IBM **InfoSphere** DataStage

Lab Book

Table of Contents

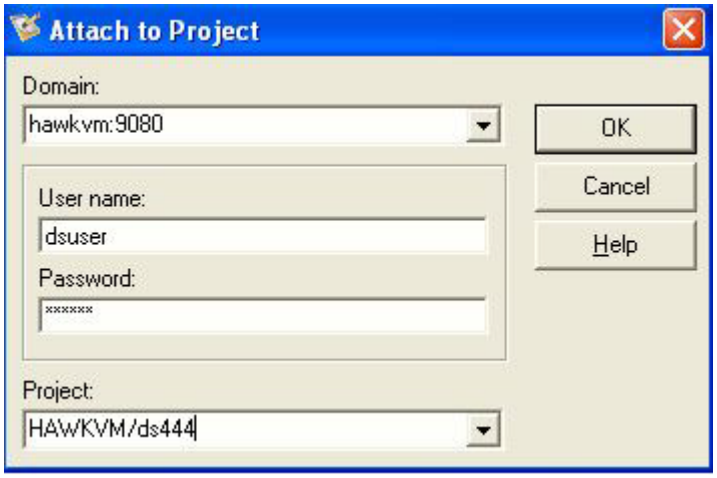
Lab 01: DataStage Designer

Assumptions

- You have created a user ID named dsuser for logging onto DataStage.
- Completed PPT training session for Lession 1 and 2

Task: Log onto DataStage Designer

1. Double-click on the DataStage Designer Client icon on the Windows desktop.
2. Type information to log into your DataStage project using the dsuser ID.

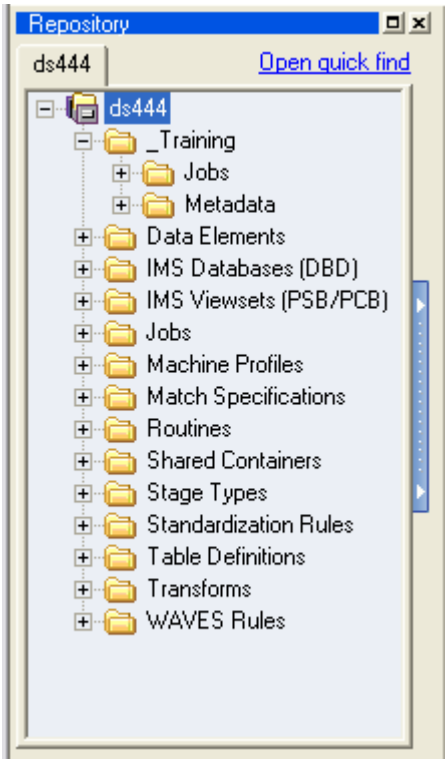


3. Click OK.

Create a Repository folder

1. Select your project folder in the Repository, click your right mouse button, and then click New>Folder. Create a folder named "_Training". Under it, create two folders: Jobs and

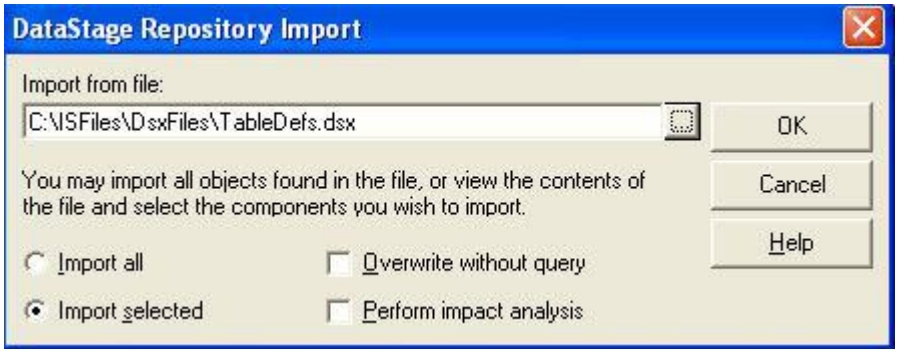
Metadata.



