5	CUST_GROUP_I	☐	VarChar	255		No	2		
6	CUST_TYPE_ID	☐	VarChar	255		No	1		
7	CUST_TYPE_ID_	☐	VarChar	255		No	1		
8	CUST_COUNTR'	☐	VarChar	255		No	2		
9	CUST_COUNTR'	☐	VarChar	255		No	2		

A T003 transformer handles the UPDATE and INSERT actions of a record. The old record is updated with current indictator flag set to no and the new record is inserted with current indictator flag set to yes, increased record version by 1 and the current date.
SCD 2 insert-update record transformer


T003_Transform - Transformer Stage

L03
CUST_ID
CUST_NAME
CUST_NAME_NEW
CUST_GROUP_ID
CUST_GROUP_ID_NEW
CUST_TYPE_ID
CUST_TYPE_ID_NEW
CUST_COUNTRY_ID

L04_UpdateOld

Constraint:

Derivation	Column Name
L03.CUST_ID	CUST_ID
L03.CUST_NAME	CUST_NAME
L03.CUST_GROUP_ID	CUST_GROUP_ID
L03.CUST_TYPE_ID	CUST_TYPE_ID
L03.CUST_COUNTRY_ID	CUST_COUNTRY_ID
L03.REC_VERSION	REC_VERSION
L03.REC_EFFDT	REC_EFFDT
"N"	REC_CURRENT_IND

L04_InsertNew

Constraint:

Derivation	Column Name
L03.CUST_ID	CUST_ID
L03.CUST_NAME_NEW	CUST_NAME
L03.CUST_GROUP_ID_NEW	CUST_GROUP_ID
L03.CUST_TYPE_ID_NEW	CUST_TYPE_ID
L03.CUST_COUNTRY_ID_NEW	CUST_COUNTRY_ID
L03.REC_VERSION+1	REC_VERSION
Oconv(@DATE-2*@INROWNUM, "D-YMD[4,2,2]")	REC_EFFDT
"Y"	REC_CURRENT_IND

L03

	Column name	Key	SQL type	Length	Scale	Nullabl
1	CUST_ID	✓	VarChar	255		No
2	CUST_NAME	☐	VarChar	50		Yes

L04_UpdateOld | L04_InsertNew

	Column name	Key	SQL type	Length	Scale	Nullable	Display	Data element	Description
1	CUST_ID	✓	VarChar	10		No	7		
2	CUST_NAME	☐	VarChar	50		Yes	50		<none>

ठ ODBC Update stage (O_DW_Customers_SCD2_Upd) - update action 'Update existing rows only' and the selected **key columns are CUST_ID and REC_VERSION** so they will appear in the constructed where part of an SQL statement.

ठ ODBC Insert stage (O_DW_Customers_SCD2_Ins) - insert action 'insert rows without clearing' and the **key column is CUST_ID**.

D_CUSTOMER dimension table after Datawarehouse refresh							
CUST_ID	**CUST_**	**CUST_**	**CUST_**	**CUST_**	**REC_**	**REC_**	**REC_**
	NAME	**GROUP_ID**	**TYPE_ID**	**COUNTRY_ID**	**VERSION**	**EFFDT**	**CURRENT_IND**
DRBOUA7	Dream Basket	EL	S	PL	1	01-10-2006	Y
ETIMAA5	**ETL tools info**	**BI**	**C**	**FI**	**1**	**29-09-2006**	**N**
FAMMFA0	Fajatso	FD	S	CD	1	27-09-2006	Y
FICILA0	First Pactonic	FD	C	IT	1	25-09-2006	Y
FRDXXA2	Frasir	EL	C	SK	1	23-09-2006	Y

GAMOPA9	Ganpa LTD.	FD	C	US	1	21-09-2006	Y
GGMOPA9	GG electronics	EL	S	RU	1	19-09-2006	Y
GLMFIA6	Glasithklini	FD	S	PL	1	17-09-2006	Y
GLMPEA9	Globiteleco	TC	S	FI	1	15-09-2006	Y
GONDWA5	Goli Airlines	BN	S	GB	1	13-09-2006	Y

IBM INFOSPHERE DATASTAGE PERFORMANCE TUNING: OVERVIEW OF BEST PRACTICES

INTRODUCTION

Data integration processes are very time and resource consuming. The amount of data and the size of the datasets are constantly growing but data and information are still expected to be delivered on-time. Performance is therefore a key element in the success of a Business Intelligence & Data Warehousing project and in order to guarantee the agreed level of service, management of data warehouse performance and performance tuning have to take a full role during the data warehouse and ETL development process.

Tuning, however, is not always straightforward. A chain is only as strong as its weakest link. In this context, there are five crucial domains that require attention when tuning an IBM Infosphere DataStage environment :

- System Infrastructure
- Network
- Database

- IBM DataStage Installation & Configuration
- IBM DataStage Jobs

It goes without saying that without a well performing infrastructure the tuning of IBM DataStage Jobs will not make much of a difference. As the first three domains are usually outside the control of the ETL development team, this article will only briefly touch upon these subjects and will mainly focus on the topics related to the developments done within the IBM InfoSphere DataStage layer. There are also major differences between the underlying architecture of the DataStage Server Edition and the DataStage Parallel Edition. This article will only cover performance tuning for the IBM InfoSphere DataStage Enterprise Edition v 8.x.

One of the first steps of performance tuning, is monitoring the current performance of the DataStage jobs. It is very important to understand what step in the job is consuming the most time and resources. To do this analysis several tools and functionalities of IBM Infosphere DataStage can be used.

PERFORMANCE MONITORING BEST PRACTICES

Usage of Job Monitor

The IBM InfoSphere DataStage job monitor can be accessed through the IBM InfoSphereDataStage Director. The job monitor provides a useful snapshot of a job's performance at a certain moment of its execution, but does not provide thorough performance metrics. Due to buffering and to some job semantics, a snapshot image of the flow might not be a representative sample of the performance over the course of the entire job. The CPU summary information provided by the job monitor is useful as a first approximation of where time is being spent in the flow. That is why a job monitor snapshot should not be used in place of a full run of the job, or a run with a sample set of data as it does not include information on sorts or similar components that might be inserted automatically by the engine in a parallel job. For these components, the score dump can be of assistance.

Usage of Score Dump

In order to resolve any performance issues it is essential to have an understanding of the data flow within the jobs. To help understand a job flow, a score dump should be taken. This can be done by setting the APT_DUMP_SCORE environment variable to “true” prior to running the job.

When enabled, the score dump produces a report which shows the operators, processes and data sets in the job and contains information about :

- Where and how data was repartitioned.
- Whether IBM InfoSphere DataStage has inserted extra operators in the flow.
- The degree of parallelism each operator has run with, and on which nodes.
- Where data was buffered.

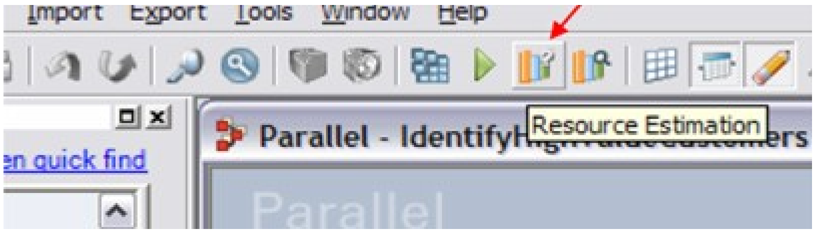
The score dump information is included in the job log when a job is run and is particularly useful in showing where IBM InfoSphere DataStage is inserting additional components/actions in the job flow, in particular extra data partitioning and sorting operators as they can both be detrimental to performance. A score dump will help to detect superfluous operators and amend the job design to remove them.

Usage of Resource Estimation

Predicting hardware resources needed to run DataStage jobs in order to meet processing time requirements can sometimes be more of an art than a science.

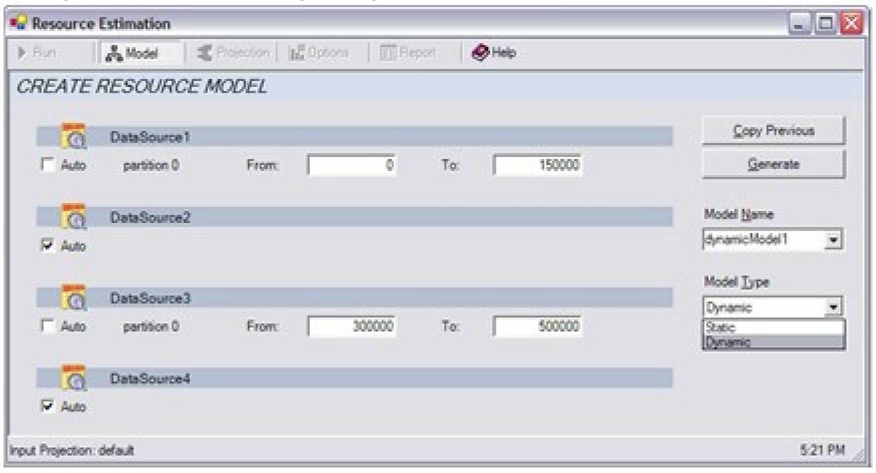
With new sophisticated analytical information and deep understanding of the parallel framework, IBM has added Resource Estimation to DataStage (and QualityStage) 8.x. This can be used to determine the needed system requirements or to analyze if the current infrastructure can support the jobs that have been created.

Within a job design, a new toolbar option is available called Resource Estimation.



This option opens a dialog called Resource Estimation. The Resource Estimation is based on a modelization of the job. There are two types of models that can be created:

- Static. The static model does not actually run the job to create the model. CPU utilization cannot be estimated, but disk space can. The record size is always fixed. The "best case" scenario is considered when the input data is propagated. The "worst case" scenario is considered when computing record size.
- Dynamic. The Resource Estimation tool actually runs the job with a sample of the data. Both CPU and disk space are estimated. This is a more predictable way to produce estimates.



CREATE RESOURCE MODEL

DataSource1
☐ Auto partition 0 From: 0 To: 150000

DataSource2
☑ Auto

DataSource3
☐ Auto partition 0 From: 300000 To: 500000

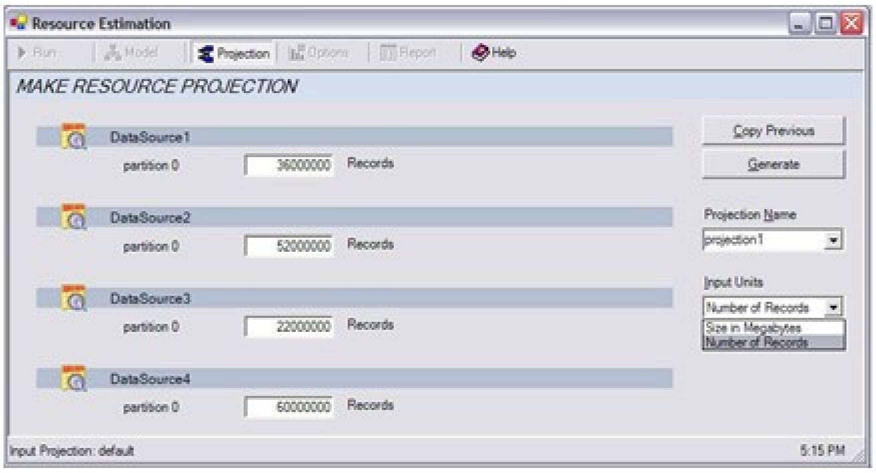
DataSource4
☑ Auto

Model Name: dynamicModel1

Model Type: Dynamic (Static, Dynamic)

Input Projection: default 5:21 PM

Resource Estimation is used to project the resources required to execute the job based on varying data volumes for each input data source.



MAKE RESOURCE PROJECTION

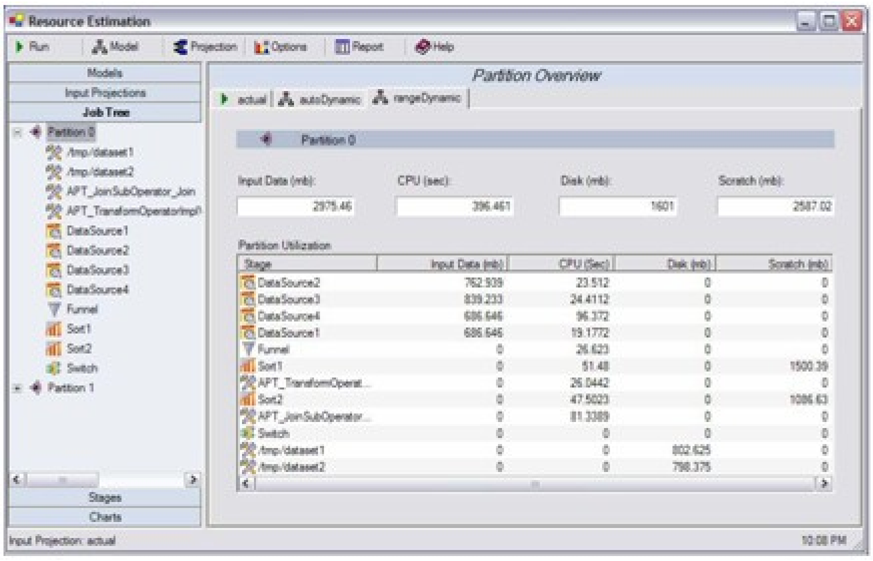
DataSource1		
partition 0	36000000	Records
DataSource2		
partition 0	52000000	Records
DataSource3		
partition 0	22000000	Records
DataSource4		
partition 0	60000000	Records

Projection Name: projection1

Input Units: Number of Records (Size in Megabytes / Number of Records)

Input Projection: default 5:15 PM

A projection is then executed using the model selected. The results show the total CPU needed, disk space requirements, scratch space requirements, and other relevant information.



Partition Overview

Partition 0

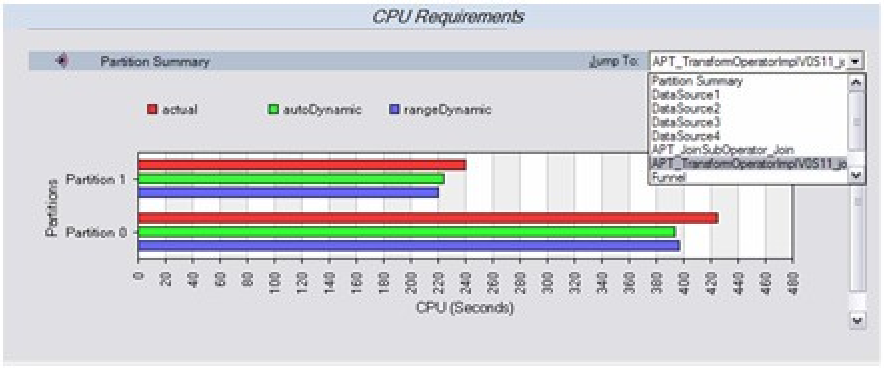
Input Data (mb): 2975.46 CPU (sec): 396.461 Disk (mb): 1601 Scratch (mb): 2587.02

Partition Utilization

Stage	Input Data (mb)	CPU (Sec)	Disk (mb)	Scratch (mb)
DataSource2	762.939	23.512	0	0
DataSource3	839.233	24.4112	0	0
DataSource4	686.646	96.372	0	0
DataSource1	686.646	19.1772	0	0
Funnel	0	26.623	0	0
Sort1	0	51.48	0	1500.39
APT_TransformOperat...	0	26.0442	0	0
Sort2	0	47.5023	0	1086.63
APT_JoinSubOperator...	0	81.3389	0	0
Switch	0	0	0	0
/tmp/dataset1	0	0	802.625	0
/tmp/dataset2	0	0	798.375	0

Input Projection: actual 10:08 PM

Different projections can be run with different data volumes and each can be saved. Graphical charts are also available for analysis, which allow the user to drill into each stage and each partition. A report can be generated or printed with the estimations.



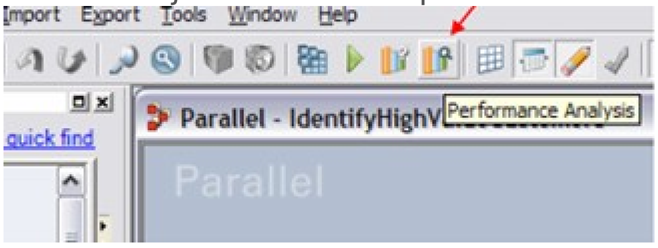
This feature will greatly assist developers in estimating the time and machine resources needed for job execution. This kind of analysis can help when analyzing the performance of a job, but IBM DataStage also offers another possibility to analyze job performance.

Usage of Performance Analysis

Isolating job performance bottlenecks during a job execution or even seeing what else was being performed on the machine during the job run can be extremely difficult. IBM Infosphere DataStage 8.x adds a new capability called Performance Analysis.

It is enabled through a job property on the execution tab which collects data at job execution time. (Note: by default, this option is disabled) . Once enabled and with a job open, a new toolbar option, called Performance Analysis, is made available .

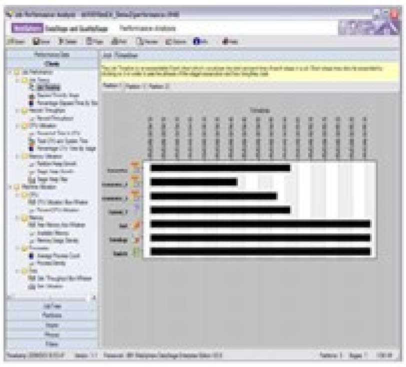
This option opens a new dialog called Performance Analysis. The first screen asks the user which job instance to perform the analysis on.

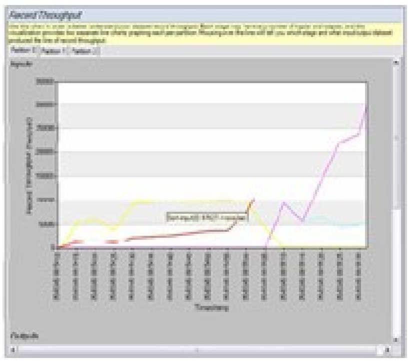


Detailed charts are then available for that specific job run including:

- Job timeline
- Record Throughput
- CPU Utilization
- Job Timing
- Job Memory Utilization
- Physical Machine Utilization (shows what else is happening overall on the machine, not just the DataStage activity).

Each partition’s information is available in different tabs.





A report can be generated for each chart.

Using the information in these charts, a developer can for instance pinpoint performance bottlenecks and re-design the job to improve performance.

In addition to instance performance, overall machine statistics are available. When a job is running, information about the machine is also collected and is available in the Performance Analysis tool including:

- Overall CPU Utilization
- Memory Utilization
- Disk Utilization

Developers can also correlate statistics between the machine information and the job performance. Filtering capabilities exist to only display specific stages.

The information collected and shown in the Performance Analysis tool can easily be analyzed to identify possible bottlenecks. These bottlenecks are usually situated in the general job design, which will be described in the following chapter.

GENERAL JOB DESIGN BEST PRACTICES

The ability to process large volumes of data in a short period of time depends on all aspects of the flow and the environment being optimized for maximum throughput and performance. Performance tuning and optimization are iterative processes that begin with job design and unit tests, proceed through integration and volume testing, and continue throughout the production life cycle of the application. Here are some performance pointers:

Columns and type conversions

Remove unneeded columns as early as possible within the job flow. Every additional unused column requires additional buffer memory, which can impact performance and make each row transfer from one stage to the next more expensive. If possible, when reading from databases, use a select list to read only the columns required, rather than the entire table. Avoid propagation of unnecessary metadata between the stages. Use the Modify stage and drop the metadata. The Modify stage will drop the metadata only when explicitly specified using the DROP clause.

So only columns that are really needed in the job should be used and the columns should be dropped from the moment they are not needed anymore. The OSH_PRINT_SCHEMAS environment variable can be set to verify that runtime schemas match the job design column definitions. When using stage variables on a Transformer stage, ensure that their data types match the expected result types. Avoid that DataStage needs to perform unnecessary type conversions as it will use time and resources for these conversions.

Transformer stages

It is best practice to avoid having multiple stages where the functionality could be incorporated into a single stage, and use other stage types to perform simple transformation operations. Try to balance load on Transformers by sharing the transformations across existing Transformers. This will ensure a smooth flow of data.

When type casting, renaming of columns or addition of new columns is required, use Copy or Modify Stages to achieve this. The Copy stage, for example, should be used instead of a Transformer for simple operations including :

- Job Design placeholder between stages
- Renaming Columns
- Dropping Columns
- Implicit (default) Type Conversions

A developer should try to minimize the stage variables in a Transformer stage because the performance of a job decreases as stage variables are added in a Transformer stage. The number of stage variables should be limited as much as possible.

Also if a particular stage has been identified as one that takes a lot of time in a job, like a Transformer stage having complex functionality with a lot of stage variables and transformations, then the design of jobs could be done in such a way that this stage is put in a separate job all together (more resources for the Transformer Stage).

While designing IBM DataStage Jobs, care should be taken that a single job is not overloaded with stages. Each extra stage put into a job corresponds to less resources being available for every stage, which directly affects the job performance. If possible, complex jobs having a large number of stages should be logically split into smaller units.

Sorting

A sort done on a database is usually a lot faster than a sort done in DataStage. So – if possible – try to already do the sorting when reading data from the database instead of using a Sort stage or sorting on the input link. This could also mean a big performance gain in the job, although it is not always possible to avoid needing a Sort stage in jobs.

Careful job design can improve the performance of sort operations, both in standalone Sort stages and in on-link sorts specified in other stage types, when not being able to make use of the database sorting power.

If data has already been partitioned and sorted on a set of key columns, specify the ″don’t sort, previously sorted″ option for the key columns in the Sort stage. This reduces the cost of sorting and takes more advantage of pipeline parallelism. When writing to parallel data sets, sort order and partitioning are preserved. When reading from these data sets, try to maintain this sorting if possible by using theSame partitioning method.

The stable sort option is much more expensive than non-stable sorts, and should only be used if there is a need to maintain row order other than as needed to perform the sort.

The performance of individual sorts can be improved by increasing the memory usage per partition using the Restrict Memory Usage(MB) option of the Sort stage. The default setting is 20 MB per partition. Note that sort memory usage can only be specified for standalone Sort stages, it cannot be changed for inline (on a link) sorts.

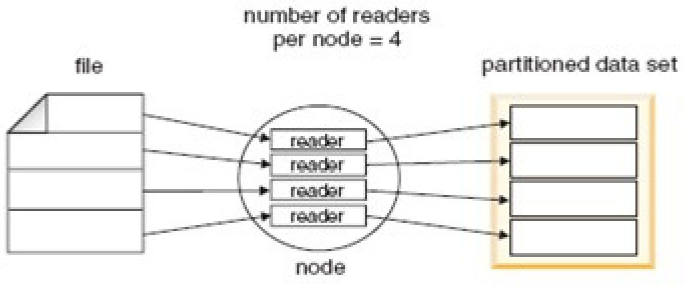
Sequential files

While handling huge volumes of data, the Sequential File stage can itself become one of the major bottlenecks as reading and writing from this stage is slow. Certainly do not use sequential files for intermediate storage between jobs. It causes performance overhead, as it needs to do data conversion before writing and reading from a file. Rather Dataset stages should be used for intermediate storage between different jobs.

Datasets are key to good performance in a set of linked jobs. They help in achieving end-to-end parallelism by writing data in partitioned form and maintaining the sort order. No repartitioning or import/export conversions are needed.

In order to have faster reading from the Sequential File stage the number of readers per node can be increased (default value is one). This means, for example, that a single file can be partitioned as it is read (even though the stage is constrained to running sequentially on the conductor mode).

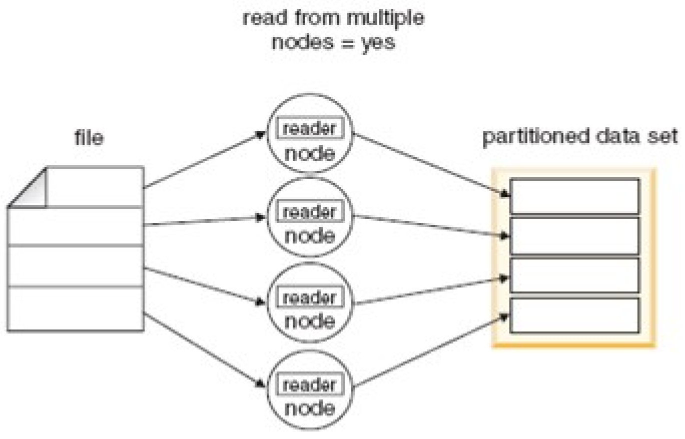
This is an optional property and only applies to files containing fixed-length records. But this provides a way of partitioning data contained in a single file. Each node reads a single file, but the file can be divided according to the number of readers per node, and written to separate partitions. This method can result in better I/O performance on an SMP (Symmetric Multi Processing) system.