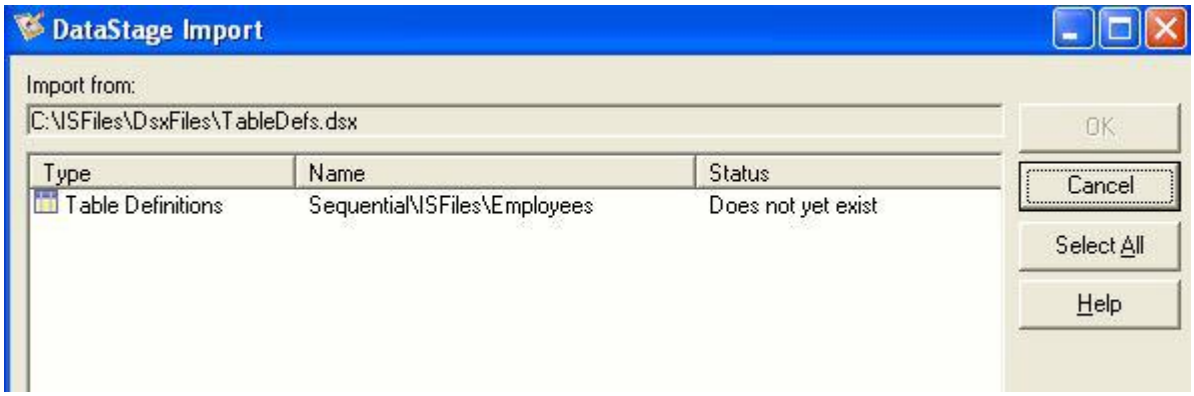
2. Click Repository>Refresh to refresh the Repository (which moves the folder you created to the top).

Task: Import DataStage component files

1. Click Import>DataStage Components.
2. In the Import from file box, select the TableDefs.dsx file in your ISFiles>DsxFiles directory on your client machine.
3. Select the Import selected button.