Read from multiple nodes

It can also be specified that single files can be read by multiple nodes. This is also an optional property and only applies to files containing fixed-

length records. Set this option to “Yes” to allow individual files to be read by several nodes. This can improve performance on cluster systems.
IBM DataStage knows the number of nodes available, and using the fixed length record size, and the actual size of the file to be read, allocates to the reader on each node a separate region within the file to process. The regions will be of roughly equal size.



The options “Read From Multiple Nodes” and “Number of Readers Per Node” are mutually exclusive.

Runtime Column Propagation

Also while designing jobs, care must be taken that unnecessary column propagation is not done. Columns, which are not needed in the job flow, should not be propagated from one stage to another and from one job to the next. As much as possible, RCP (Runtime Column Propagation) should be disabled in the jobs.

Join, Lookup or Merge

One of the most important mistakes that developers make is to not have volumetric analyses done before deciding to use Join, Lookup or Merge stages.

IBM DataStage does not know how large the data set is, so it cannot make an informed choice whether to combine data using a Join stage or a Lookup stage. Here is how to decide which one to use …

There are two data sets being combined. One is the primary or driving data set, sometimes called the left of the join. The other dataset are the reference data set or the right of the join.

In all cases, the size of the reference data sets is a concern. If these take up a large amount of memory relative to the physical RAM memory size of the computer DataStage is running on, then a Lookup stage might crash because the reference datasets might not fit in RAM along with everything else that has to be in RAM. This results in a very slow performance since each lookup operation can, and typically will, cause a page fault and an I/O operation.

So, if the reference datasets are big enough to cause trouble, use a join. A join does a high-speed sort on the driving and reference datasets. This can involve I/O if the data is big enough, but the I/O is all highly optimized and sequential. After the sort is over, the join processing is very fast and never involves paging or other I/O.

Databases

The best choice is to use Connector stages if available for the database. The next best choice is the Enterprise database stages as these give maximum parallel performance and features when compared to 'plug-in' stages. The Enterprise stages are:

- DB2/UDB Enterprise
- Informix® Enterprise
- Oracle Enterprise
- Teradata Enterprise
- SQLServer Enterprise
- Sybase Enterprise
- ODBC Enterprise
- iWay Enterprise
- Netezza Enterprise

Avoid generating target tables in the database from the IBM DataStage job (that is, using the Create write mode on the database stage) unless they are intended for temporary storage only. This is because this method does not allow, for example, specifying target table space, and inadvertently data-management policies on the database can be violated.

When there is a need to create a table on a target database from within a job, use the Open command property on the database stage to explicitly create the table and allocate table space, or any other options required.

The Open command property allows to specify a command (for example some SQL) that will be executed by the database before it processes any data from the stage. There is also a Close property that allows specifying a command to execute after the data from the stage has been processed. (Note that, when using user-defined Open and Close commands, locks should be specified where appropriate).

Tune the database stages for ‘Array Size’ and ‘Rows per Transaction’ numerical values for faster inserts, updates and selects. Experiment in changing these values to see what the best performance is for the DataStage job. The default value used is low and not optimal in terms of performance.

Finally, try to work closely with the database administrators so they can examine the SQL-statements used in DataStage jobs. Appropriate indexes on tables can deliver a better performance of DataStage queries.

Also try to examine if the job is faster when the indexes are dropped before data loading and recreated after loading data into the tables. Recreation of the indexes also takes some time, so test if this has a performance gain or a performance loss on the total process chain.

CONCLUSION

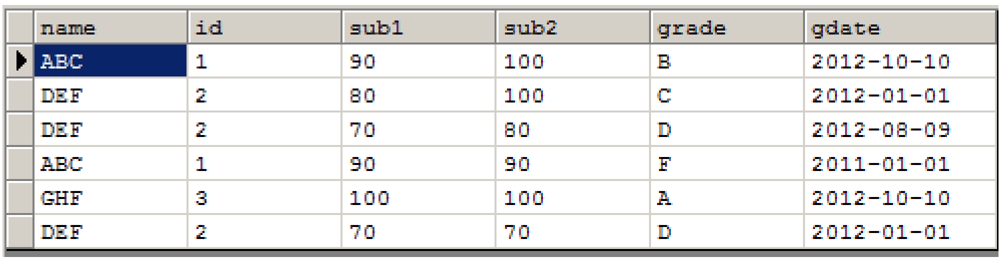
Performance tuning can be a labor intensive and quite costly process. That is exactly the reason why care for optimization and performance should be taken into account from the beginning of the development process. With the combination of best practices, performance guidelines and past experience, the majority of performance problems can be avoided during the development process.

If performance issues still occur even when performance guidelines have been taken into account during development, then these issues can be tackled and analyzed using the available, discussed tools such as Resource Estimation and Performance Analysis functionalities.

LOOPING IN TRANSFORMER STAGE – DATASTAGE 8.5 : EXAMPLE 2

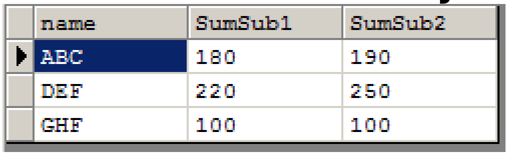
Performing aggregation using a Transformer:

Input: Below is the sample data of three students, their marks in two subjects, the corresponding grades and the dates on which they were graded.



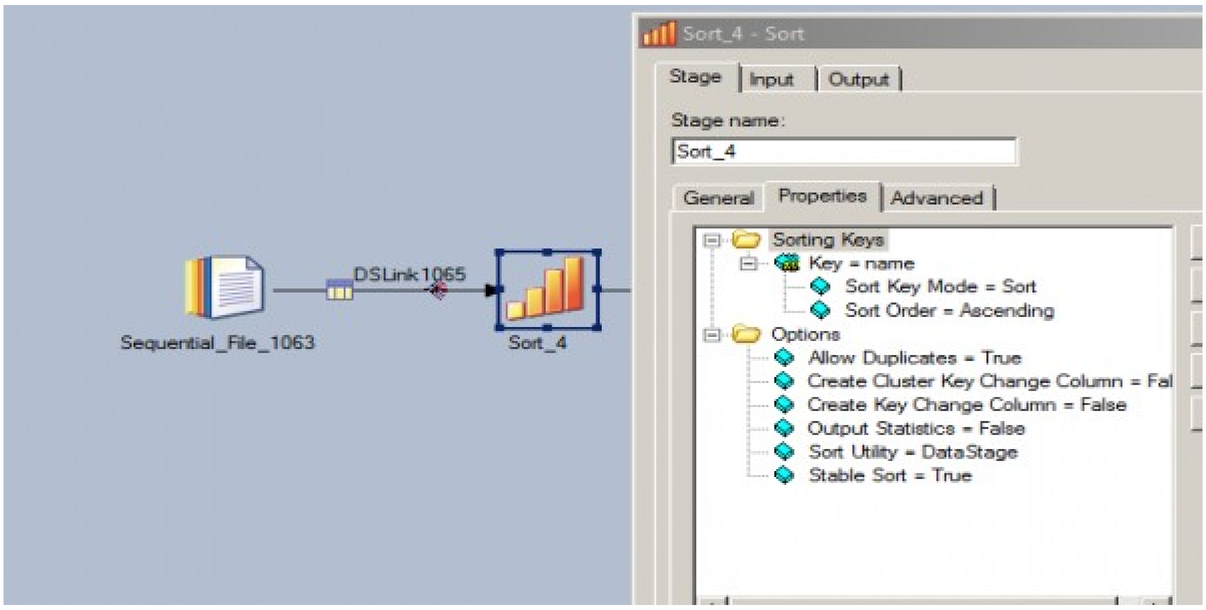
name	id	sub1	sub2	grade	gdate
ABC	1	90	100	B	2012-10-10
DEF	2	80	100	C	2012-01-01
DEF	2	70	80	D	2012-08-09
ABC	1	90	90	F	2011-01-01
GHF	3	100	100	A	2012-10-10
DEF	2	70	70	D	2012-01-01

Output:Our requirement is to sum the marks obtained by each student in a subject and display it in the output



name	SumSub1	SumSub2
ABC	180	190
DEF	220	250
GHF	100	100

Step 1: Once we have read the data from the source we have to sort data on our key field. In our example the key field is the student name



Stage name: Sort_4

```
Sorting Keys
    Key = name
        Sort Key Mode = Sort
        Sort Order = Ascending
Options
    Allow Duplicates = True
    Create Cluster Key Change Column = Fal
    Create Key Change Column = False
    Output Statistics = False
    Sort Utility = DataStage
    Stable Sort = True
```

Once the data is sorted we have to implement the looping function in transformer to calculate the aggregate value

Before we get into the details, we need to know a couple of functions

- **SaveInputRecord(): This function saves the entire record in cache and returns the number of records that are currently stored in cache**

- **LastRowInGroup(input-column): When a input key column is passed to this function it will return 1 when the last row for that column is found and in all other cases it will return 0**
  To give an example, lets say our input is

Student	Code
ABC	1
ABC	2
ABC	3
DEF	2

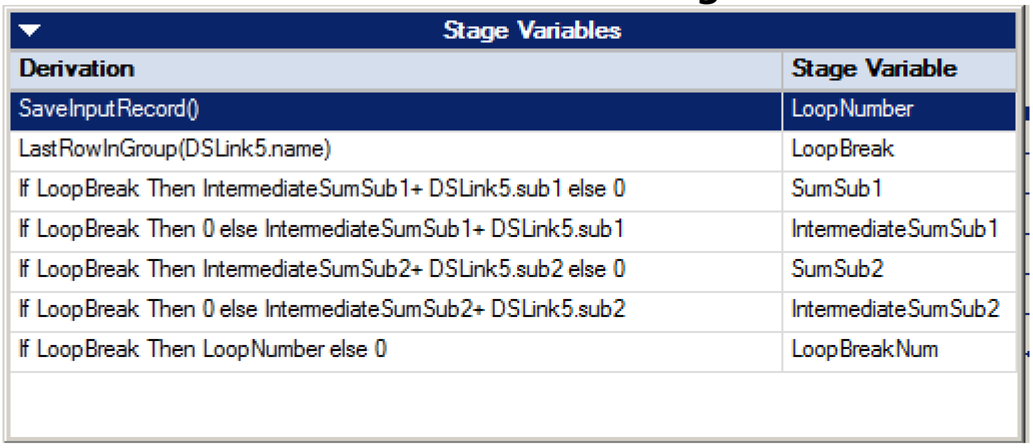
- **For the first two records the function will return 0 but for the last record ABC,3 it will return 1 indicating that it is the last record for the group where student name is "ABC"**

- **GetSavedInputRecord(): This function returns the record that was stored in cache by the function SaveInputRecord()**

Back to the task at hand, we need 7 stage variables to perform the aggregation operation successfully.

1. LoopNumber: Holds the value of number of records stored in cache for a student
2. LoopBreak: This is to identify the last record for a particular student
3. SumSub1: This variable will hold the final sum of marks for each student in subject 1
4. IntermediateSumSub1: This variable will hold the sum of marks until the final record is evaluated for a student (subject 1)
5. SumSub2: Similar to SumSub1 (for subject 2)
6. IntermediateSumSub2: Similar to IntermediateSumSub1 (for subject 2)
7. LoopBreakNum: Holds the value for the number of times the loop has to run

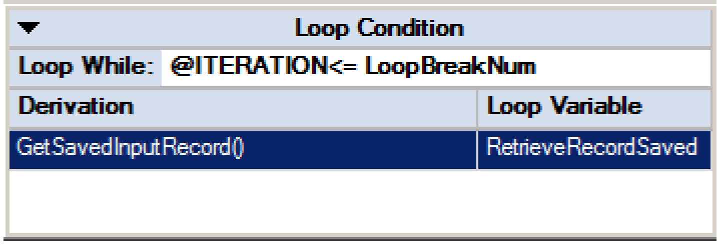
Below is the screenshot of the stage variables



Stage Variables

Derivation	Stage Variable
SaveInputRecord()	LoopNumber
LastRowInGroup(DSLink5.name)	LoopBreak
If LoopBreak Then IntermediateSumSub1+ DSLink5.sub1 else 0	SumSub1
If LoopBreak Then 0 else IntermediateSumSub1+ DSLink5.sub1	IntermediateSumSub1
If LoopBreak Then IntermediateSumSub2+ DSLink5.sub2 else 0	SumSub2
If LoopBreak Then 0 else IntermediateSumSub2+ DSLink5.sub2	IntermediateSumSub2
If LoopBreak Then LoopNumber else 0	LoopBreakNum

We also need to define the Loop Variables so that the loop will execute for a student until his final record is identified



Loop Condition

Loop While: @ITERATION<= LoopBreakNum

Derivation	Loop Variable
GetSavedInputRecord()	RetrieveRecordSaved

To explain the above use of variables -

When the first record comes to stage variables, it is saved in the cache using the function SaveInputRecord() in first stage variableLoopNumber

The second stage variable checks if it is the last record for this particular student, if it is it stores 1 else 0

The third SumSub1 is executed only if the record is the last record

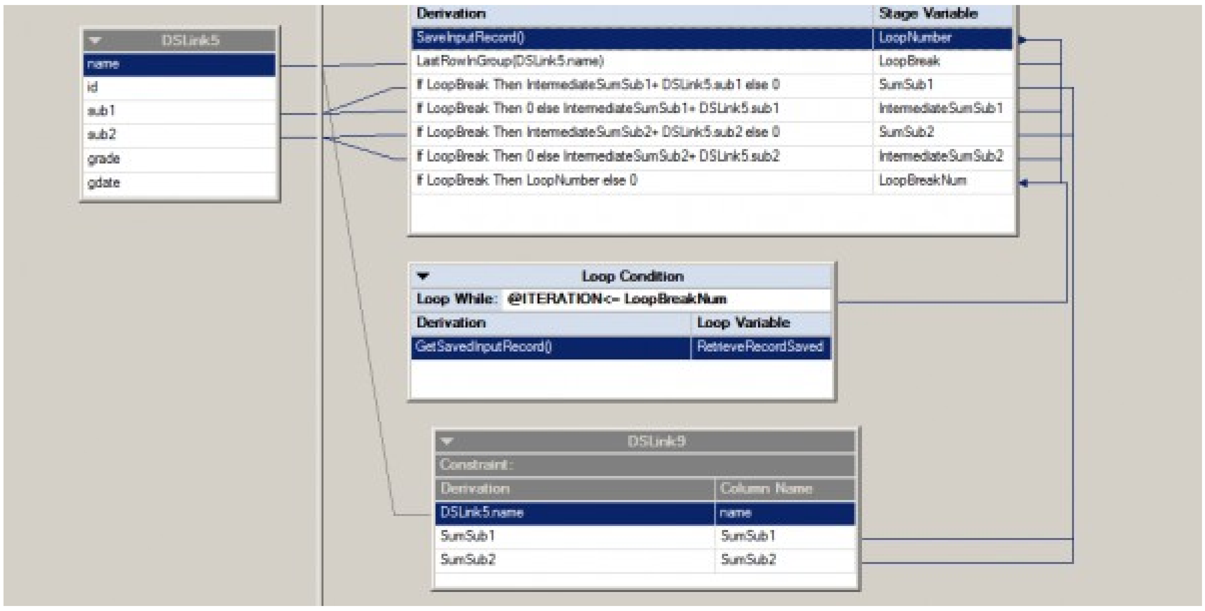
The fourth IntermediateSumSum1 is executed when the input record is not the last record, thereby storing the intermediate sum of the subject for a student

Fifth and sixth are the same as 3 and 4 stage variables

Seven will have the first value as 1 and for the second record also if the same student is fetched it will change to 2 and so on

The loop variable will be executed until the final record for a student is identified and the GetSavedInputRecord() function will make sure the current record is processed before the next record is brought for processing.

Finally your transformer should look like this



DSLink5

name
id
sub1
sub2
grade
gdate

Derivation	Stage Variable
SaveInputRecord()	LoopNumber
LastRowInGroup(DSLink5.name)	LoopBreak
If LoopBreak Then IntermediateSumSub1+ DSLink5.sub1 else 0	SumSub1
If LoopBreak Then 0 else IntermediateSumSub1+ DSLink5.sub1	IntermediateSumSub1
If LoopBreak Then IntermediateSumSub2+ DSLink5.sub2 else 0	SumSub2
If LoopBreak Then 0 else IntermediateSumSub2+ DSLink5.sub2	IntermediateSumSub2
If LoopBreak Then LoopNumber else 0	LoopBreakNum

Loop Condition

Loop While: @ITERATION<= LoopBreakNum

Derivation	Loop Variable
GetSavedInputRecord()	RetrieveRecordSaved

DSLink9

Constraint:

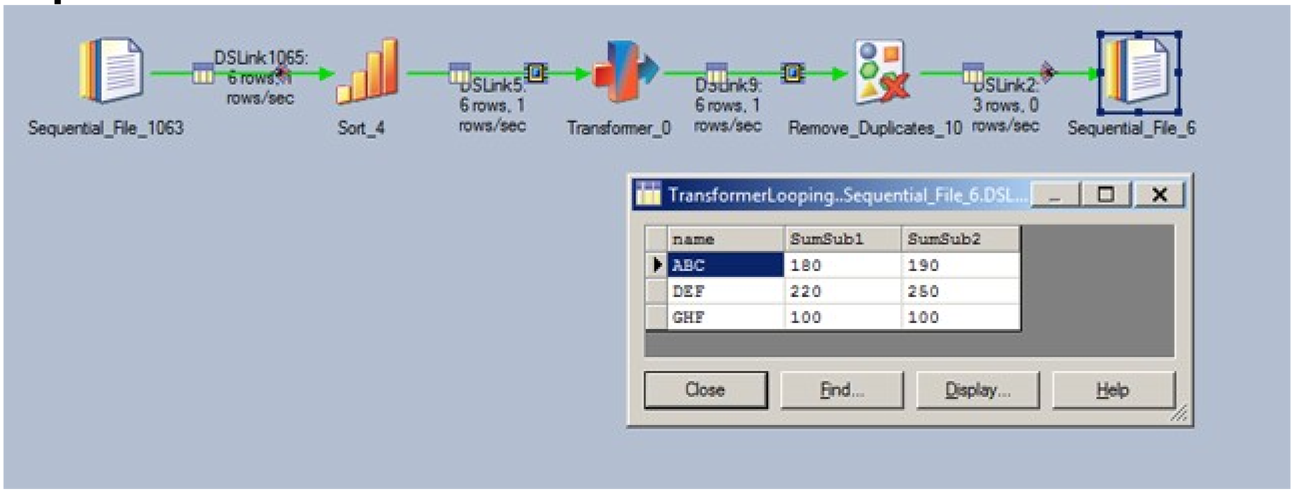
Derivation	Column Name
DSLink5.name	name
SumSub1	SumSub1
SumSub2	SumSub2

What the above logic does is for each and every record it will send the sum of marks scored by each student to the output. But our requirement is to have only one record per student in the output.

So we simply add a remove duplicates stage and add the student name as a primary key

Once done our job design is done it will look like this

Run the job and the output will be according to our initial expectation



TransformerLooping..Sequential_File_6.DSL...

name	SumSub1	SumSub2
ABC	180	190
DEF	220	250
GHF	100	100

We have successfully implemented AGGREGATION using TRANSFORMER Stage

LOOPING IN TRANSFORMER STAGE – DATASTAGE 8.5 : EXAMPLE 1

Looping in a Transformer was new feature that was introduced in DataStage 8.5. Now we don't have to design complex logic to perform looping functions on input data.

Rather than going on about what is new and what variables or functions have been added to achieve Looping in DataStage, we will look at a couple of examples that will explain all the new functionalities in a simple and easy way.

Example 1: How to convert a single row into multiple rows ?

Now you can argue that this is possible using a pivot stage. But for the sake of this article lets try doing this using a Transformer!

Below is a screenshot of our input data

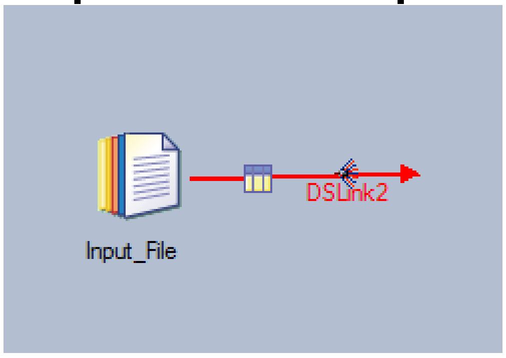
City	State	Name1	Name2	Name3
xy	FGH	Sam	Dean	Winchester

We are going to read the above data from a sequential file and transform it to look like this

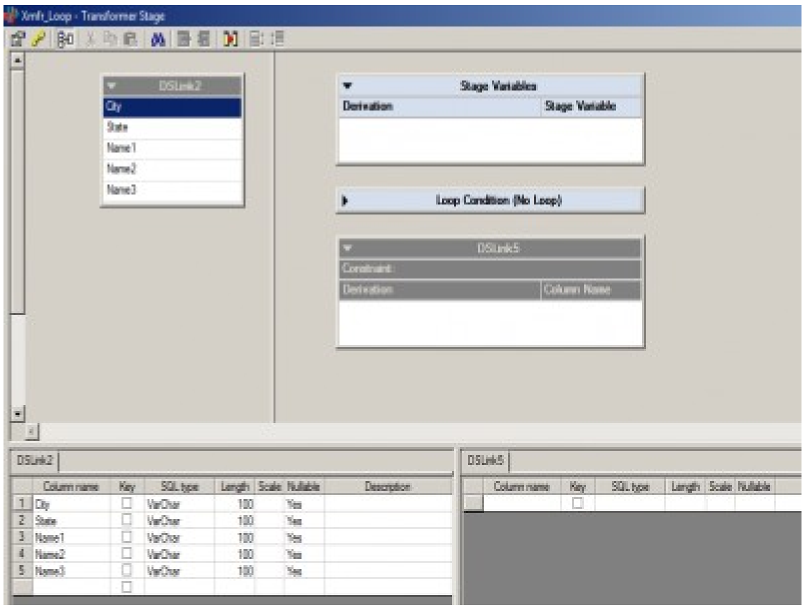
City	State	Name
xy	FGH	Sam
xy	FGH	Dean
xy	FGH	Winchester

So lets get to the job design

Step 1: Read the input data



Step 2: Logic for Looping in Transformer



DSLink2

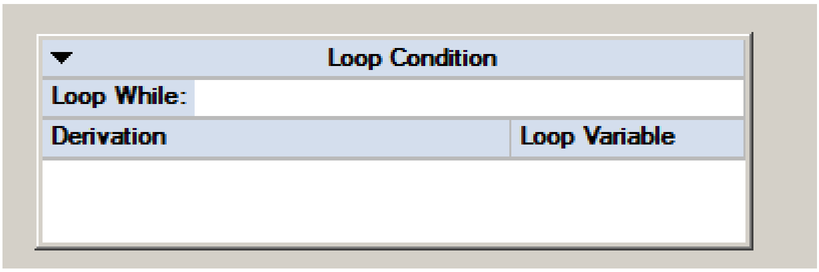
	Column name	Key	SQL type	Length	Scale	Nullable	Description
1	City	☐	VarChar	100		Yes	
2	State	☐	VarChar	100		Yes	
3	Name1	☐	VarChar	100		Yes	
4	Name2	☐	VarChar	100		Yes	
5	Name3	☐	VarChar	100		Yes	
		☐					

DSLink5

	Column name	Key	SQL type	Length	Scale	Nullable
		☐				

In the adjacent image you can see a new box called Loop Condition. This where we are going to control the loop variables.

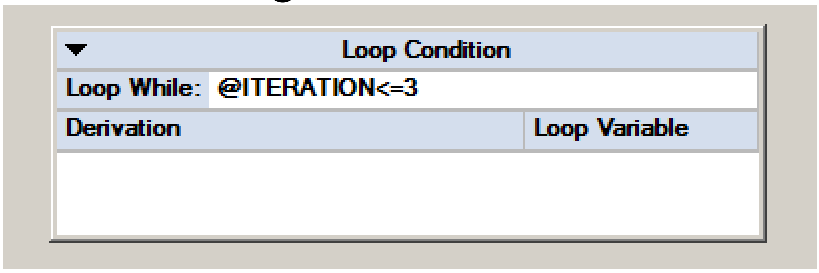
Below is the screenshot when we expand the Loop Condition box



The Loop While constraint is used to implement a functionality similar to "WHILE" statement in programming. So, similar to a while statement need to have a condition to identify how many times the loop is supposed to be executed.

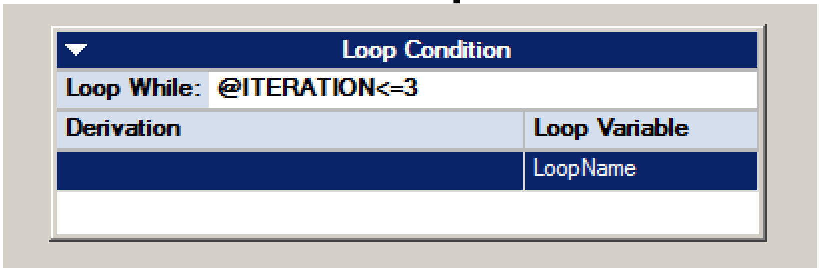
To achieve this @ITERATION system variable was introduced. In our example we need to loop the data 3 times to get the column data onto subsequent rows.

So lets have @ITERATION <=3



Loop While: @ITERATION<=3

Now create a new Loop variable with the name LoopName



Loop Condition

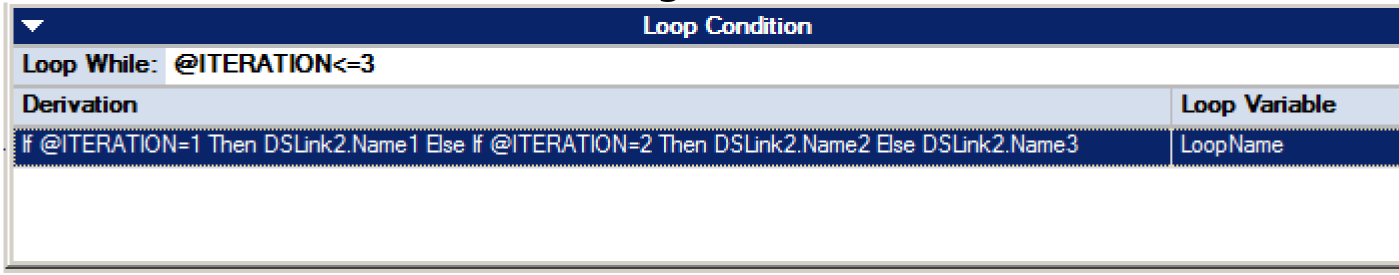
Loop While: @ITERATION<=3

Derivation	Loop Variable
	LoopName

The derivation for this loop variable should be

```
If @ITERATION=1 Then DSLink2.Name1 Else If @ITERATION=2 Then DSLink2.Name2
Else DSLink2.Name3
```

Below is a screenshot illustrating the same

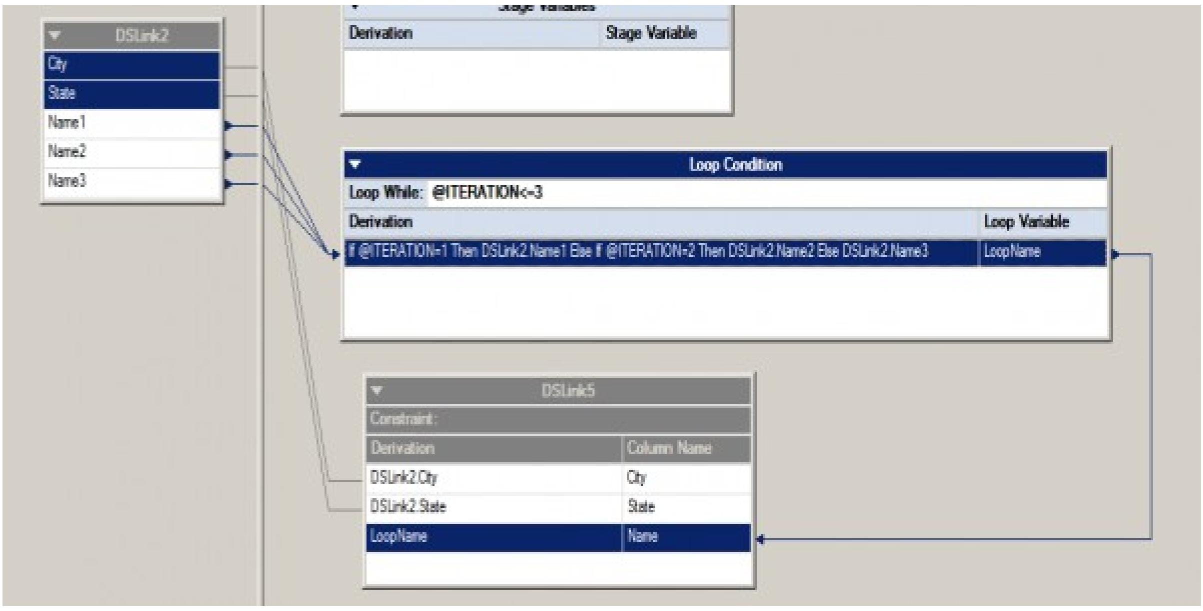


Loop Condition

Loop While: @ITERATION<=3

Derivation	Loop Variable
If @ITERATION=1 Then DSLink2.Name1 Else If @ITERATION=2 Then DSLink2.Name2 Else DSLink2.Name3	LoopName

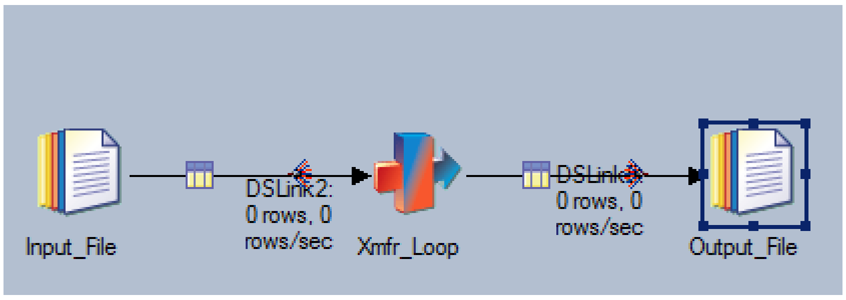
Now all we have to do is map this Loop variable LoopName to our output column Name



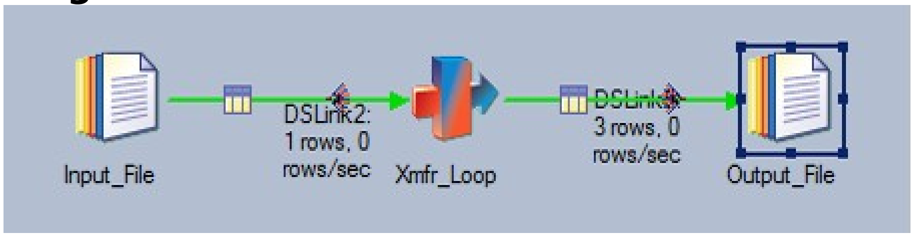
DSLink2
City
State
Name1
Name2
Name3
Derivation
Stage Variable
Loop Condition
Loop While: @ITERATION<=3
Derivation
Loop Variable
If @ITERATION=1 Then DSLink2.Name1 Else If @ITERATION=2 Then DSLink2.Name2 Else DSLink2.Name3
LoopName
DSLink5
Constraint:

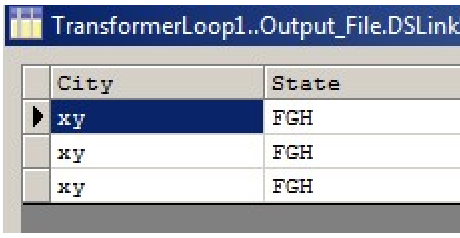
Derivation	Column Name
DSLink2.City	City
DSLink2.State	State
LoopName	Name

Lets map the output to a sequential file stage and see if the output is a desired.



After running the job, we did a view data on the output stage and here is the data as desired.




TransformerLoop1..Output_File.DSLink

City	State
xy	FGH
xy	FGH
xy	FGH

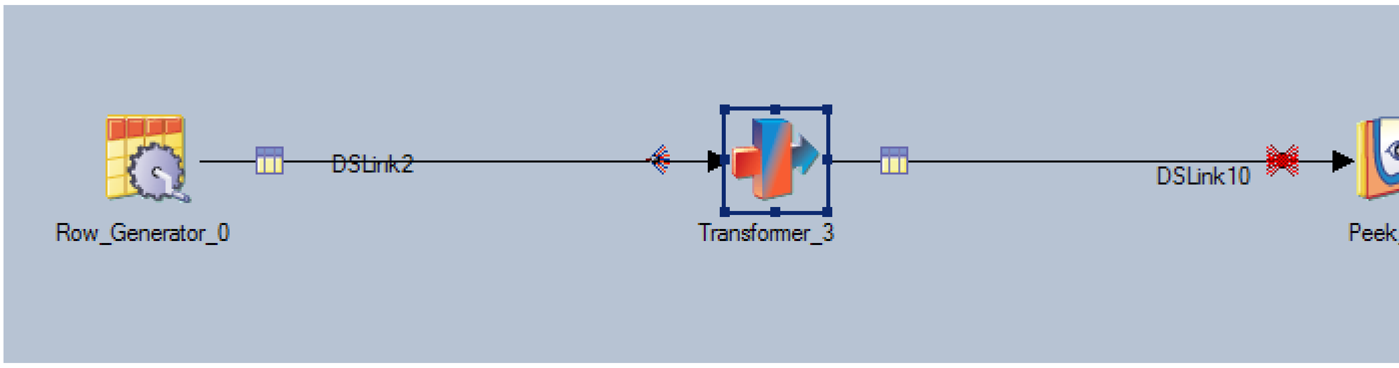
Making some tweaks to the above design we can implement things like

- **Adding new rows to existing rows**
- **Splitting data in a single column to multiple rows and many more such stuff..**

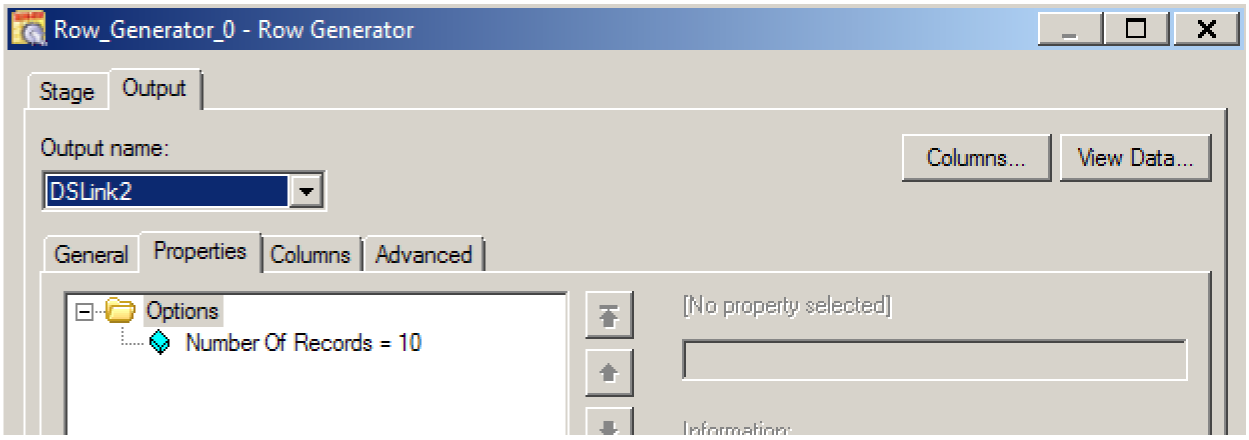
PARALLEL DEBUGGING IN DATASTAGE 8.5

IBM has added a new debug feature for DataStage from the 8.5 version. It provides basic debugging features for testing/debugging the job design.

This feature is used from the designer client. So lets jump into creating a simple job and start debugging it!

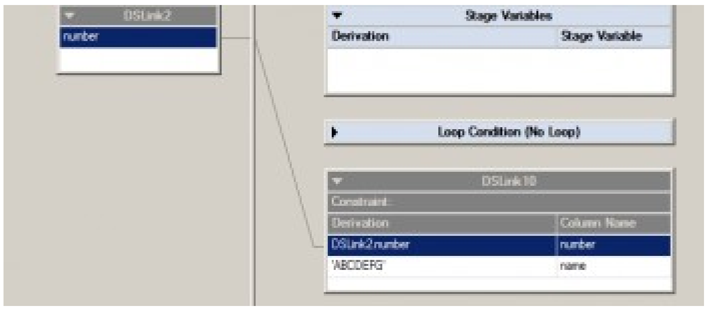


As you can see the above job design is very simple; I have a row generator stage that will generate 10 records for the column "number"



Options
Number Of Records = 10

And in the transformer I've hard-coded a column called "name" with the value 'ABCDEFG'

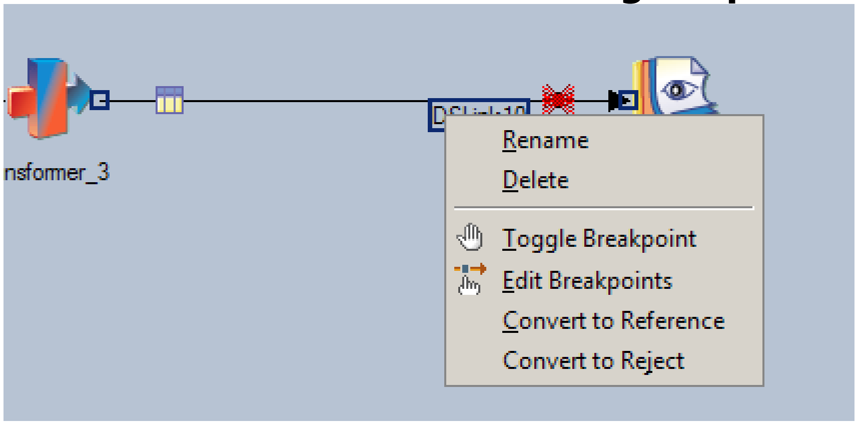


Derivation	Column Name
DSLink2.number	number
'ABCDEFG'	name

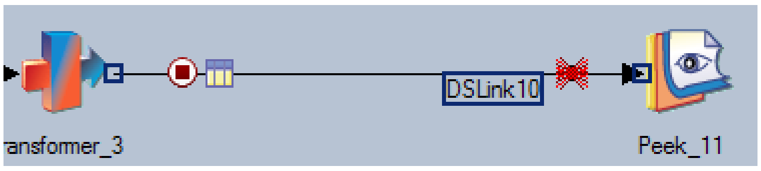
The final stage is the peek stage that we all have been using for debugging our job design all these years.
Now that we understand the job desing, we'll start debugging it.

Add breakpoints to links

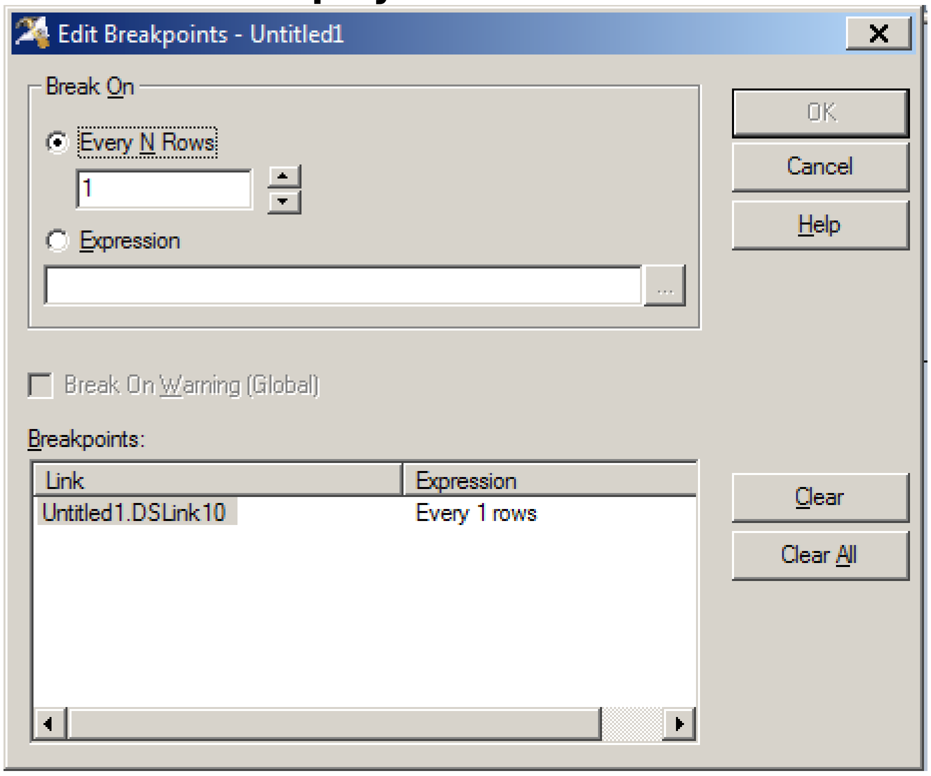
- **Right click on the link where you wish to check/debug the data and the following drop down menu appears**



- **Select the option, “Toggle Breakpoint”**
- **Once selected a small icon appears on the link as shown in the below screen shot**



- **We can also edit the Breakpoint, to do so we have to right click on the link again and select “Edit Breakpoints”. Following which the following popup will be displayed**



Break On

[✓ Every N Rows] 1

Expression

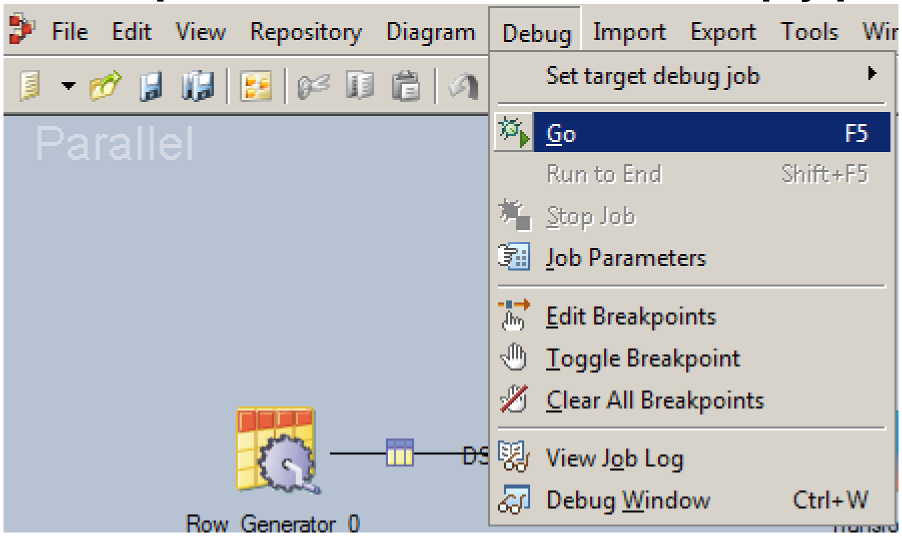
Break On Warning (Global)

Breakpoints:

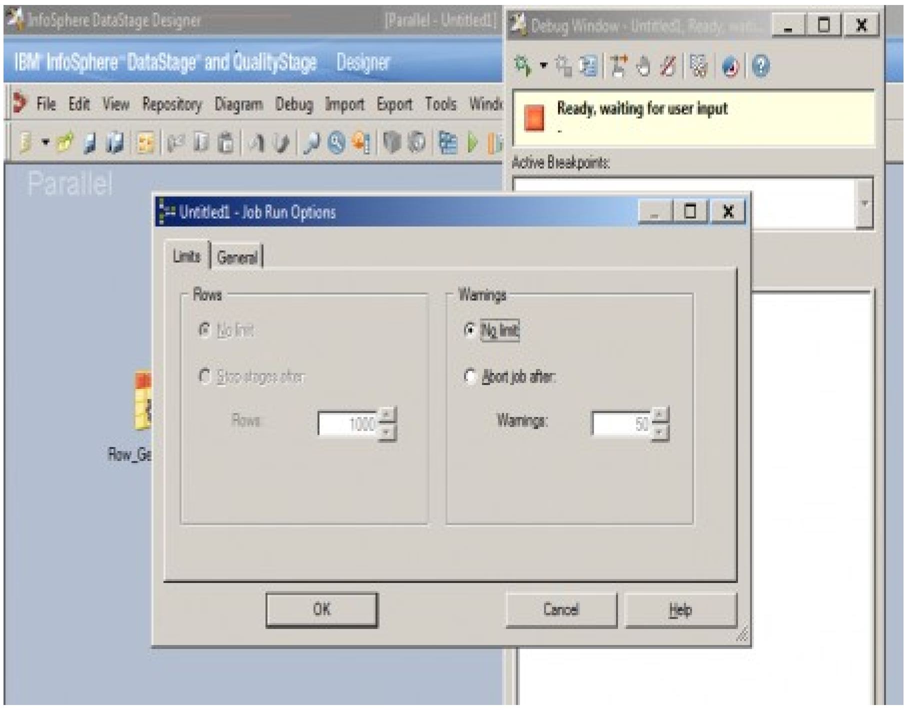
Link	Expression
Untitled1.DSLink10	Every 1 rows

- **By default the breakpoint will be activated for every row. If the need arises, we can change this to any value and the debugger will pause the flow of records as and when it reaches that number**
  Ex: If we set the value for “Every N Rows” to 5, the debugger will display every record that is a multiple of 5, i.e., 5th records, 10th record, 15th record an so on..
- **For the sake of this example we’ll change the value to 2**

We are done setting up the breakpoints for the debugger, so lets compile our job now. Once compiled, click on debug menu option and select "GO". Or simply press F5



The following popup appears and we are asked to run our job. Go ahead and run it -


InfoSphere DataStage Designer
[Parallel - Untitled1]
IBM InfoSphere DataStage and QualityStage Designer
File Edit View Repository Debug Import Export Tools
Debug Window - Untitled1, Ready, wait...
Parallel
Row_Ge

Ready, waiting for user input

Active Breakpoints:

Untitled1 - Job Run Options

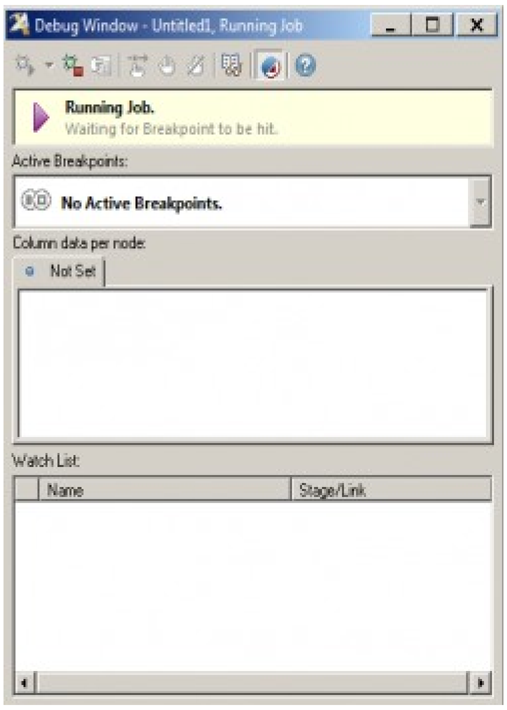
Limits | General

Rows
[✓ No limit]
Stop stages after
Rows: 1000

Warnings
[✓ No limit]
Abort job after:
Warnings: 50

OK | Cancel | Help

Screenshot of job running in debug mode


Debug Window - Untitled1, Running Job

Running Job.
Waiting for Breakpoint to be hit.

Active Breakpoints:

No Active Breakpoints.

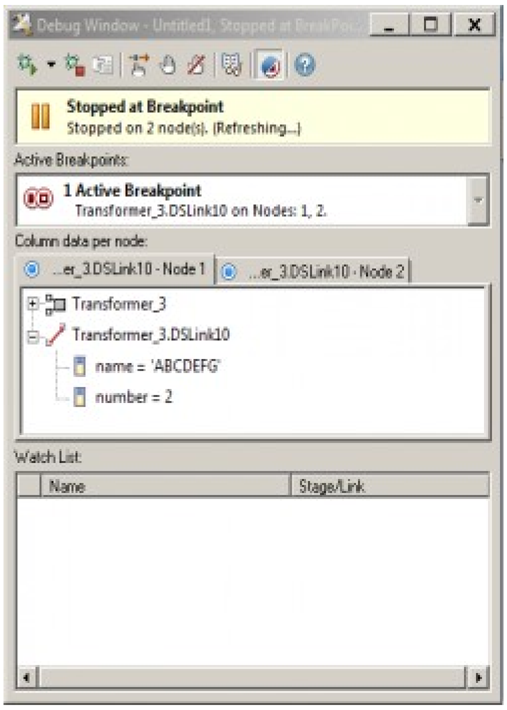
Column data per node:

Not Set

Watch List:

Name	Stage/Link

Now our job is running in debug mode and as soon as the 2nd record passes through the link (DSLink10) the details of that particular record will be displayed in the debugger window for our needs


Debug Window - Untitled1, Stopped at BreakPoi

Stopped at Breakpoint
Stopped on 2 node(s). (Refreshing...)

Active Breakpoints:

1 Active Breakpoint
Transformer_3.DSLink10 on Nodes: 1, 2.

Column data per node:

...er_3.DSLink10 - Node 1 | ...er_3.DSLink10 - Node 2

```
Transformer_3
Transformer_3.DSLink10
    name = 'ABCDEFG'
    number = 2
```

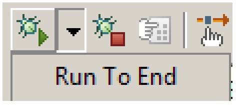
Watch List:

Name	Stage/Link

As you can see in the above screenshot, the data for that particular link is displayed along with the corresponding column name.

Once we are done analyzing the data we can request the debugger to either go to the next record of run till the end

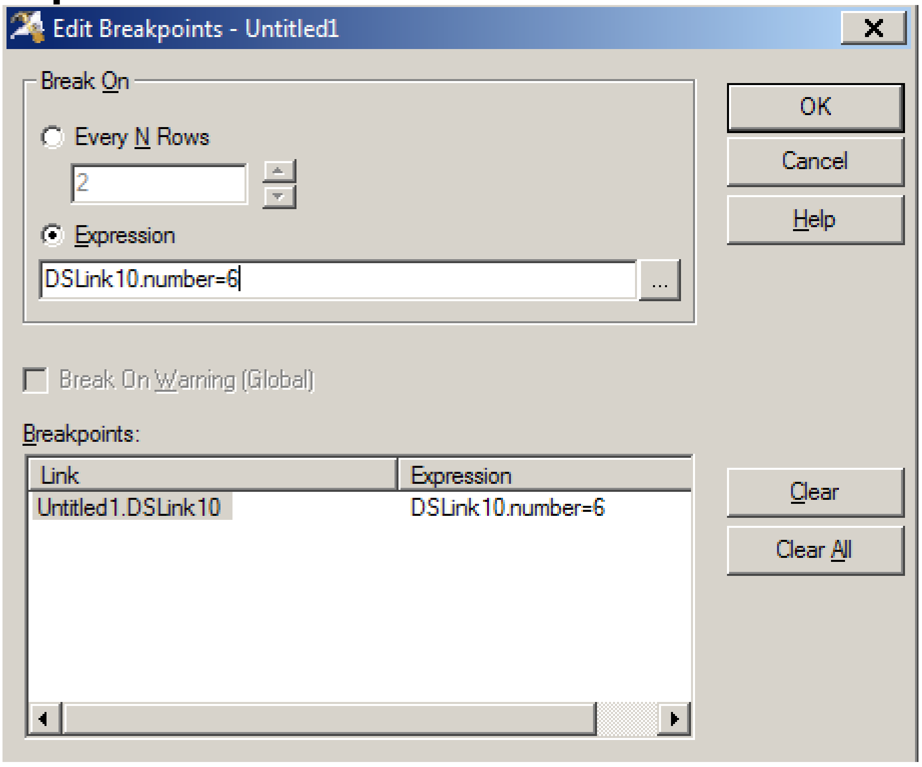
without hitting the breakpoints by clicking on the run button on the top left hand corner of the debugger screen



Debugging based on a condition
We saw until now how to set the debugger to break at a particular record count. Now lets select the option to break when a condition is satisfied.

In the Edit Breakpoint menu, we also have the option to select expression (shown in the above screen shots).

Select that option and enter a expression/condition which matches your needs.
For the sake of this example, I'll have the below expression



Edit Breakpoints - Untitled1

Break On

Every N Rows: 2

[✓ Expression]: DSLink10.number=6

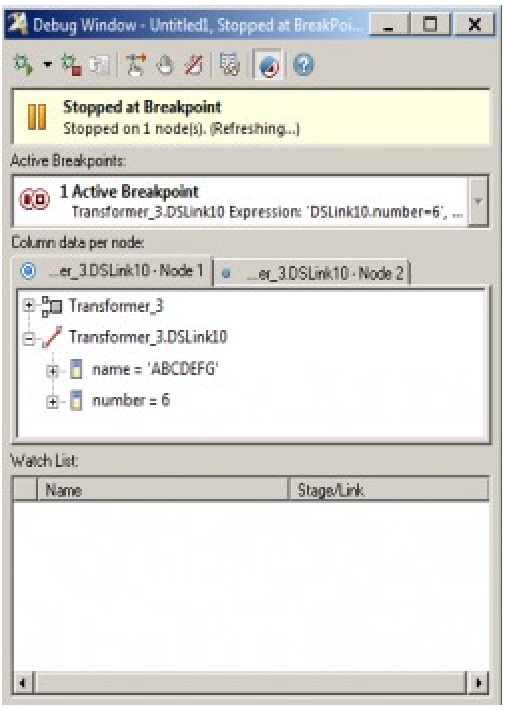
Break On Warning (Global)

Breakpoints:

Link	Expression
Untitled1.DSLink10	DSLink10.number=6

Now as per my condition, the debugger should pause the job when it encounters 6 in the column "number"

As you can see below the debugger paused the job when it saw the value 6 for the column number and showed the details for the other columns as well.


Debug Window - Untitled1, Stopped at BreakPoi...

Stopped at Breakpoint
Stopped on 1 node(s). (Refreshing...)

Active Breakpoints:

1 Active Breakpoint
Transformer_3.DSLink10 Expression: 'DSLink10.number=6', ...

Column data per node:

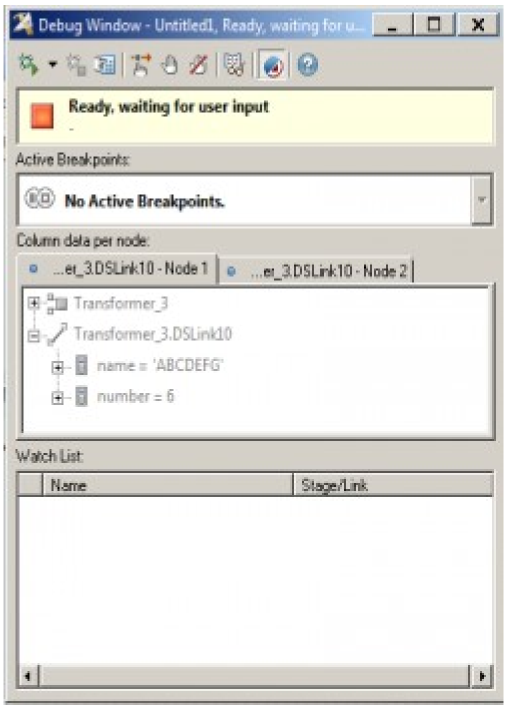
...er_3.DSLink10 - Node 1 | ...er_3.DSLink10 - Node 2

```
Transformer_3
Transformer_3.DSLink10
    name = 'ABCDEFG'
    number = 6
```

Watch List:

Name	Stage/Link

Select the "Run to End" option on the debugger window and the job will finish and wait for the next user input.


Debug Window - Untitled1, Ready, waiting for u...

Ready, waiting for user input

Active Breakpoints:

No Active Breakpoints.

Column data per node:

...er_3.DSLink10 - Node 1 | ...er_3.DSLink10 - Node 2

```
Transformer_3
Transformer_3.DSLink10
    name = 'ABCDEFG'
    number = 6
```

Watch List:

Name	Stage/Link

Close the window and go to the menu option "debug". In that select "Clear All Breakpoints". This will remove all breakpoints that you have set in your job.

DATASTAGE COMMON ERRORS/WARNINGS AND SOLUTIONS – 3

1. While connecting "Remote Desktop", Terminal server has been exceeded maximum number of allowed connections

SOL: In Command Prompt, type mstsc /v: ip address of server /admin
OR mstsc /v: ip address /console
2. SQL20521N. Error occurred processing a conditional compilation directive near *string*. Reason code=*rc*.
Following link has issue description:
http://pic.dhe.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=%2Fcom.ibm.db2.luw.messages.sql.doc%2Fdoc%2Fmsql20521n.html
3. SK_RETAILER_GROUP_BRDIGE,1: runLocally() did not reach EOF on its input data set 0.
SOL: Warning will be disappeared by regenerating SK File.
4. While connecting to Datastage client, there is no response, and while restarting websphere services, following errors occurred

```
[root@poluloro01 bin]# ./stopServer.sh  server1 -user wasadmin -password Wasadmin0708
ADMU0116I: Tool information is being logged in file
           /opt/ibm/WebSphere/AppServer/profiles/default/logs/server1/stopServer.log
ADMU0128I: Starting tool with the default profile
ADMU3100I: Reading configuration for server: server1
ADMU0111E: Program exiting with error: javax.management.JMRuntimeException:
ADMN0022E: Access is denied for the stop operation on Server MBean because of insufficient or empty credentials.
ADMU4113E: Verify that username and password information is on the command line
           (-username and -password) or in the <conntype>.client.props file.
ADMU1211I: To obtain a full trace of the failure, use the -trace option.
```

ADMU0211I: Error details may be seen in the file:
/opt/ibm/WebSphere/AppServer/profiles/default/logs/server1/stopServer.log

SOL: Wasadmin and XMeta passwords needs to be reset and commands are below..

```
[root@poluloro01 bin]# cd
/opt/ibm/InformationServer/ASBServer/bin/
[root@poluloro01 bin]# ./AppServerAdmin.sh -was -user
wasadmin
-password Wasadmin0708
Info WAS instance /Node:poluloro01/Server:server1/
updated with new user information
Info MetadataServer daemon script updated with new user
information
[root@poluloro01 bin]# ./AppServerAdmin.sh -was -user
xmeta -password Xmeta0708
Info WAS instance /Node:poluloro01/Server:server1/
updated with new user information
Info MetadataServer daemon script updated with new user
information
```

5. “The specified field doesn’t exist in view adapted schema”

SOL: Most of the time “The specified field: XXXXXX does not exist in the view adapted schema” occurred when we missed a field to map. Every stage has got an output tab if used in the between of the job. Make sure you have mapped every single field required for the next stage. Sometime even after mapping the fields this error can be occurred and one of the reason could be that the view adapter has not linked the input and output fields. Hence in this case the required field mapping should be dropped and recreated.

Just to give an insight on this, the view adapter is an operator which is responsible for mapping the input and output fields. Hence DataStage creates an instance of APT_ViewAdapter which translate the components of the operator input interface schema to matching components of the interface schema. So if the interface schema is not having the same columns as operator input interface schema then this error will be reported.

DATASTAGE COMMON ERRORS/WARNINGS AND SOLUTIONS – 2

1. No jobs or logs showing in IBM DataStage Director Client, however jobs are still accessible from the Designer Client.

SOL: SyncProject cmd that is installed with DataStage 8.5 can be run to analyze and recover projects

SyncProject -ISFile islogin -project dstage3 dstage5 –Fix

2. CASHOUT_DTL: Invalid property value /Connection/Database (CC_StringProperty::getValue, file CC_StringProperty.cpp, line 104)

SOL: Change the Data Connection properties manually in the produced

DB2 Connector stage.

A patch fix is available for this issue JR35643

3. Import .dsx file from command line

SOL: DSXImportService -ISFile dataconnection –DSProject dstage –DSXFile c:\export\oldproject.dsx

4. Generate Surrogate Key without Surrogate Key Stage

SOL: @PARTITIONNUM + (@NUMPARTITIONS * (@INROWNUM – 1)) + 1

Use above Formula in Transformer stage to generate a surrogate key.

5. Failed to authenticate the current user against the selected Domain: Could not connect to server.

RC: Client has invalid entry in host file

Server listening port might be blocked by a firewall

Server is down

SOL: Update the host file on client system so that the server hostname can be resolved from client.

Make sure the WebSphere TCP/IP ports are opened by the firewall.

Make sure the WebSphere application server is running. (OR)

Restart Websphere services.

6. The connection was refused or the RPC daemon is not running (81016)

RC: The dsprcd process must be running in order to be able to login to DataStage.

If you restart DataStage, but the socket used by the dsrpcd (default is 31538) was busy, the dsrpcd will fail to start. The socket may be held by dsapi_slave processes that were still running or recently killed when DataStage was restarted.

SOL: Run "ps -ef | grep dsrpcd" to confirm the dsrpcd process is not running.

Run "ps -ef | grep dsapi_slave" to check if any dsapi_slave processes exist. If so, kill them.

Run "netstat -a | grep dsprc" to see if any processes have sockets that are ESTABLISHED, FIN_WAIT, or CLOSE_WAIT. These will prevent the dsprcd from starting. The sockets with status FIN_WAIT or CLOSE_WAIT will eventually time out and disappear, allowing you to restart DataStage.

Then Restart DSEngine.   (if above doesn't work) Needs to reboot the system.

7. To save Datastage logs in notepad or readable format

SOL: a) /opt/ibm/InformationServer/server/DSEngine (go to this directory)

./bin/dsjob -logdetail project_name job_name >/home/dsadm/log.txt

b) In director client, Project tab à Print à select print to file option and save it in local directory.

8. "Run time error '457′. This Key is already associated with an element of this collection."

SOL: Needs to rebuild repository objects.

a)   Login to the Administrator client

b)   Select the project

c)   Click on Command

d)   Issue the command ds.tools

e)   Select option '2'

f) Keep clicking next until it finishes.

g) All objects will be updated.

9. To stop the datastage jobs in linux level

SOL: ps –ef | grep dsadm

To Check process id and phantom jobs

Kill -9 process_id

10. To run datastage jobs from command line

SOL: cd /opt/ibm/InformationServer/server/DSEngine

./dsjob -server $server_nm -user $user_nm -password $pwd -run $project_nm $job_nm

11. Failed to connect to JobMonApp on port 13401.

SOL: needs to restart jobmoninit script (in /opt/ibm/InformationServer/Server/PXEngine/Java)

Type sh jobmoninit start $APT_ORCHHOME

Add 127.0.0.1 local host in /etc/hosts file

(Without local entry, Job monitor will be unable to use the ports correctly)

12. SQL0752N. Connect to a database is not permitted within logical unit of work CONNECT type 1 settings is in use.

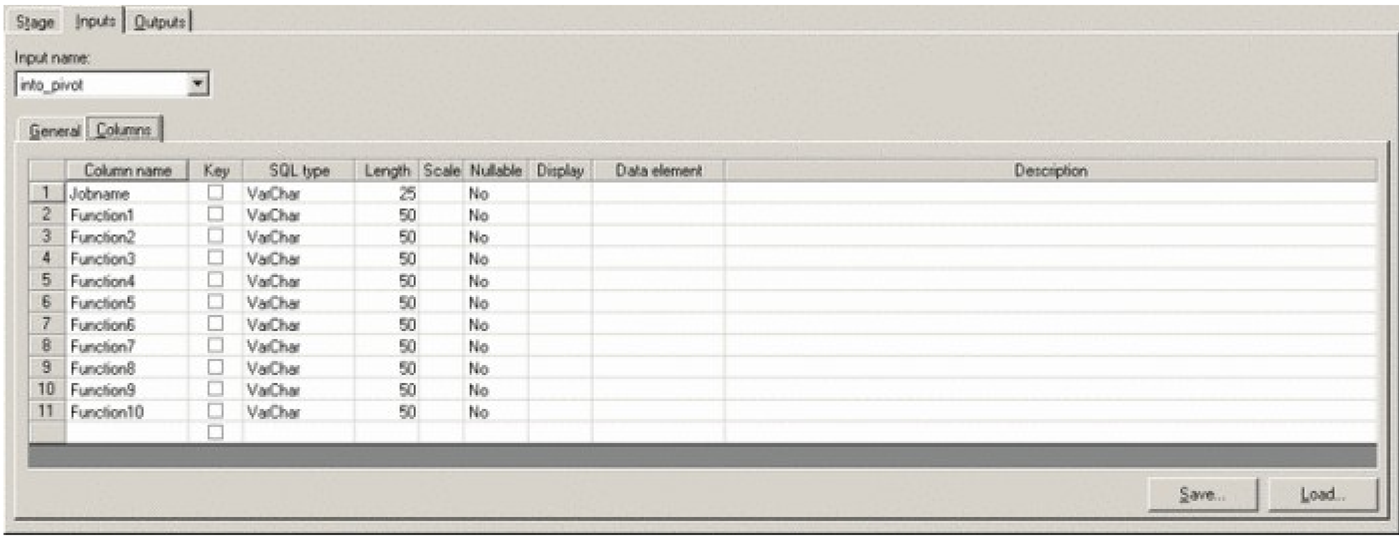
SOL: COMMIT or ROLLBACK statement before requesting connection to another database.

USAGE OF PIVOT STAGE

I couldn't find an example of the DataStage pivot stage and it's not very obvious how to use it. That's why I've created this page. It shows an example of how you use it.

Input link

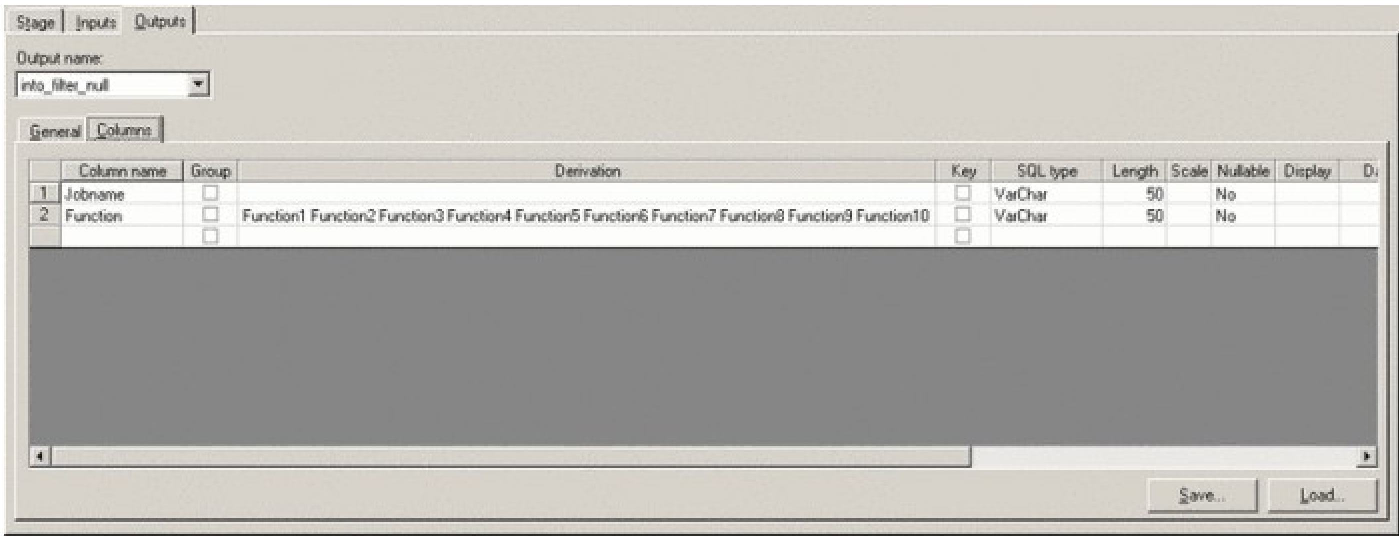
The input link has 11 columns. One for the jobname and 10 for used functions.



	Column name	Key	SQL type	Length	Scale	Nullable	Display	Data element	Description
1	Jobname	☐	VarChar	25		No			
2	Function1	☐	VarChar	50		No			
3	Function2	☐	VarChar	50		No			
4	Function3	☐	VarChar	50		No			
5	Function4	☐	VarChar	50		No			
6	Function5	☐	VarChar	50		No			
7	Function6	☐	VarChar	50		No			
8	Function7	☐	VarChar	50		No			
9	Function8	☐	VarChar	50		No			
10	Function9	☐	VarChar	50		No			
11	Function10	☐	VarChar	50		No			
		☐							

Output link

The output link has two columns. The jobname and one for the function names. Per input row, there are 10 rows outputted. Even if the fields are NULL. Divide the fieldnames with spaces, not with any other delimiter.



	Column name	Group	Derivation	Key	SQL type	Length	Scale	Nullable	Display	D
1	Jobname	☐		☐	VarChar	50		No		
2	Function	☐	Function1 Function2 Function3 Function4 Function5 Function6 Function7 Function8 Function9 Function10	☐	VarChar	50		No		
		☐		☐						

DATASTAGE PARALLEL ROUTINES – 3: TO CHECK IF A GIVEN DATE IS VALID OR NOT

Requirement: To check if a given date is valid or not.

Datastage has a built in function (IsValid) which can check if a date is valid or not, syntax to this is

```
IsValid('DATE', StringToDate (<Date String>,"%mm/%dd/%yyyy"))
```

For example if we have to test for string 01/01/1999 the syntax will be

```
IsValid('DATE', StringToDate("01/01/1999","%mm/%dd/%yyyy"))
```

If the date is valid date, it will return one else zero. But, if we have the date as “1/1/1999″ though the date is valid it will return 0, as date and month fields have only one digit. Therefore, we will create a Parallel Routine, which will take date in various format viz. mm/dd/yyyy, m/dd/yyyy, mm/d/yyyy, mm/dd/yy, m/d/yy etc… and return it in a fixed format of mmddyyyy, which we can use in the function specified above.

C/C++ Code:

The code written below takes a date string in any of the formats mentioned above and returns a date string in mmddyyyy format (if the year is provided in two digits ‘yy’ then it will return ’19nn’ if nn>=50 and ’20nn’ if nn<50 as the year part).

```
//FileName: date_format.cpp

#include <stdio.h>

#include <string.h>

#include<stdlib.h>



char* date_format ( char str[20])

{

    char * fst,*lst,date[10];

    int len,decade;

    len=strlen(str);

    fst=strchr(str,'/');

    lst=strrchr(str,'/');
```

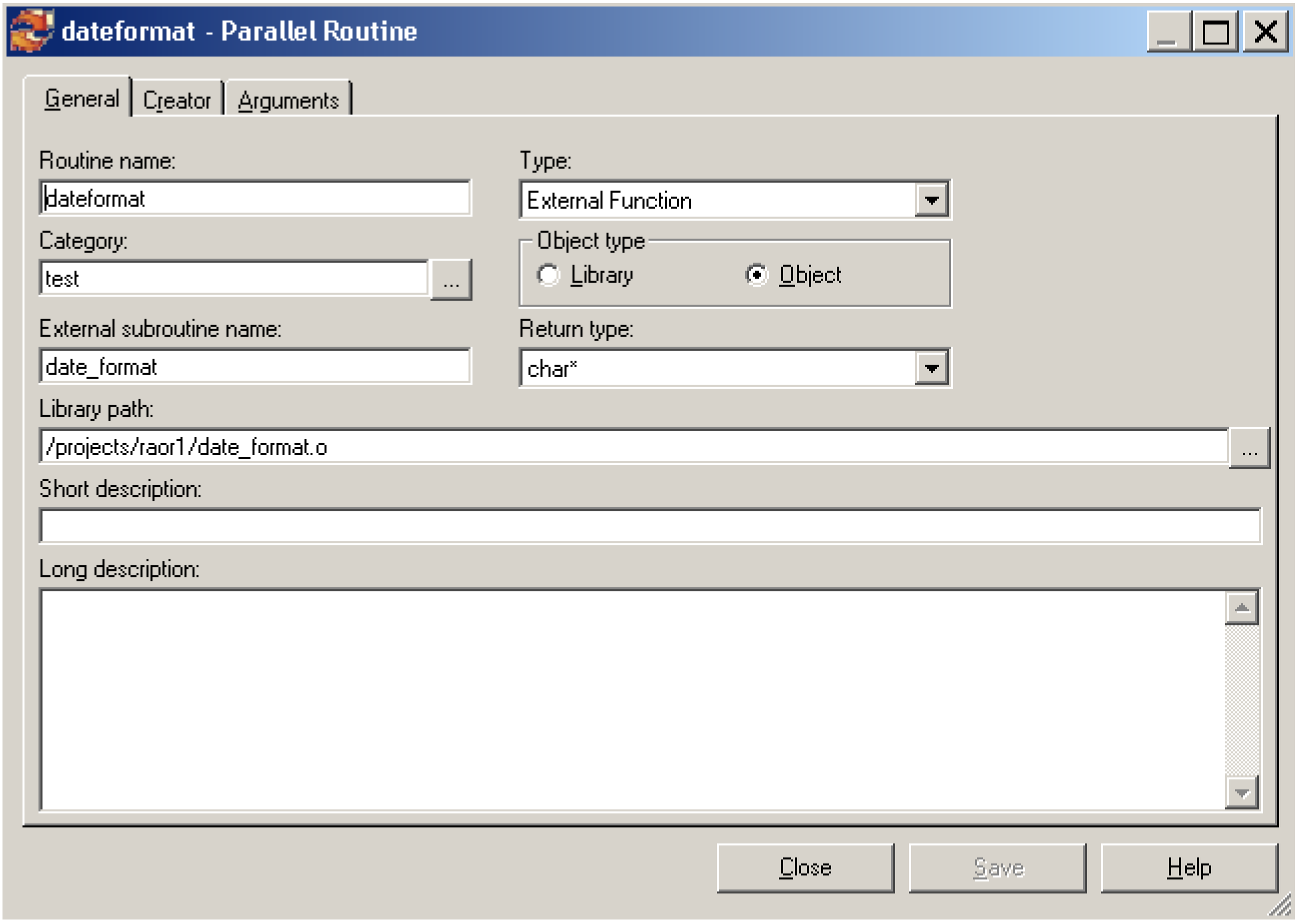
```
//handling the date part

if ((fst-str)==1)

{

    strcpy(date,"0");

    strncat(date,str,1);

}

else if ((fst-str)==2)

{

    strncpy(date,str,2);

}

else

    return ("Invalid");



//handling the month part

if ((lst-fst-1)==1)

{

    strcat(date,"0");

    strncat(date,fst+1,1);

}

else if ((lst-fst-1)==2)

{

    strncat(date,fst+1,2);
}

else

    return ("Invalid");


//handling the year part

if ((len-(lst-str)-1)==2)

{

    decade=atoi(lst+1);

    if (decade>=50)

    {

        strcat(date,"19");

    }

    else

    {

        strcat(date,"20");

    }

    strncat(date,lst+1,2);

    }

else if ((len-(lst-str)-1)==4)

    {

        strncat(date,lst+1,4);

    }
    else

    return ("Invalid");

    return(date);

}
```

Create the .o file (date_format.o) for the above code by following the steps mentioned in this post.

Creating a Parallel Routine

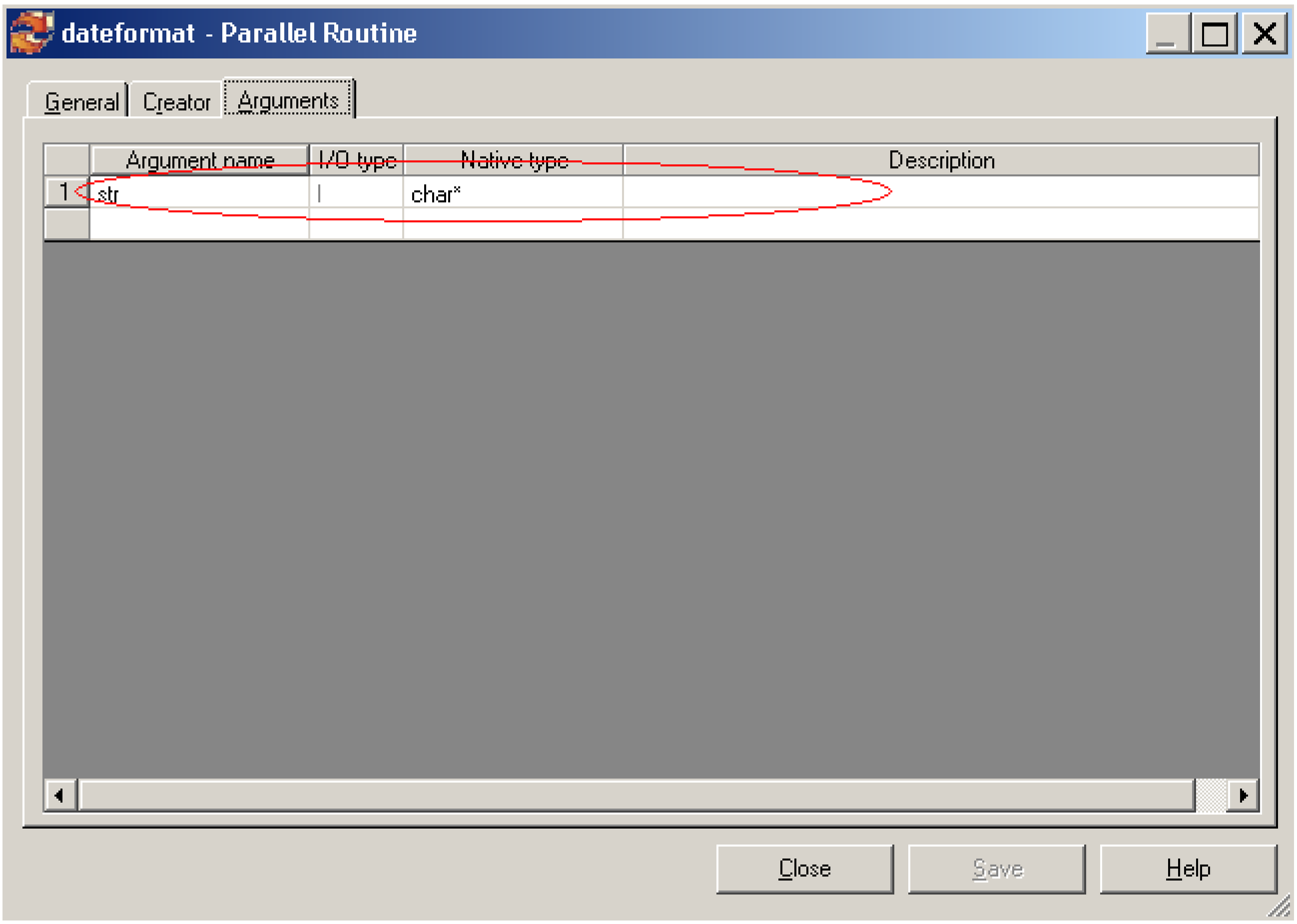
Open a Parallel Routine and give appropriate values, a screen shot is shown below for reference

General tab for "dateformat" parallel routine



Routine name: dateformat
Type: External Function
Category: test
Object type: Library [✓ Object]
External subroutine name: date_format
Return type: char*
Library path: /projects/raor1/date_format.o
Short description:
Long description:

Arguments tab for "dateformat" parallel routine.



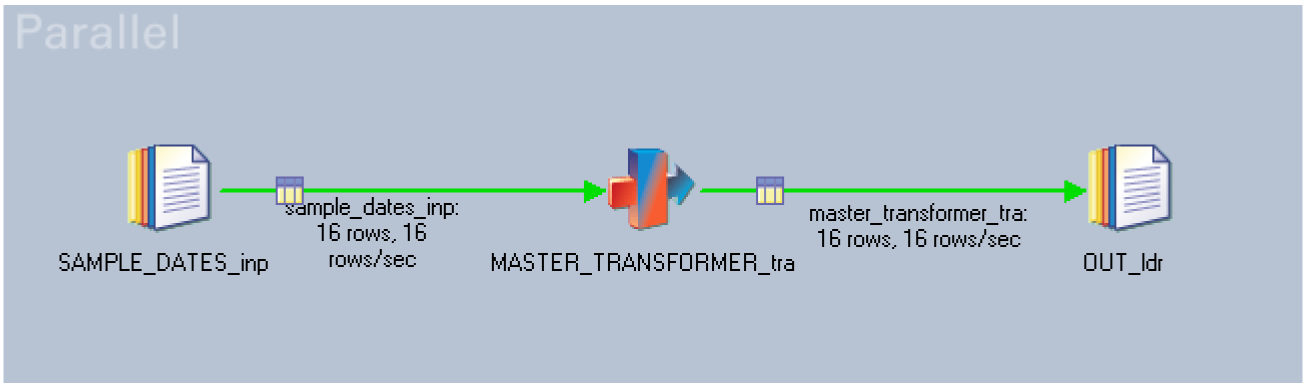
	Argument name	I/O type	Native type	Description
1	str	I	char*	

As we are passing a date string as input we have added a parameter here.

Save and Close it.

Usage in Datastage Job

We have created a simple job to demonstrate of the Parallel Routine "dateformat" which we have just created.

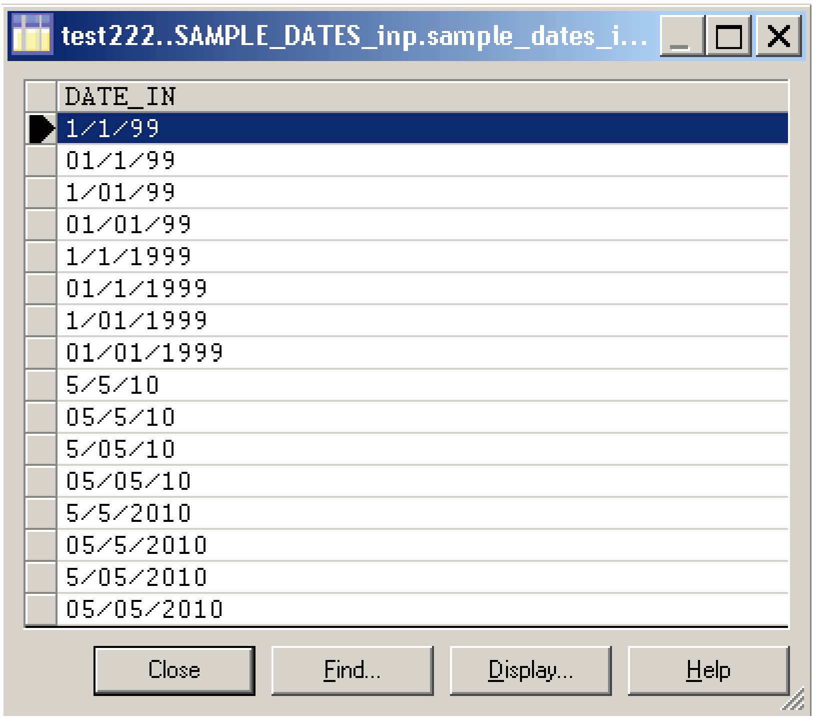

Parallel
SAMPLE_DATES_inp
sample_dates_inp: 16 rows, 16 rows/sec
MASTER_TRANSFORMER_tra
master_transformer_tra: 16 rows, 16 rows/sec
OUT_ldr

The job has three stages:

1. **SAMPLE_DATES_inp stage provides sample dates for testing.**

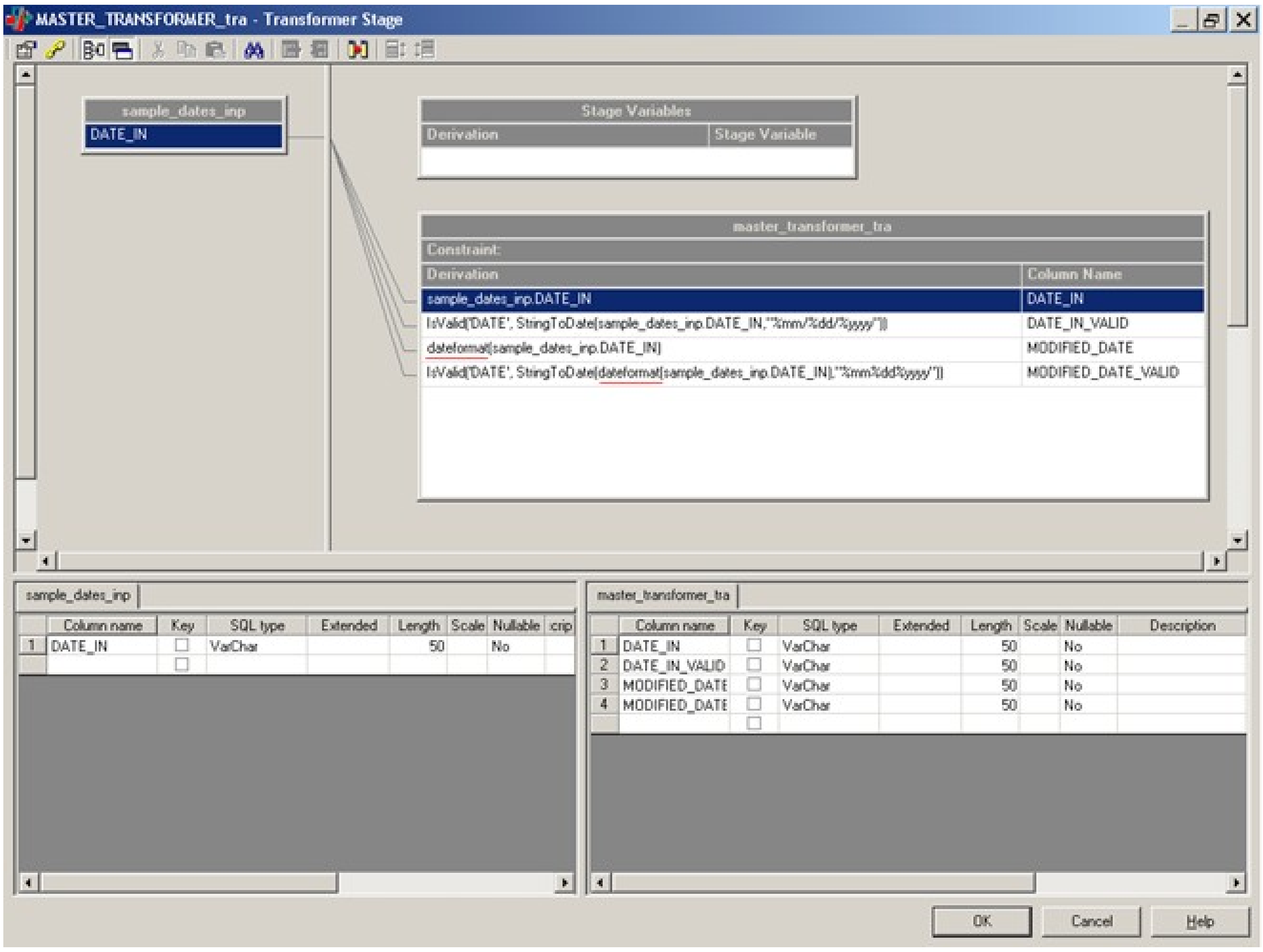
2. **MASTER_TRANSFORMER_tra stage calls the parallel routine.**
3. **OUT_ldr loads the data into a tab delimited file.**

The figure shows the sample data in SAMPLE_DATES_inp


test222..SAMPLE_DATES_inp.sample_dates_i...

DATE_IN
1/1/99
01/1/99
1/01/99
01/01/99
1/1/1999
01/1/1999
1/01/1999
01/01/1999
5/5/10
05/5/10
5/05/10
05/05/10
5/5/2010
05/5/2010
5/05/2010
05/05/2010

The figure shows the properties of the MASTER_TRANSFORMER_tra stage


MASTER_TRANSFORMER_tra - Transformer Stage

sample_dates_inp

DATE_IN

Stage Variables

Derivation	Stage Variable

master_transformer_tra

Constraint:

Derivation	Column Name
sample_dates_inp.DATE_IN	DATE_IN
IsValid('DATE', StringToDate(sample_dates_inp.DATE_IN,"%mm/%dd/%yyyy"))	DATE_IN_VALID
dateformat(sample_dates_inp.DATE_IN)	MODIFIED_DATE
IsValid('DATE', StringToDate(dateformat(sample_dates_inp.DATE_IN),"%mm%dd%yyyy"))	MODIFIED_DATE_VALID

sample_dates_inp

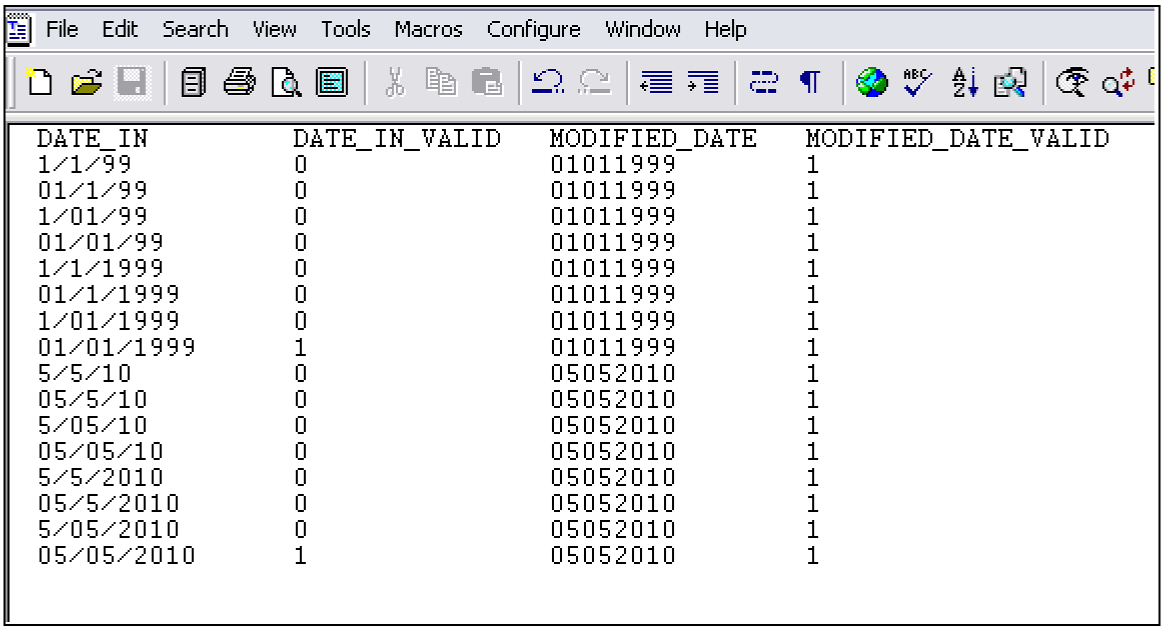
	Column name	Key	SQL type	Extended	Length	Scale	Nullable	crip
1	DATE_IN	☐	VarChar		50		No	
		☐						

master_transformer_tra

	Column name	Key	SQL type	Extended	Length	Scale	Nullable	Description
1	DATE_IN	☐	VarChar		50		No	
2	DATE_IN_VALID	☐	VarChar		50		No	
3	MODIFIED_DATE	☐	VarChar		50		No	
4	MODIFIED_DATE	☐	VarChar		50		No	
		☐						

Various transformation rules are clearly seen, the place where "dateformat" is called is underlined in red color

The figure shows the data loaded in the output file.



```
DATE_IN          DATE_IN_VALID    MODIFIED_DATE    MODIFIED_DATE_VALID
1/1/99           0                01011999         1
01/1/99          0                01011999         1
1/01/99          0                01011999         1
01/01/99         0                01011999         1
1/1/1999         0                01011999         1
01/1/1999        0                01011999         1
1/01/1999        0                01011999         1
01/01/1999       1                01011999         1
5/5/10           0                05052010         1
05/5/10          0                05052010         1
5/05/10          0                05052010         1
05/05/10         0                05052010         1
5/5/2010         0                05052010         1
05/5/2010        0                05052010         1
5/05/2010        0                05052010         1
05/05/2010       1                05052010         1
```

DATE_IN: this column consists of all the records from the input file as it is.

```
sample_dates_inp.DATE_IN
```

DATE_VALID: this column show the value returned by the IS_VALID function without using the parallel routine “dateformat” created by us.

```
IsValid('DATE', StringToDate(sample_dates_inp.DATE_IN,"%mm/%dd/%yyyy"))
```

MODIFIED_DATE: this column shows the date returned by routine “dateformat” when DATE_IN is given as the input.

```
dateformat(sample_dates_inp.DATE_IN)
```

MODIFIED_DATE_VALID: this column show the value returned by the IS_VALID function AFTER using the parallel routine “dateformat” created by us.

```
IsValid('DATE', StringToDate(dateformat(sample_dates_inp.DATE_IN),"%mm%dd
%yyyy"))
```

We can see that all date formats have been identified as valid dates after using the routine.

Let us see one more usage of Datastage parallel routines in my next post.

DATASTAGE COMMON ERRORS/WARNINGS AND SOLUTIONS – 1

1. While running ./NodeAgents.sh start command… getting the following error: “LoggingAgent.sh process stopped unexpectedly”

SOL: needs to kill LoggingAgentSocketImpl
Ps -ef | grep LoggingAgentSocketImpl (OR)
PS -ef | grep Agent (to check the process id of the above)

2. Warning: A sequential operator cannot preserve the partitioning of input data set on input port 0

SOL: Clear the preserve partition flag before Sequential file stages.

3. Warning: A user defined sort operator does not satisfy the requirements.

SOL: Check the order of sorting columns and make sure use the same order when use join stage after sort to joing two inputs.

4. Conversion error calling conversion routine timestamp_from_string data may have been lost. xfmJournals,1: Conversion error calling conversion routine decimal_from_string data may have been lost

SOL: check for the correct date format or decimal format and also null values in the date or decimal fields before passing to datastage StringToDate, DateToString,DecimalToString or StringToDecimal functions.

5. To display all the jobs in command line

SOL:

```
cd /opt/ibm/InformationServer/Server/DSEngine/bin

./dsjob -ljobs <project_name>
```

6. “Error trying to query dsadm[]. There might be an issue in database server”

SOL: Check XMETA connectivity.

db2 connect to xmeta (A connection to or activation of database “xmeta” cannot be made because of BACKUP pending)

7. “DSR_ADMIN: Unable to find the new project location”

SOL: Template.ini file might be missing in /opt/ibm/InformationServer/Server.
Copy the file from another severs.

8. “Designer LOCKS UP while trying to open any stage”

SOL: Double click on the stage that locks up datastage
Press ALT+SPACE
Windows menu will popup and select Restore
It will show your properties window now
Click on “X” to close this window.
Now, double click again and try whether properties window appears.

9. “Error Setting up internal communications (fifo RT_SCTEMP/job_name.fifo)

SOL: Remove the locks and try to run (OR)
Restart DSEngine and try to run (OR)
Go to /opt/ibm/InformationServer/server/Projects/proj_name/

```
ls RT_SCT* then
rm -f  RT_SCTEMP
then try to restart it.
```

10. While attempting to compile job, "failed to invoke GenRunTime using Phantom process helper"

RC: /tmp space might be full
Job status is incorrect
Format problems with projects uvodbc.config file
SOL: a) clean up /tmp directory
b) DS Director à JOB à clear status file
c) confirm uvodbc.config has the following entry/format:

```
[ODBC SOURCES]
<local uv>
DBMSTYPE = UNIVERSE
Network  = TCP/IP
Service =  uvserver
Host = 127.0.0.1
```

DATASTAGE ERROR CODES

CODE	ERROR TOKEN	DESCRIPTION
0	DSJE_NOERROR	No InfoSphere DataStage API error has occurred.
-1	DSJE_BADHANDLE	Invalid *JobHandle.*
-2	DSJE_BADSTATE	Job is not in the right state (compiled, not running).
-3	DSJE_BADPARAM	*ParamName* is not a parameter name in the job.
-4	DSJE_BADVALUE	Invalid *MaxNumber* value.
-5	DSJE_BADTYPE	Information or event type was unrecognized.
-6	DSJE_WRONGJOB	Job for this *JobHandle* was not started from a call to **DSRunJob** by the

CODE	ERROR TOKEN	DESCRIPTION
0	DSJE_NOERROR	No InfoSphere DataStage API error has occurred.
		current process.
-7	DSJE_BADSTAGE	*StageName* does not refer to a known stage in the job.
-8	DSJE_NOTINSTAGE	Internal engine error.
-9	DSJE_BADLINK	*LinkName* does not refer to a known link for the stage in question.
-10	DSJE_JOBLOCKED	The job is locked by another process.
-11	DSJE_JOBDELETED	The job has been deleted.
-12	DSJE_BADNAME	Invalid project name.
-13	DSJE_BADTIME	Invalid *StartTime* or *EndTime* value.
-14	DSJE_TIMEOUT	The job appears not to have started after waiting for a reasonable length of time. (About 30 minutes.)
-15	DSJE_DECRYPTERR	Failed to decrypt encrypted values.
-16	DSJE_NOACCESS	Cannot get values, default values or design default values for any job except the current job.
-99	DSJE_REPERROR	General engine error.
-100	DSJE_NOTADMINUSER	User is not an administrator.
-101	DSJE_ISADMINFAILED	Failed to determine whether user is an administrator.
-102	DSJE_READPROJPROPERTY	Failed to read property.
-103	DSJE_WRITEPROJPROPERTY	Property not supported.
-104	DSJE_BADPROPERTY	Unknown property name.
-105	DSJE_PROPNOTSUPPORTED	Unsupported property.
-106	DSJE_BADPROPVALUE	Invalid value for this property.
-107	DSJE_OSHVISIBLEFLAG	Failed to get value for OSHVisible.
-108	DSJE_BADENVVARNAME	Invalid environment variable name.

CODE	ERROR TOKEN	DESCRIPTION
0	DSJE_NOERROR	No InfoSphere DataStage API error has occurred.
-109	DSJE_BADENVVARTYPE	Invalid environment variable type.
-110	DSJE_BADENVVARPROMPT	No prompt supplied.
-111	DSJE_READENVVARDEFNS	Failed to read environment variable definitions.
-112	DSJE_READENVVARVALUES	Failed to read environment variable values.
-113	DSJE_WRITEENVVARDEFNS	Failed to write environment variable definitions.
-114	DSJE_WRITEENVVARVALUES	Failed to write environment variable values.
-115	DSJE_DUPENVVARNAME	Environment variable being added already exists.
-116	DSJE_BADENVVAR	Environment variable does not exist.
-117	DSJE_NOTUSERDEFINED	Environment variable is not user-defined and therefore cannot be deleted.
-118	DSJE_BADBOOLEANVALUE	Invalid value given for a boolean environment variable.
-119	DSJE_BADNUMERICVALUE	Invalid value given for an integer environment variable.
-120	DSJE_BADLISTVALUE	Invalid value given for a list environment variable.
-121	DSJE_PXNOTINSTALLED	Environment variable is specific to parallel jobs which are not available.
-122	DSJE_ISPARALLELLICENCED	Failed to determine if parallel jobs are available.
-123	DSJE_ENCODEFAILED	Failed to encode an encrypted value.
-124	DSJE_DELPROJFAILED	Failed to delete project definition.
-125	DSJE_DELPROJFILESFAILED	Failed to delete project files.
-126	DSJE_LISTSCHEDULEFAILED	Failed to get list of scheduled jobs for project.

CODE	ERROR TOKEN	DESCRIPTION
0	DSJE_NOERROR	No InfoSphere DataStage API error has occurred.
-127	DSJE_CLEARSCHEDULEFAILED	Failed to clear scheduled jobs for project.
-128	DSJE_BADPROJNAME	Invalid project name supplied.
-129	DSJE_GETDEFAULTPATHFAILED	Failed to determine default project directory.
-130	DSJE_BADPROJLOCATION	Invalid path name supplied.
-131	DSJE_INVALIDPROJECTLOCATION	Invalid path name supplied.
-132	DSJE_OPENFAILED	Failed to open UV.ACCOUNT file.
-133	DSJE_READUFAILED	Failed to lock project create lock record.
-134	DSJE_ADDPROJECTBLOCKED	Another user is adding a project.
-135	DSJE_ADDPROJECTFAILED	Failed to add project.
-136	DSJE_LICENSEPROJECTFAILED	Failed to license project.
-137	DSJE_RELEASEFAILED	Failed to release project create lock record.
-138	DSJE_DELETEPROJECTBLOCKED	Project locked by another user.
-139	DSJE_NOTAPROJECT	Failed to log to project.
-140	DSJE_ACCOUNTPATHFAILED	Failed to get account path.
-141	DSJE_LOGTOFAILED	Failed to log to UV account.
-201	DSJE_UNKNOWN_JOBNAME	The supplied job name cannot be found in the project.
-1001	DSJE_NOMORE	All events matching the filter criteria have been returned.
-1002	DSJE_BADPROJECT	*ProjectName* is not a known InfoSphere DataStage project.
-1003	DSJE_NO_DATASTAGE	InfoSphere DataStage is not installed on the system.
-1004	DSJE_OPENFAIL	The attempt to open the job failed – perhaps it has not been compiled.

CODE	ERROR TOKEN	DESCRIPTION
0	DSJE_NOERROR	No InfoSphere DataStage API error has occurred.
-1005	DSJE_NO_MEMORY	Failed to allocate dynamic memory.
-1006	DSJE_SERVER_ERROR	An unexpected or unknown error occurred in the engine.
-1007	DSJE_NOT_AVAILABLE	The requested information was not found.
-1008	DSJE_BAD_VERSION	The engine does not support this version of the InfoSphere DataStage API.
-1009	DSJE_INCOMPATIBLE_SERVER	The engine version is incompatible with this version of the InfoSphere DataStage API.

API error codes in numerical order

ERROR NUMBER	DESCRIPTION
39121	The InfoSphere DataStage license has expired.
39134	The InfoSphere DataStage user limit has been reached.
80011	Incorrect system name or invalid user name or password provided.
80019	Password has expired.

API Communication Layer Error Codes

IBM REDBOOK CORNER

Below are a list of redbooks that are available on the IBM site for topics related to Datastage. Its pretty useful for the person who wants to know more. Check it out

Information Server Installation and Configuration Guide

This IBM redbook publication provides suggestions, hints and tips, directions, installation steps, checklists of

prerequisites, and configuration information collected from a number of Information Server experts. It is intended to minimize the time required to successfully install and configure Information Server.

http://www.redbooks.ibm.com/abstracts/redp4596.html?Open

Preparing for DB2 Near-Realtime Business Intelligence

This IBM redbook is primarily intended for use by IBM Clients and IBM Business Partners. The purpose of the redbook is to discuss the topic of preparing for near-realtime business intelligence (BI). In addition, it is to provide information that will help convince you that DB2®, and other IBM and IBM Business Partner products, can be used today to implement a near-realtime BI environment.

http://www.redbooks.ibm.com/abstracts/sg246071.html?Open

Dimensional Modeling: In a Business Intelligence Environment

A redbook on BI and what it means today, with good concepts on data modelling

http://www.redbooks.ibm.com/abstracts/sg247138.html?Open

SOA Solutions Using IBM Information Server

This IBM® Redbooks® publication documents the procedures for implementing IBM Information Server and related technologies using a typical financial services business scenario. It is aimed at IT architects, Information Management specialists, and Information Integration specialists responsible for developing IBM Information Server on a Microsoft® Windows® 2003 platform.

http://www.redbooks.ibm.com/abstracts/sg247402.html?Open

IBM InfoSphere DataStage Data Flow and Job Design

This IBM® Redbooks® publication documents the procedures for implementing IBM InfoSphere™

DataStage® and related technologies using a typical retail industry scenario. It is aimed at IT architects, Information Management specialists, and Information Integration specialists responsible for developing IBM Information Server on a Microsoft® Windows® 2003 platform.

http://www.redbooks.ibm.com/abstracts/sg247576.html?Open

Deploying a Grid Solution with the IBM InfoSphere Information Server

This IBM® Redbooks® publication documents the migration of an IBM InfoSphere™ Information Server Version 8.0.1 parallel framework implementation on a Red Hat Enterprise Linux® (RHEL) AS 4 platform to a high availability (HA) grid environment.

http://www.redbooks.ibm.com/abstracts/sg247625.html?Open

InfoSphere DataStage Parallel Framework Standard Practices

In this IBM® Redbooks® publication, are present guidelines for the development of highly efficient and scalable information integration applications with InfoSphere™ DataStage® (DS) parallel jobs. InfoSphere DataStage is at the core of IBM Information Server, providing components that yield a high degree of freedom

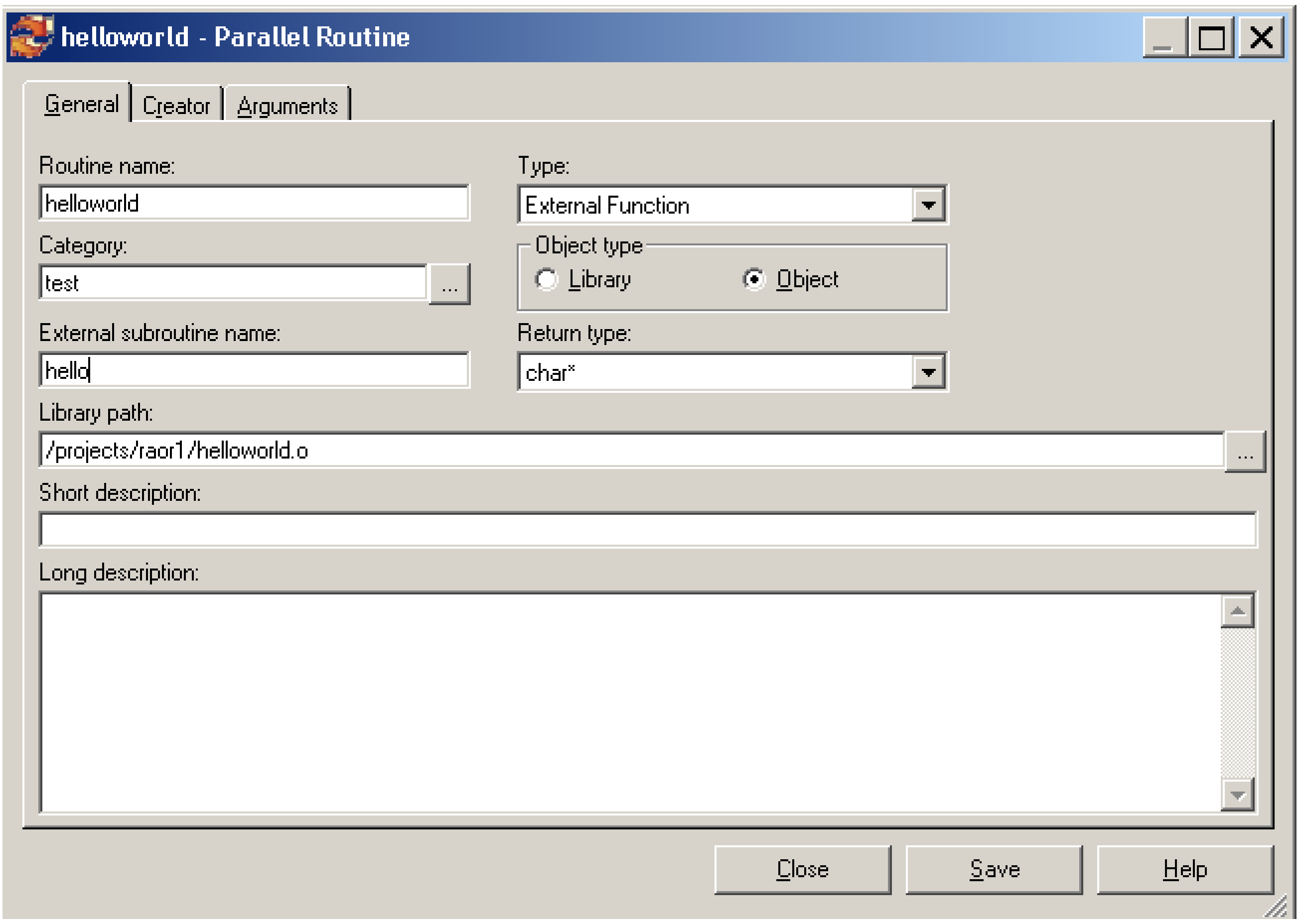
http://www.redbooks.ibm.com/abstracts/sg247830.html?Open

DATASTAGE PARALLEL ROUTINES – 2

In the previous post, we have seen how to write custom C/C++ code. In this post we will see how to call it in DataStage.

We will link the .o file created earlier to the parallel routine in DataStage. From the File menu in the DataStage designer, select parallel routine, it should look something like below.

GENERAL tab of a Parallel Routine


helloworld - Parallel Routine

Routine name: helloworld

Type: External Function

Category: test

Object type: Library [✓ Object]

External subroutine name: hello

Return type: char*

Library path: /projects/raor1/helloworld.o

Short description:

Long description:

Description of various fields in the General tab:

Routine name: The name of the routine as it will appear in the DataStage Repository.

Category: The name of the category where the routine is stored in the Repository. If you do not enter a name in this field when creating a routine, the routine is created under the main Routines branch.

External subroutine name: The C function that this routine is calling (must be the name of a valid routine in a shared library).

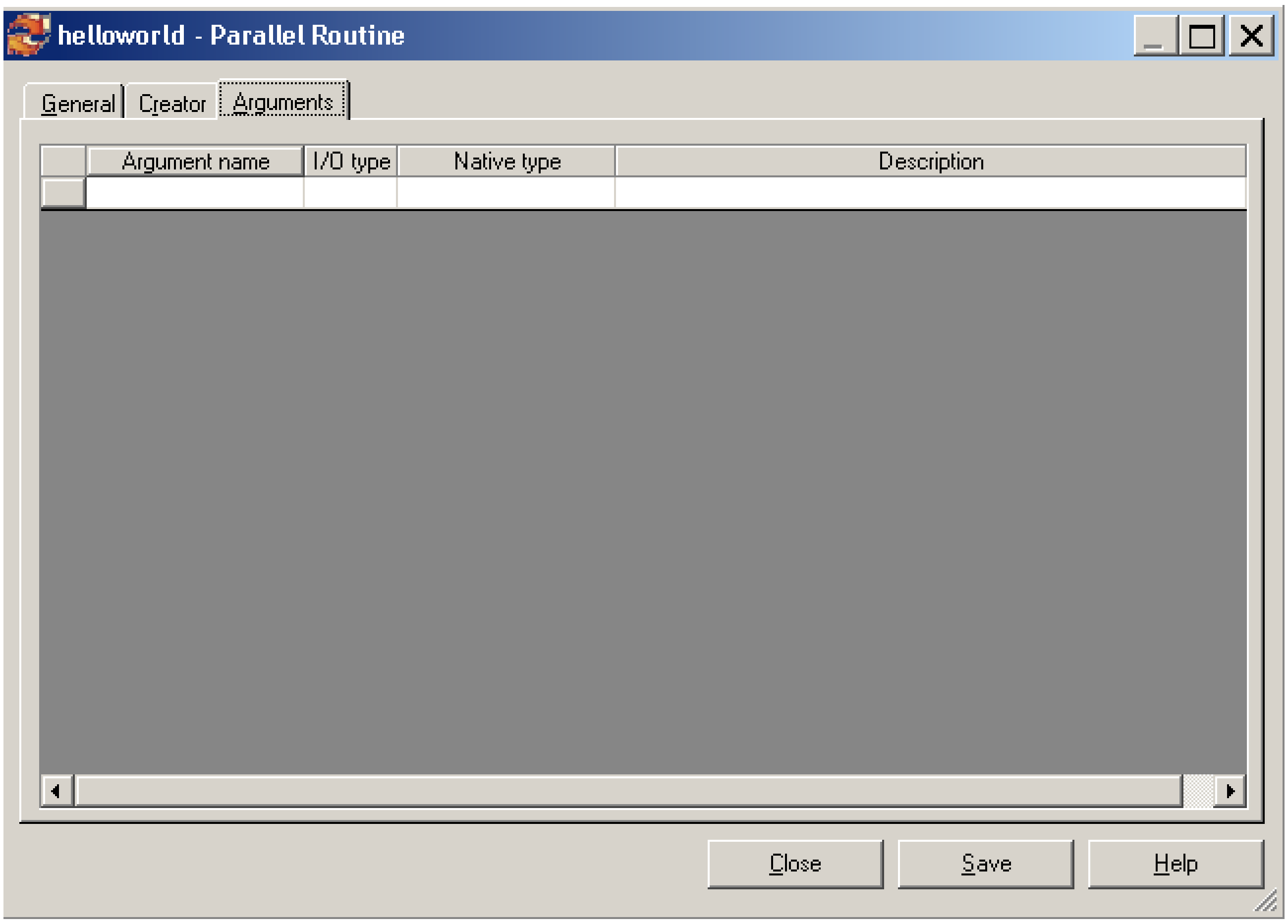
Library Path: The path where the .o file is located.

Type: keep it as External Function.

Object Type: keep it as Object.

Return Type: Choose the type of the value that the function will return.

ARGUMENTS tab of a Parallel Routine



Under the arguments tab enter the names of all the parameters that the function takes as input with their corresponding types. (as our sample code does not take any input parameters leave this blank). Once you have entered the appropriate values save and close the Parallel Routine. Now this routine can be called in Transformer like any other regular function.

DATASTAGE PARALLEL ROUTINES – 1

Sometimes it may be difficult to implement certain functionality in DATASTAGE directly, in such cases we can use parallel routine to build custom functions in C/C++ and call them in datastage. These functions can be called like any other built in datastage function for e.g TRIM(),IntegerToString() etc.

This series of posts explains in detail the process of creating a routine with the help of examples.But before coding anything, please check out the post compiler specification.

Coding in C/C++:

First we will write a C++ code, convert it to .obj file then we will link it to a Parallel Routine in the DataStage.

Sample Code:

Write a code and test it, if it is working fine then replace the main with a new function name

A sample C++ code to display "HELLO WORLD" is shown below

```
#include<stdio.h>

int main()

{

          char * s="hello world"

          printf ("%s",s);

          return 1;

}
```

After writing the code you can compile it in UNIX BOX using the command

```
/usr/vacpp/bin/xlC_r  -O -qspill=32704 <FILE NAME>
```

For e.g if the file name is helloworld.cpp then to compile it you will have to type

```
/usr/vacpp/bin/xlC_r  -O -qspill=32704 helloworld.cpp
```

Once it is successfully compiled u can run it by typing a.out in the UNIX BOX. On running it should display"hello world" on the screen. Now that the code has been tested and is running fine remove main from the code and change its return type to CHAR*, this is because,

this code when called from DataStage can return the string.

After making the changes the above code will look like this:

```
#include<stdio.h>

char * hello ()

{

          char * s="hello world"

          return s;

}
```

Now you will have to create an obj file for the above code that can be done by using the command

```
/usr/vacpp/bin/xlC_r  -O -c -qspill=32704 <FILE NAME>
```

In this case it will be

```
/usr/vacpp/bin/xlC_r  -O -c -qspill=32704 hellworld.cpp
```

If the code is correct the command will create a file named helloworld.o in the same folder.

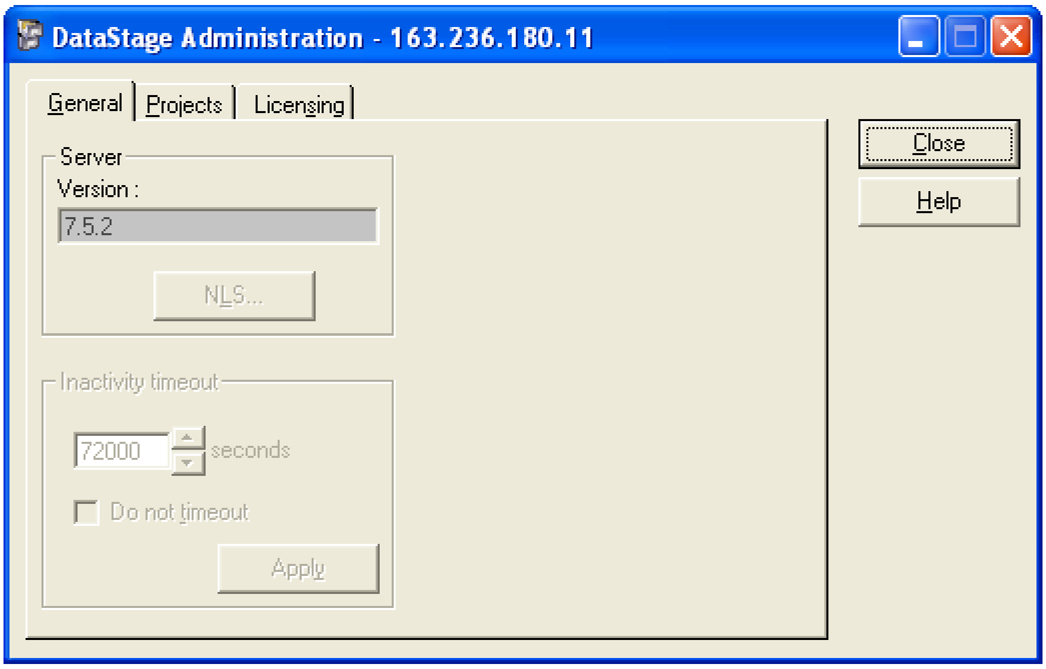
In next post in this series, we will create a DataStage routine and link it to the object code that we developed.

COMPILER SPECIFICATION IN DATASTAGE

One of the pre-requisites to writing Parallel rotines using C/C++ is to check for the compiler specification. This post guides in seeing the C/C++ compiler available in our installation. To find compiler, we need to login into Administrator and go to the path below

DataStage –> Administrator –> Properties –> Environment –> Parallel –> Compiler

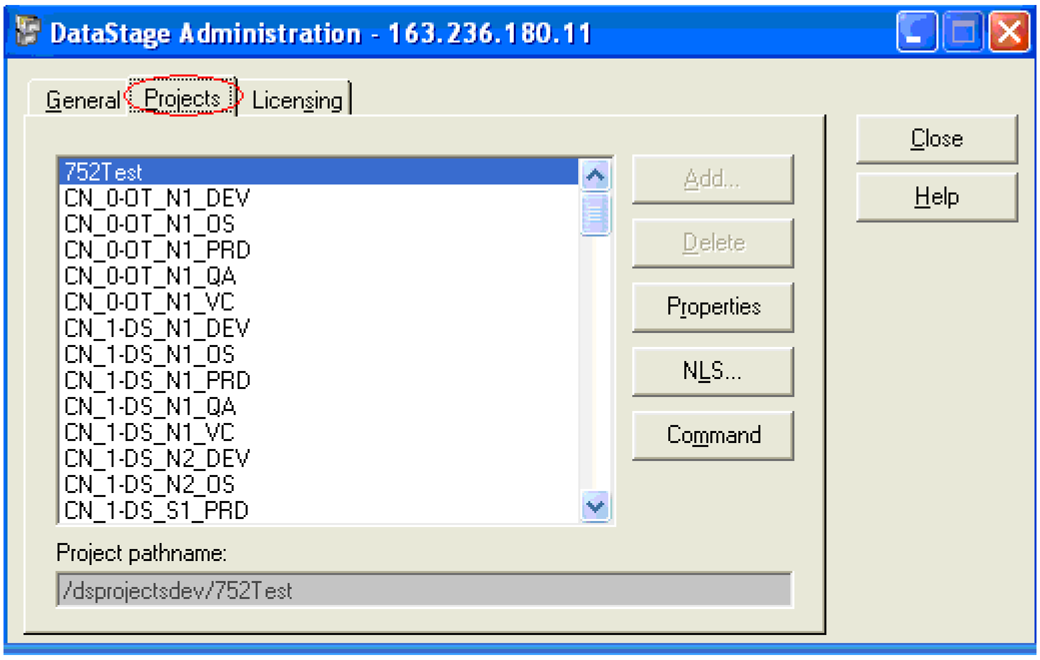
Step 1: Log in to Data Stage Administration.


DataStage Administration - 163.236.180.11

Server
Version : 7.5.2

Inactivity timeout
72000 seconds
Do not timeout

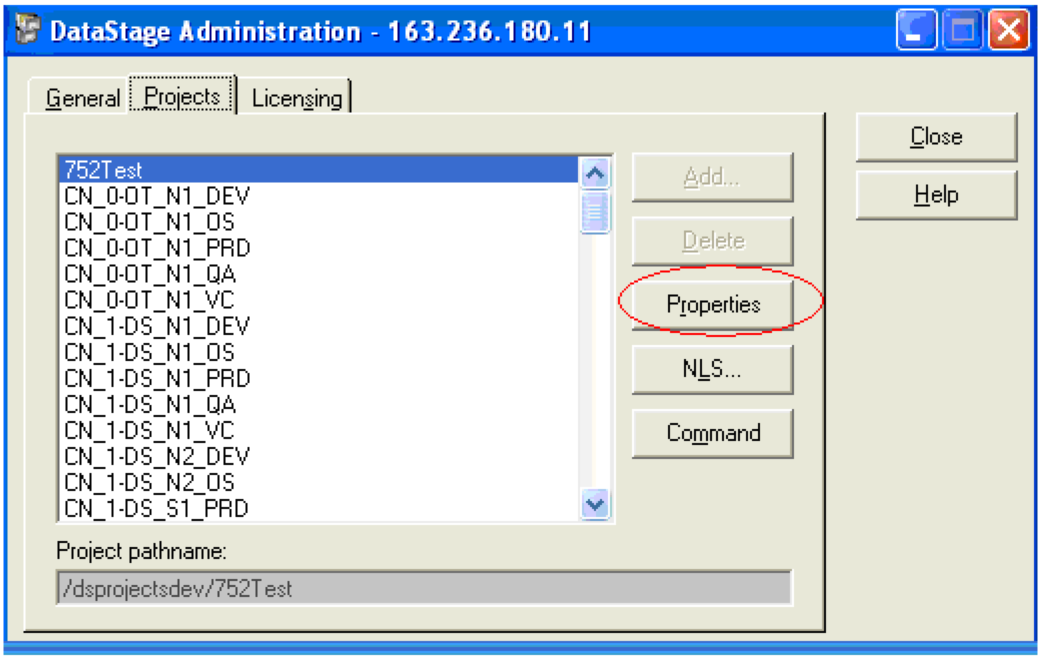
Step 2: Click on the Projects tab.


DataStage Administration - 163.236.180.11

752Test
CN_0-OT_N1_DEV
CN_0-OT_N1_OS
CN_0-OT_N1_PRD
CN_0-OT_N1_QA
CN_0-OT_N1_VC
CN_1-DS_N1_DEV
CN_1-DS_N1_OS
CN_1-DS_N1_PRD
CN_1-DS_N1_QA
CN_1-DS_N1_VC
CN_1-DS_N2_DEV
CN_1-DS_N2_OS
CN_1-DS_S1_PRD

Project pathname:
/dsprojectsdev/752Test

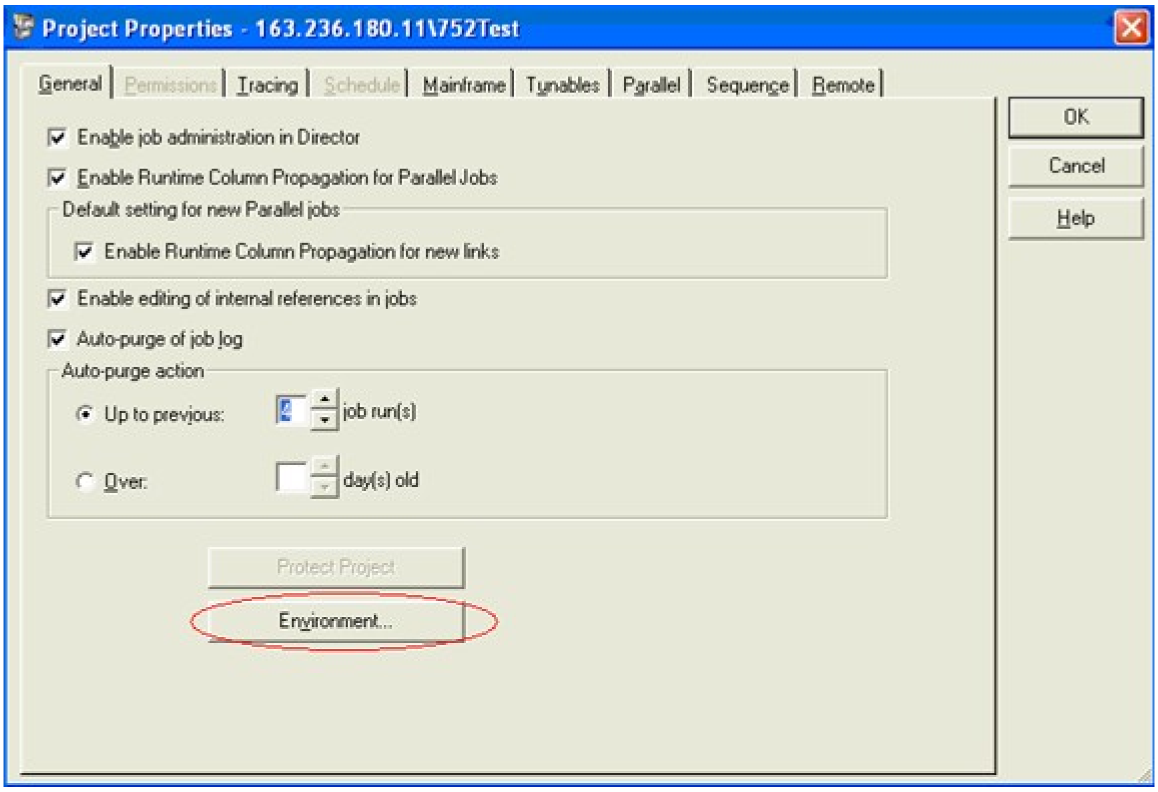
Step 3: Click on the Properties tab



752Test
CN_0-OT_N1_DEV
CN_0-OT_N1_OS
CN_0-OT_N1_PRD
CN_0-OT_N1_QA
CN_0-OT_N1_VC
CN_1-DS_N1_DEV
CN_1-DS_N1_OS
CN_1-DS_N1_PRD
CN_1-DS_N1_QA
CN_1-DS_N1_VC
CN_1-DS_N2_DEV
CN_1-DS_N2_OS
CN_1-DS_S1_PRD

Project pathname: /dsprojectsdev/752Test

Step 4: Click on Environment.



- [✓] Enable job administration in Director
- [✓] Enable Runtime Column Propagation for Parallel Jobs

Default setting for new Parallel jobs

- [✓] Enable Runtime Column Propagation for new links

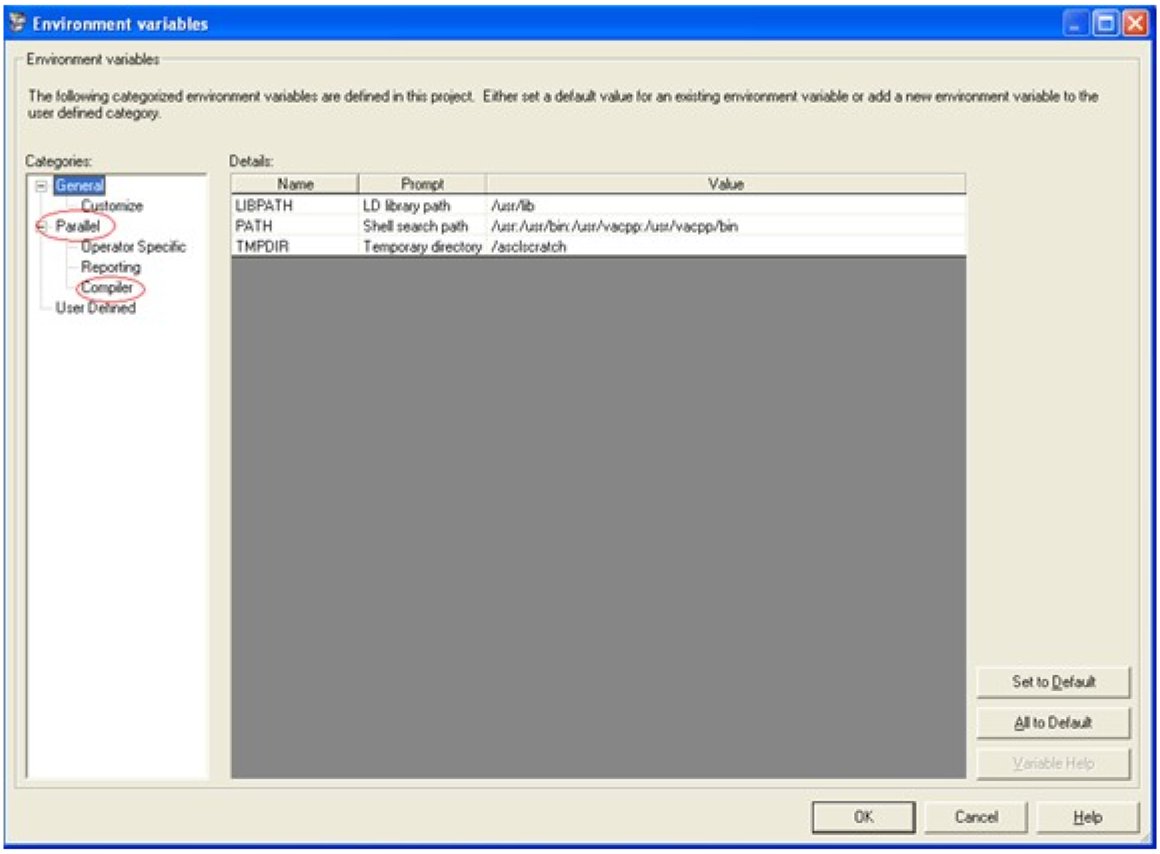
- [✓] Enable editing of internal references in jobs
- [✓] Auto-purge of job log

Auto-purge action

[✓ Up to previous:] 2 job run(s)

Over: day(s) old

Step 5: Select the compiler option under Parallel tab.


Environment variables

Environment variables

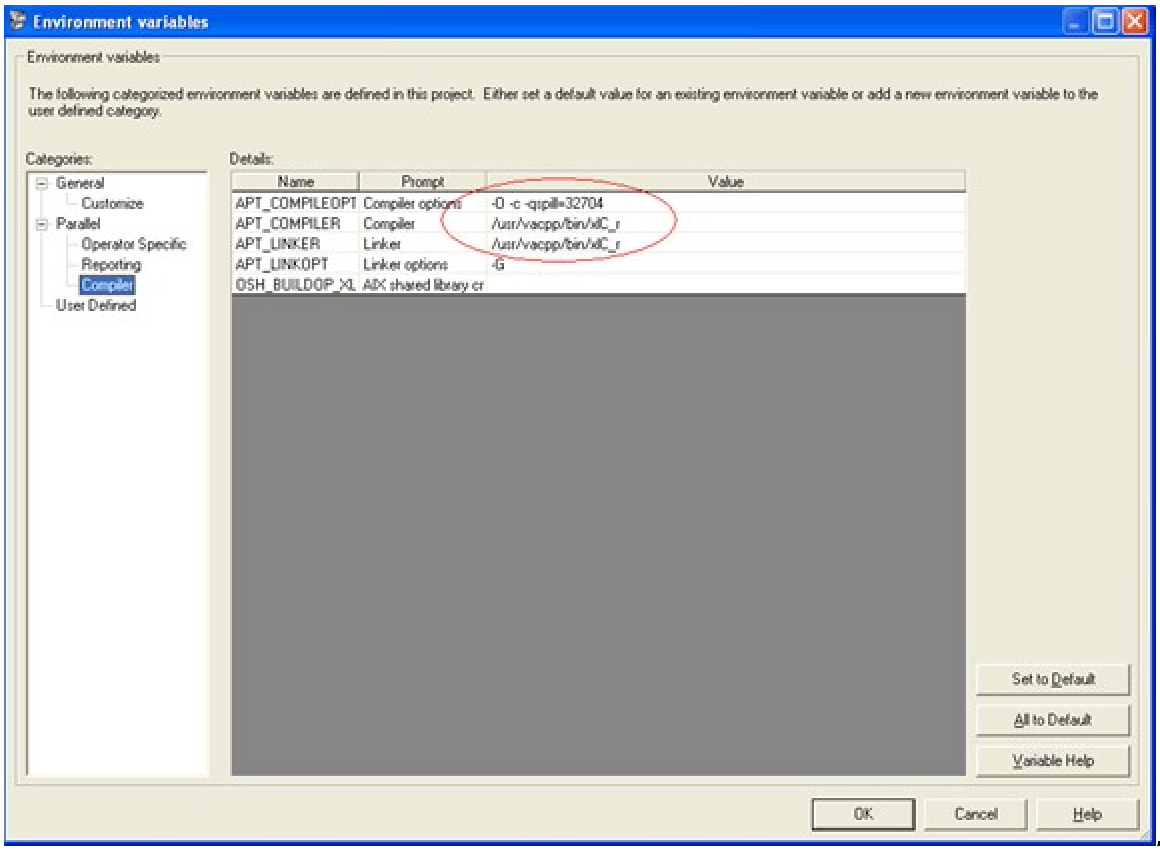
The following categorized environment variables are defined in this project. Either set a default value for an existing environment variable or add a new environment variable to the user defined category.

Categories: General, Customize, Parallel, Operator Specific, Reporting, Compiler, User Defined

Details:

Name	Prompt	Value
LIBPATH	LD library path	/usr/lib
PATH	Shell search path	/usr:/usr/bin:/usr/vacpp:/usr/vacpp/bin
TMPDIR	Temporary directory	/ascl/scratch

Step 6: Compiler specification of datastage is present here.


Environment variables

Environment variables

The following categorized environment variables are defined in this project. Either set a default value for an existing environment variable or add a new environment variable to the user defined category.

Categories: General, Customize, Parallel, Operator Specific, Reporting, Compiler, User Defined

Details:

Name	Prompt	Value
APT_COMPILEOPT	Compiler options	-O -c -qspill=32704
APT_COMPILER	Compiler	/usr/vacpp/bin/xlC_r
APT_LINKER	Linker	/usr/vacpp/bin/xlC_r
APT_LINKOPT	Linker options	-G
OSH_BUILDOP_XL	AIX shared library cr	

.cp
p

In Case Of GNU compiler we compile a cpp code like this

<Compiler Name> < Compile option> < cpp File name>

e.g. `g++ -o test.cpp`

Compiling command for datastage routine will be:-

/usr/vacpp/bin/xlC_r -O -c -qspill=32704 test

This command will create an object file test.o

[Disclaimer: - The Compiler name and options are installation dependent.]

DATASTAGE LEARNINGS-4: REUSABILITY IN DATASTAGE

Below are some of the ways through which reusability can be achieved in DataStage.

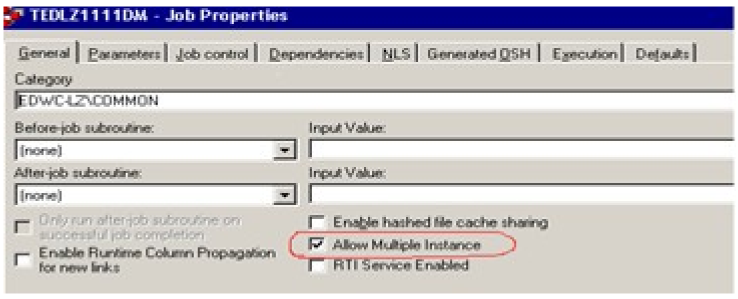
- **Multiple Instance Jobs.**
- **Parallel Shared Container**
- **After-job Routines.**

Multiple-Instance Jobs:
Generally, in data warehousing, there would be scenarios/requirement to load the maintenance tables in addition to the tables in which we load the actual data. These maintenance tables typically have data like

1. **Count of records loaded in the actual target tables.**
2. **Batch Load Number/Day on which the load occurred**
3. **The last processed sequence id for a particular load**

This information can also be used for data count validation, email notification to users. So, for loading these maintenance tables, it is always suggested to design a single multiple-instance job instead of different jobs with the same logic. In the Job Properties tab, check the option "Allow Multiple Instance". The job activity which attach a multiple-instance job shows an additional tab "Invocation Id" .So we can have different job sequences triggering the same job with different invocation ids. Note that a job triggered by different

sequences with same invocation id will abort either one or both of the job sequences.



TEDLZ1111DM - Job Properties

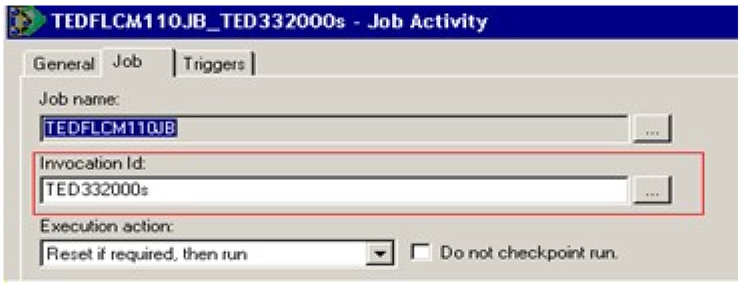
General | Parameters | Job control | Dependencies | NLS | Generated OSH | Execution | Defaults

Category: EDWC-LZ\COMMON

Before-job subroutine: (none) — Input Value:

After-job subroutine: (none) — Input Value:

- [ ] Only run after-job subroutine on successful job completion
- [ ] Enable Runtime Column Propagation for new links
- [ ] Enable hashed file cache sharing
- [x] Allow Multiple Instance
- [ ] RTI Service Enabled



TEDFLCM110JB_TED332000s - Job Activity

General | Job | Triggers

Job name: TEDFLCM110JB

Invocation Id: TED332000s

Execution action: Reset if required, then run

- [ ] Do not checkpoint run.

Parallel Shared Container:

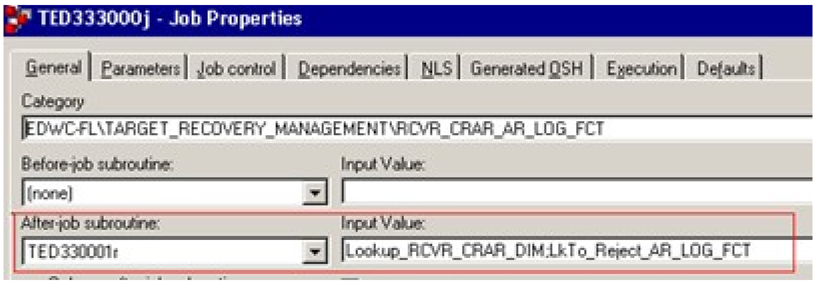
A shared container would allow a common logic to be shared across multiple jobs. Enable RCP at the stage preceding the Shared container stage.For the purpose of sharing across various jobs, we will not propagate all the column metadata from the job into the stages present in shared container. Also ,ensure that jobs are re-compiled when the shared container is changed.

In our project, we have used shared container for implementing the logic to generate unique sequence id for each record that is loaded into the target table. [We will discuss different ways to generate sequence id (one-up number) in later section.]

After-job Routines:

After/Before job subroutines are types of routines which run after/before the job to which the routine is attached. We might have a scenario where in we shouldn’t have any of the input records to be rejected by any of the stages in the job. So we design a job which have reject links for different stages in the job and then code a common after-job routine which counts the number of records in the reject links of the job and aborts the job when the count exceeds a pre-defined limit.

This routine can be parameterised for stage and link names and can then be re-used for different jobs.



TED333000j - Job Properties

General | Parameters | Job control | Dependencies | NLS | Generated OSH | Execution | Defaults

Category: EDWC-FL\TARGET_RECOVERY_MANAGEMENT\RCVR_CRAR_AR_LOG_FCT

Before-job subroutine: (none) Input Value:

After-job subroutine: TED330001r Input Value: Lookup_RCVR_CRAR_DIM:LkTo_Reject_AR_LOG_FCT

DATASTAGE LEARNINGS-3: USAGE OF APT_OLD_BOUNDED_LENGTH

For variable-length fields, the parallel engine always allocates the space equivalent to their maximum specified lengths. This is because the engine can easily determine the field boundaries without having to look for delimiters, which will ease the CPU processing. However, the I/O operations will be costlier. Therefore, if there are many values, which are much shorter than their maximum length, then there is lot of unused space, which moves around between the operators, datasets and fixed-width files. This is a huge performance hit for all stages especially sort stage, which consumes a lot of scratch space, buffer memory, CPU resources. This is even more dangerous if we have many of such columns.

This is common for fields like address, where lengths allocated are much higher than the average length of an address. One way to prevent this scenario is by setting the environmental variable**APT_OLD_BOUNDED_LENGTH.**

Disadvantages: Some parallel datasets generated with InfoSphere® DataStage® 7.0.1 and later releases require more disk space when the columns are of type VarChar when compared to 7.0. This is due to changes added for performance improvements for bounded length VarChars in 7.0.1.

Set APT_OLD_BOUNDED_LENGTH to any value to revert to pre-7.0.1 storage behavior when using bounded length varchars. Setting this variable can have adverse performance effects. The preferred and more performant solution is to use unbounded length VarChars (don’t set any length) for columns where the maximum length is rarely used, rather than set this environment variable

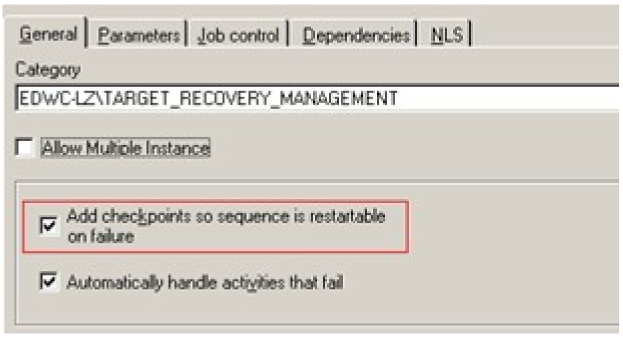
DATASTAGE LEARNINGS-1: RESTARTABILITY IN JOB SEQUENCE

MAR

Below are the important points to be taken care of, for proper restartability to be implemented in Job Sequence

- **Checkpoint information has to be maintained and sequence has to be made restartable. So, the option "Add checkpoints so sequence is restartable" in the Job Sequence Properties, should be checked. Also only then we would find the option "Do not checkpoint run" under the "Job" tab in each of the job activities present in the job sequence.**
- **The option "Do not checkpoint run" in job activity should not be checked because if another job later in the sequencer fails, and the sequencer is restarted, this job will not rerun. However if the logic demands, then it has to be checked.**
- **The option "Reset if required, then run" has to be chosen in the job activity so that when an aborted sequence is re-run then, the aborted job(due to which sequence is aborted) will re-run successfully.**

Below are the screenshots :

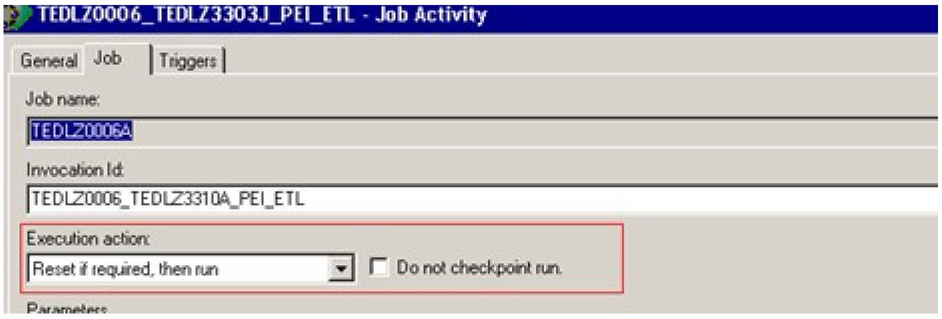


Category
EDWC-LZ\TARGET_RECOVERY_MANAGEMENT

☐ Allow Multiple Instance

☑ Add checkpoints so sequence is restartable on failure

☑ Automatically handle activities that fail



TEDLZ0006_TEDLZ3303J_PEI_ETL - Job Activity

Job name:
TEDLZ0006A

Invocation Id:
TEDLZ0006_TEDLZ3310A_PEI_ETL

Execution action:
Reset if required, then run

☐ Do not checkpoint run.

Parameters

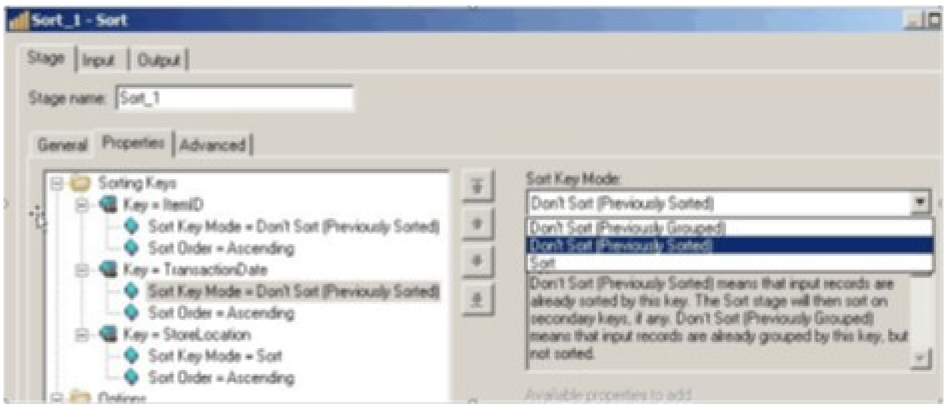
DATASTAGE LEARNINGS-2: SORT OPERATION IN DATASTAGE

DataStage provides two methods for parallel sorts:

- **Link Sort :** This is used when we use a keyed partitioning method when using any other stage (i.e lookup, join, remove duplicate etc)
- **Sort Stage :** This is used when we intend to implement a sort operation using a separate sort stage

By default, both methods use the same internal sort package (Tsort operator)
When compared to Sort Stage1, Link sort provides fewer options but makes maintenance of job easier. Therefore, it is better to prefer Link Sort unless we need to use options present only in Sort Stage. Two of the commonly used options are Sort Key mode, Restrict Memory Usage.

Sort Key mode: This is used in a scenario wherein we need to subgroup/subsort an already sorted/grouped data on a different key/s(secondary keys). Note that all key columns must be defined in sort stage. Reused sort keys must be specified with “Don’t sort (Previously sorted)” or “Don’t sort (Previously Grouped)” mode accordingly. New sort keys are specified with Sort Mode.



Sort_1 - Sort

Stage | Input | Output

Stage name: Sort_1

General | Properties | Advanced

```
Sorting Keys
  Key = ItemID
    Sort Key Mode = Don't Sort (Previously Sorted)
    Sort Order = Ascending
  Key = TransactionDate
    Sort Key Mode = Don't Sort (Previously Sorted)
    Sort Order = Ascending
  Key = StoreLocation
    Sort Key Mode = Sort
    Sort Order = Ascending
Options
```

Sort Key Mode:
Don't Sort (Previously Sorted)
- Don't Sort (Previously Grouped)
- Don't Sort (Previously Sorted)
- Sort

Don't Sort (Previously Sorted) means that input records are already sorted by this key. The Sort stage will then sort on secondary keys, if any. Don't Sort (Previously Grouped) means that input records are already grouped by this key, but not sorted.

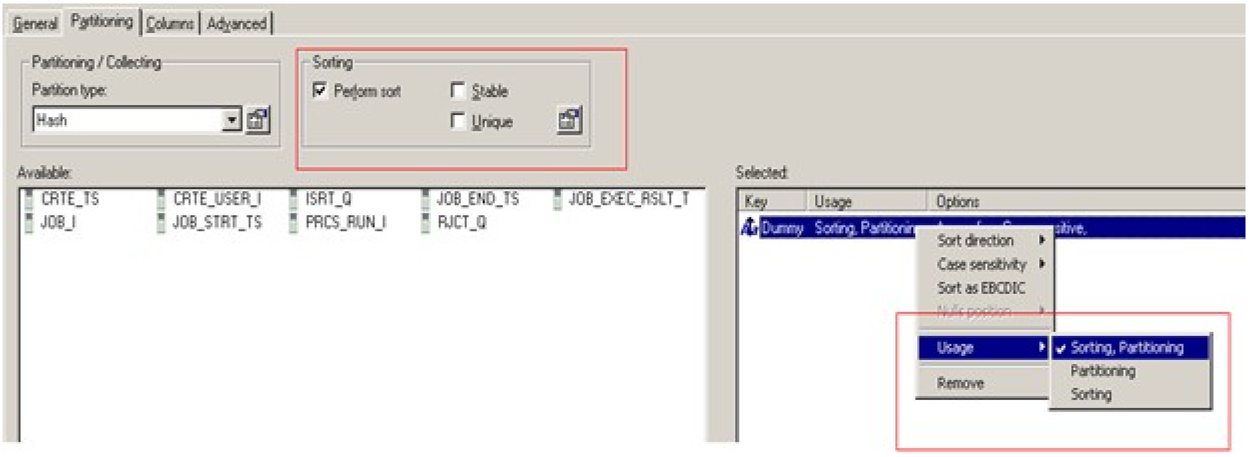
Available properties to add

Key points to remember while using sort key mode:

- All the Keys with the “Don’t Sort(Previously Sorted)/ (Previously Grouped)” property must be specified before the new key sortkey(Sort Mode).
- Key column order for the above keys must match the key columns and order defined in the previously-sorted input dataset.

Link Sort : This is used when we use a keyed partitioning method in any other stage. Here the various options available are as follows :

- Stable Sort : This preserves the order of non-key columns in each sort group.However ,unless needed this option has to be disabled since it incurs additional overhead in the sort algorithm.
- Unique Sort : Enable this option to ensure that , if multiple records have identical sorting key values ,only one of them is retained.If stable sort is set , then the first record is retained.
- Other options include sort direction(asc/desc) , case sensitivity and position of nulls (before/after) if the column is nullable.



Partition type: Hash

[✓ Perform sort] Stable Unique

Dummy Sorting, Partitioning

Usage: [✓ Sorting, Partitioning]

Restrict Memory Usage: Sort is a particularly expensive task which requires CPU, memory, and disk resources. The Sort operator implements a segmented merge sort which invariably uses the scratch disk space. So if input data is huge, then there is a chance that the scratch space is not enough for sorting. In that case, we have the option to increase the buffer used for sort. The default buffer is 20MB /node.

FIND DATASTAGE JOBS FROM COMMAND LINE

USING DSJOB COMMAND IN DATASTAGE, WE CAN FIND JOB AND ITS LOCATION.

This script uses two parameters: 1.ProjectName 2.JobName

```
#!/bin/ksh
```

```
. $DSHOME/dsenv

proj=$1

jobname=$2

# You have to search where exactly your project is placed.

#In our case it is in below mentioned path

cd /data/projectlib/$proj

echo Location for the job $jobname is

$DSHOME/bin/uvsh “SELECT CATEGORY FROM DS_JOBS WHERE NAME = ‘$jobname’;”
```

ACCESSING DATASTAGE LATEST LOGS IN UNIX

FEBRUARY 28, 2013

To get datastage job log from UNIX command line using dsjob command

```
#!/bin/ksh

PARMDIR=/home/dsadm/testdir

. $DSHOME/dsenv

cd $DSHOME/bin

PROJECT_NAME=datastage_test

JOB_NAME=DS_Tesing_of_logs

##Getting the first and last event id

start=`dsjob -logsum -type STARTED -max 2 $PROJECT_NAME $JOB_NAME|cut -f1`

echo $start

dsjob -logdetail $PROJECT_NAME $JOB_NAME $start >
$PARMDIR/output_log_new.txt
```

Why Entire partition is used in LOOKUP stage ?

Entire partition has all data across the nodes So while matching(in lookup) the records all data should be present across all nodes.

For lookup sorting is not required.so when we are not using entire partition then reference data splits into all nodes. Then each primary record need check with all nodes for matched reference record.Then we face performance issue.If we use entire in lookup then one primary record needs to look into 1 node is enough.if match found then that record goes to target otherwise it move to reject,drop etc(based on requirement)no need check in another node.In this case if we are running job in 4 nodes then at a time 4 records should process.
Note:Please remember we go for lookup only we have small reference data.If we go for big data it is performance issue(I/O work will increase here) and also some times job will abort.

What is stable sort?
Since keeping track of relative record location means more work, setting Stable to "False" will speed up performance.

Stable sort means "if you have two or more records that have the same exact keys, keep them in the same order on output that they were on input".

What are HLD and LLD?
HLD: It refers to the functionlity to be achieved to meet the client requirement. Precisely speaking it is a
diagramatic representation of clients operational systems,staging areas, dwh n datamarts. also how n what frequency
the data is extracted n loaded into the target database.

LLD: It is prepared for every mapping along with unit test plan. It contains the names of source definitions, target
definitions, transformatoins used, column names, data types, business logic written n source to target field
matrix, session name, mapping name.

How to handle values in Sequential file stage?
Open Sequential File---> Go to Format--->click on Field Defaults--->bottom right side you will find AVAILABLE Properties to ADD, Under that select, "NULL FIELD VALUE" and give the value as " 0" [zero]. You'll get the Null records in your output sequential file.

Have you ever used or use RCP option in your project?
Yes, to Load Data from Multiple Sources to Target Tables/Files with out Much Transformation i.e Straight Load.

How do you protect your project?
In Administrator

Can aggregator and transformer stage used for sorting data? How

Yes , in Transformer , I want to take count in stage Variable for rec the records should be sorted.

What are XML files? How do you read data from XML files and which stage to be used?

Using XML input file Stage.

How do you pass filename as the parameter for a job?
Using Job Properties

What are stages in sequences?

Job Activity , Expection Handler , Terminator, Exec Command, Wait for a File, Mail Notification

what is the main diffrence between keychange column and cluster key change column in sort stage
create key change column generates while sorting the data..It generate one for first record and zero for rest of the records by group wise..

cluster key column generates on sorted data when sort mode is donot sort

FEATURES OF DATASTAGE | DATASTAGE FEAUTURES

Datastage Features are

1) Any to Any (Any Source to Any Target)
2) Platform Independent.
3) Node Configuration.
4) Partition Parallelism.
5) Pipeline Parallelism.

1)Any to Any
That means Datastage can Extrace the data from any source and can loads the data into the any target.

2) Platform Independent
The Job developed in the one platform can run on the any other platform. That means if we designed a job in the Uni level processing, it can be run in the SMP machine.

3)Node Configuration
Node Configuration is a technique to create logical C.P.U
Node is a Logical C.P.U

4)Partition Parallelism
Partition parallelim is a technique distributing the data across the nodes based on the partition techniques. Partition Techniques are
a) Key Based
b) Key Less

a) Key based Techniques are
1) Hash 2)Modulus 3) Range 4) DB2

b) Key less Techniques are
1) Same 2) Entire 3) Round Robin 4) Random

5) Pipeline Parallelism
Pipeline Parallelism is the process, the extraction, transformation and loading will be occurred simultaneously.

Re- Partitioning: The distribution of distributed data is Re-Partitioning.

Reverse Partitioning: Reverse Partitioning is called as Collecting.

Collecting methods are
Ordered
Round Robin
Sort Merge
Auto

WHAT IS COMPILER IN DATASTAGE | COMPILATION PROCESS IN DATASTAGE

Compilation is the process of converting the GUI into its machine code .That is nothing but machine understandable language.

In this process it will checks all the link requirements, stage mandatory property values, and if there any logical errors.

And Compiler produces OSH Code.

WHAT IS MODELING OF DATASTAGE | MODELING OF DATASTAGE

Modeling is a Logical and physical representation of Source system.
Modeling have two types of Modeling Tools
They are
ERWIN AND ER-STUDIO

In Source System there will be a ER-Model and
in the Target system there will be a ER-Model and Dimensional Model
Dimension:- The table which was designed for the client perspective. We can see in many ways in the Dimension tables.

And there are two types of Models.
They are
Forward Engineering (F.E)
Reverse Engineering (R.E)

F.E:- F.E is the process starting the process from the scratch for banking sector.
Ex: Any Bank which was required Datawarehouse.
R.E:- R.E is the process altering existing model for another bank.

HOW DOES HASH PARTITION WORKS IN DATASTAGE

Hash Partition technique is used to send the rows with same key column values to the same partition.
In Datastage, partition techniques are usually distributed into two types. They are
a) Key based partition Technique

b) Key Less Partition Technique

In Key based partition technique
a)Hash Partition Technique
b)Modulus
c)Range
d)DB2

In Key less partition technique, they are
a)Same
b)Entire
c)Round Robin
d)Random

Example for Hash partition technique:

If we use a hash partition technique and if we have a sample data as below

e_id,dept_no
1,10
2,10
3,20
4,20
5,30
6,40

Data will partitioned as below
In 1st partition

10,10

In 2nd Partition

20,20

In 3rd Partition

30
In 4th Partition

40

DIFFERENT BETWEEN HASH AND MODULUS TECHNIQUE

Hash and Modulus techniques are Key based partition techniques.

Hash and Modulus techniques are used for different purpose.

If Key column data type is textual then we use has partition technique for the job.

If Key column data type is numeric, we use modulus partition technique.

If one key column numeric and another text then also we use has partition technique.

if both the key columns are numeric data type then we use modulus partition technique.

WHAT IS PARTITION PARALLELISM

Partition Parallelism is a technique of distributing the records across the nodes

based on different partition techniques.

Partition techniques are very important to get the good performance of the job.

We need to select right partition technique for the right stage.

Partition techniques are

Key based Techniques And

Key less Techniques

Key based Techniques are

a) Hash

b) Modulus

c) Range

d) DB2

Key Less Techniques are

a) Same

b) Entire

c) Round Robin
d) Random

PARTITIONS IN DATASTAGE

Partition Techniques are used in datastage to get good performance.

They are different types of Partition Techniques in datastag. They are

a) Key Based Partition Techniques

b) Key Less Partition Techniques

Key Less Techniques are

1) Same

2) Entire

3) Round Robin

4) Random

a) Same: This technique is used in oder to do not alter the existing partition technique in the previous stage.

b) Entire: Each Partition gets the entire dataset. That is rows are duplicated.

c) Round Robin :In Round Robin Technique rows are evenly distributed among the Partition.

d) Random: Partition a row is assigned to is Random.

KEY BASED PARTITIONED TECHNIQUES | PARTITIONING TECHNIQUES

Key Partitioned Techniques are

1) Hash

2) Modulus

3) Range

4) DB2

Hash:-- In Hash Partitioning Technique, Rows with same Key Column values go to the same partition. Hash is the technique often used in the datastage.

We Hash Partitioning technique when the key column data type is text.

Modulus: --- It Assigns each rown of an input dataset to partition , as determined by a specified numeric Key column.
And we use Modulus Partition technique, when the key column data type is numeric.

If one key column is numeric and another is text , the also we will go with the Hash Partitioning technique.

Modulus Partition technique can be performed only on the Numbers.

Range: Range Partition technique is similar to the Hash Partition technique.

But the partition mapping is user determined and partitions are ordered.

Rows are distributed according to the values in one or more key fields, using a range map.

Db2: Db2 Partitioning technique matches db2 EEE Partitioning.

PERFORMANCE TUNING IN DATASTAGE

It is more important to do the performance tuning in any job of datastage.
If Performance of the Job taking too much time to compile, we need to modify the job design. So that we can good performance to the Job.

For that

a) Avoid using Transformer stage where ever necessary. For example if you are using Transformer stage to change the column names or to drop the column names. Use Copy stage, rather than using Transformer stage. It will give good performance to the Job.

b)Take care to take correct partitioning technique, according to the Job and requirement.

c) Use User defined queries for extracting the data from databases .

d) If the data is less , use Sql Join statements rather then using a Lookup stage.

e) If you have more number of stages in the Job, divide the job into multiple jobs.

WHAT IS ETL PROJECT PHASE | PROJECT PHASE WITH ETL TOOL(DATASTAGE)

ETL Project contains with four phases to implement the project.

ETL Means Extraction Transformation Loading

ETL is the tool used to extract the data
Transformation the Job
And to Load the Data

It is used for Business Developments

And four phases are

1) Data Profiling

2) Data Quality

3) Data Transformation

4) Meta data management

Data Profiling:-

Data Profiling performs in 5 steps. Data Profiling will analysis weather the source data is good or dirty or not.
And these 5 steps are

a) Column Analysis
b) Primary Key Analysis
c) Foreign Key Analysis
d) Cross domain Analysis
e) Base Line analysis

After completing the Analysis, if the data is good not a problem. If your data is dirty, it will be sent for cleansing. This will be done in the second phase.

Data Quality:-

Data Quality, after getting the dirty data it will clean the data by using 5 different ways.

They are

a) Parsing
b) Correcting
c) Standardize
d) Matching
e) Consolidate

Data Transformation:-

After competing the second phase, it will gives the Golden Copy.
Golden copy is nothing but single version of truth.
That means , the data is good one now.

HOW DOES HASH PARTITION WORKS IN DATASTAGE

Hash Partition technique is used to send the rows with same key column values to the same partition.
In Datastage, partition techniques are usually distributed into two types. They are
a) Key based partition Technique
b) Key Less Partition Technique

In Key based partition technique
a)Hash Partition Technique
b)Modulus
c)Range
d)DB2

In Key less partition technique, they are
a)Same
b)Entire
c)Round Robin
d)Random

Example for Hash partition technique:

If we use a hash partition technique and if we have a sample data as below

e_id,dept_no
1,10
2,10
3,20
4,20
5,30
6,40

Data will partitioned as below
In 1st partition

10,10

In 2nd Partition

20,20

In 3rd Partition

30
In 4th Partition

40

job.   They are 1. APT_CONFIG_FILE   2. APT_RECORD_COUNT   3. APT_DUMP_SCORE
How to create user defined environment variable (parameter)?u can create parameter in 2 ways

1.job level params

 2.project level params.Job-Level: when u want to create job level params ,go to job properties and use it.

 Project -Level :when u want to create project level, go to DS-Admin and click on environment variable and define which parameter u need and open design go to job properties and call those params (which u defined in DS admin) in job properties and use it.

30 jobs are running in unix.i want to find out my job.how to do this? Give me command?
ps -ef|grep PID

 how to move project from developement to uat?
By using the Datastage Manager we can move the project from Dev to Uat. Through datastage manager Export the project into your local machine as .dsx format (project.dsx) from DEV server. The same .dsx (project.dsx) import into UAT server by using the datastage manager.

How to do error handling in datastage?
 Error handling can be done by using the reject file link.what are the errors coming through job needs to be capture in sequential file and that file needs to be fetch in job which will load this exceptions or errors in database.
 How can u Call the Shell Scripting/Unix Commands in Job Sequence?
There are two scenarios where u myt want to call a script
Scenario 1(Dependency exists between script and a job): Where a job has to be executed first then the script has to run, upon completion of script execution only the sec job has to be invoked. In this case develop a sequencer job where first job activity will invoke the first job then using Execute command activity call the script u would desire to invoke by typing "sh <script name>" in the command property of the activity, then with the other job activity call the second job.
Scenario 2: (Script and job are independent) : In this case right in your parallel job say job1, under job properties u can find "After-job subroutine" where u need to select "ExecSH" and pass the script name which you would like to execute. By doing this once the job1 execution completes the script gets invoked. The job succeeding the job1 say job2 doesn’t wait for the execution of the script.

WHAT IS OLAP (ONLINE ANALYTICAL PROCESS)

Online Analytical Process (OLTP) is a characterized by relatively low volume of transactions. Actually the queries are often very complex. In the OLAP System response time more. In OLAP Database there is Aggregated, historical Inf. Data , stored in multi-dimensional schemas.

WHAT IS OLTP ? USES OF OLTP?

OLTP is nothing but Online Transaction Processing.
It will be characterized by a large number of short online transactions. The main emphasis for OLTP system is to put on very fast query processing.In order to get the data faster to the end-users. And we use the Online transaction process for the fast process. And Oltp system is used for data integrity in multi access environments, and effectiveness measured by number of transactions per second.

WHAT IS RCP?

RCP is nothing but runtime column propagation. When we send the data from source to the target, some times we need to send only required columns . In this time, if we like to send only required columns to the target we can do it by enabling the RCP Option.

WHAT IS RCP | WHAT IS RUNTIME COLUMN PROPOGATION

RCP is nothing but Runtime Column Propagation.

When we run the Datastage Jobs, the columns may change from one stage to another

stage. At that point of time we will be loading the unnecessary columns in to the

stage, which is not required. If we want to load the required columns to load into

the target, we can do this by enabling a RCP.

If we enable RCP, we can sent the required columns into the target.

HOW TO REMOVE SPECIAL CHARACTERS DATA AND LOAD REST OF THE DATA

Here we are going to know how to remove Special characters data rows and load rest of the rows into the target.

Some times we get the data with special characters added for some of the rows.
If we like to remove those special characters mixed rows in the column. We can use **Alpha**function.

Alpha Function is used for this Job.

If we use "Alpha" function. It will drop the special characters mixed rows and loads the rest of the rows into the target.

So you can take sequential file to read the data and you can take Transformer stage to apply business logic.
In Transformer stage in Constrain you can write the Alpha function.
And Drag and Drop into the Target. .
Then Compile and Run.

Datastage Common Errors-Warnings and resolution

1)When we use same partitioning in datastage transformer stage we get the following warning in 7.5.2 version.

TFCP000043 2 3 input_tfm: Input dataset 0 has a partitioning method other than entire specified; disabling memory sharing.

This is known issue and you can safely demote that warning into informational by adding this warning to Project specific message handler.

2) **Warning: A sequential operator cannot preserve the partitioning of input data set on input port 0**

Resolution: Clear the preserve partition flag before Sequential file stages.

3)**DataStage parallel job fails with fork() failed, Resource temporarily unavailable**

On aix execute following command to check maxuproc setting and increase it if you plan to run multiple jobs at the same time.

```
lsattr -E -l sys0 | grep maxuproc
maxuproc        1024            Maximum number of PROCESSES allowed per user      True
```

4)**TFIP000000 3 Agg_stg: When checking operator: When binding input interface field "CUST_ACT_NBR" to field "CUST_ACT_NBR": Implicit conversion from source type "string[5]" to result type "dfloat": Converting string to number.**

Resolution: use the Modify stage explicitly convert the data type before sending to aggregator stage.

5)**Warning: A user defined sort operator does not satisfy the requirements.**

Resolution:check the order of sorting columns and make sure use the same order when use join stage after sort to joing two inputs.

6)**TFTM000000 2 3 Stg_tfm_header,1: Conversion error calling conversion routine timestamp_from_string data may have been lost**

TFTM000000 1 xfmJournals,1: Conversion error calling conversion routine decimal_from_string data may have been lost

Resolution:check for the correct date format or decimal format and also null values in the date or decimal fields before passing to datastage StringToDate, DateToString,DecimalToString or StringToDecimal functions.

7)**TOSO000119 2 3 Join_sort: When checking operator: Data claims to already be sorted on the specified keys the 'sorted' option can be used to confirm this. Data will**

be resorted as necessary. Performance may improve if this sort is removed from the flow

Resolution: Sort the data before sending to join stage and check for the order of sorting keys and join keys and make sure both are in the same order.

8)**TFOR000000 2 1 Join_Outer: When checking operator: Dropping component "CUST_NBR" because of a prior component with the same name.**

Resolution:If you are using join,diff,merge or comp stages make sure both links have the differnt column names other than key columns

9)**TFIP000022 1 oci_oracle_source: When checking operator: When binding output interface field "MEMBER_NAME" to field "MEMBER_NAME": Converting a nullable source to a non-nullable result;**

Resolution:If you are reading from oracle database or in any processing stage where incoming column is defined as nullable and if you define metadata in datastage as non-nullable then you will get above issue.if you want to convert a nullable field to non nullable make sure you apply available null functions in datastage or in the extract query.

Null function in oracle:NVL,NVL2

Datastage:IsNull,NullToEmpty,NullToZero