4. Click OK.

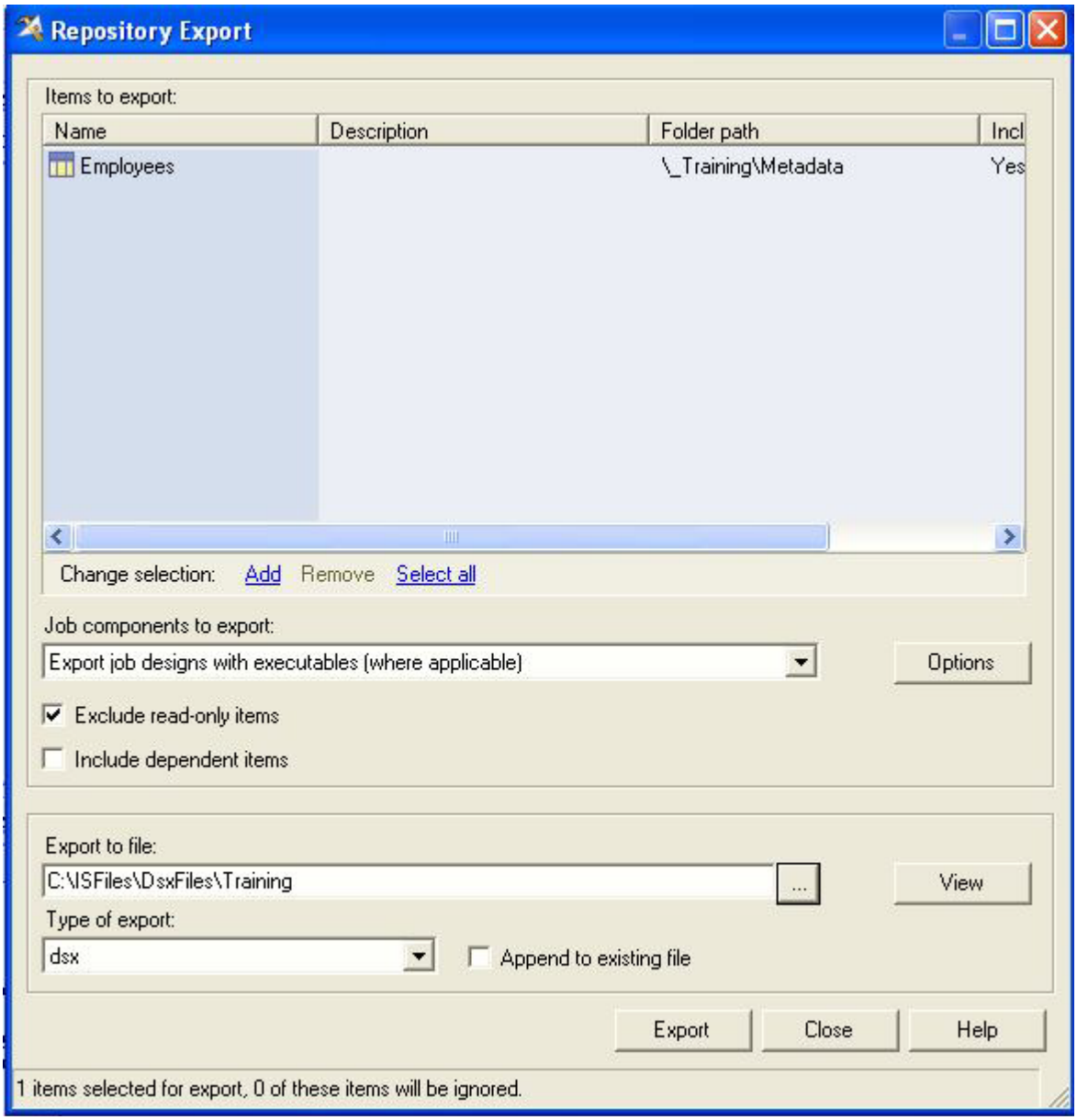


5. Select the table definition and then click OK.
6. Open up the table definition you've imported and examine it. You will find it in the _Training>Metadata folder.
    - On the General tab, note the Data source type, Data source name, and Table/file name.
    - Note the column definitions and their types on the Columns tab.

Task: Backup your project

In this task, you backup (export) your _Training folder into a file named Training.dsx.

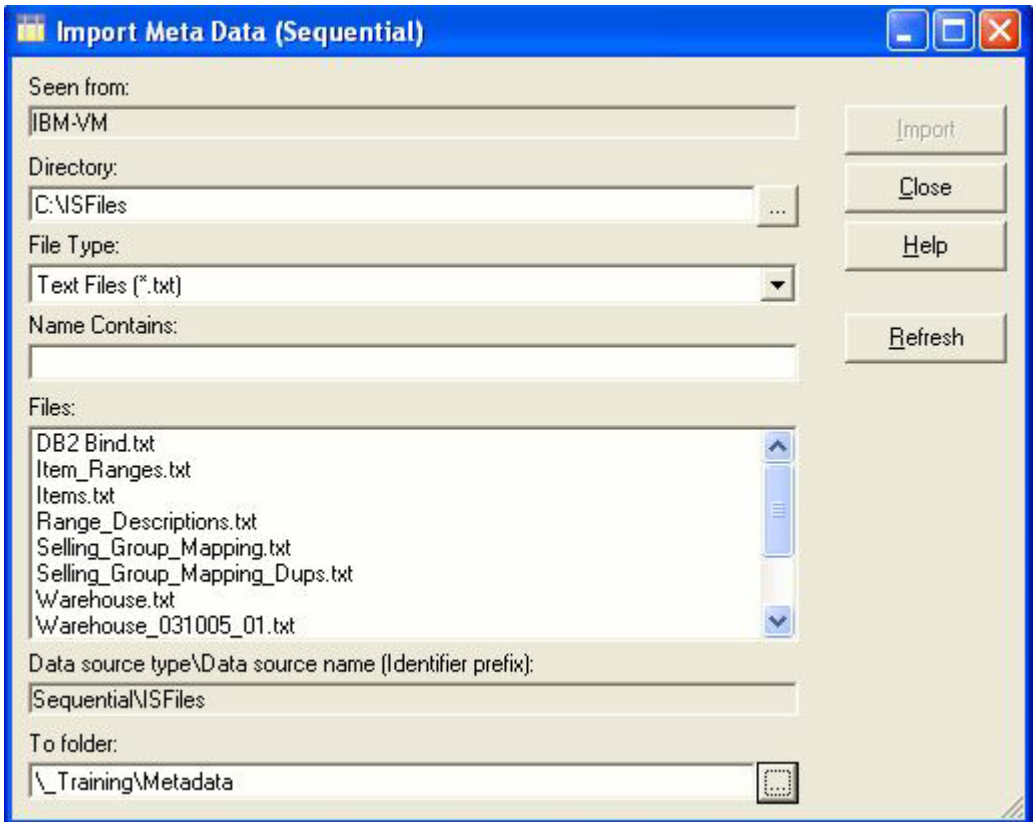
1. Select your _Training folder, click your right mouse button, and then click Export.
2. In the Export to file box, select your ISFiles>DsxFiles folder and specify a file named Training.dsx.



3. Click Export.

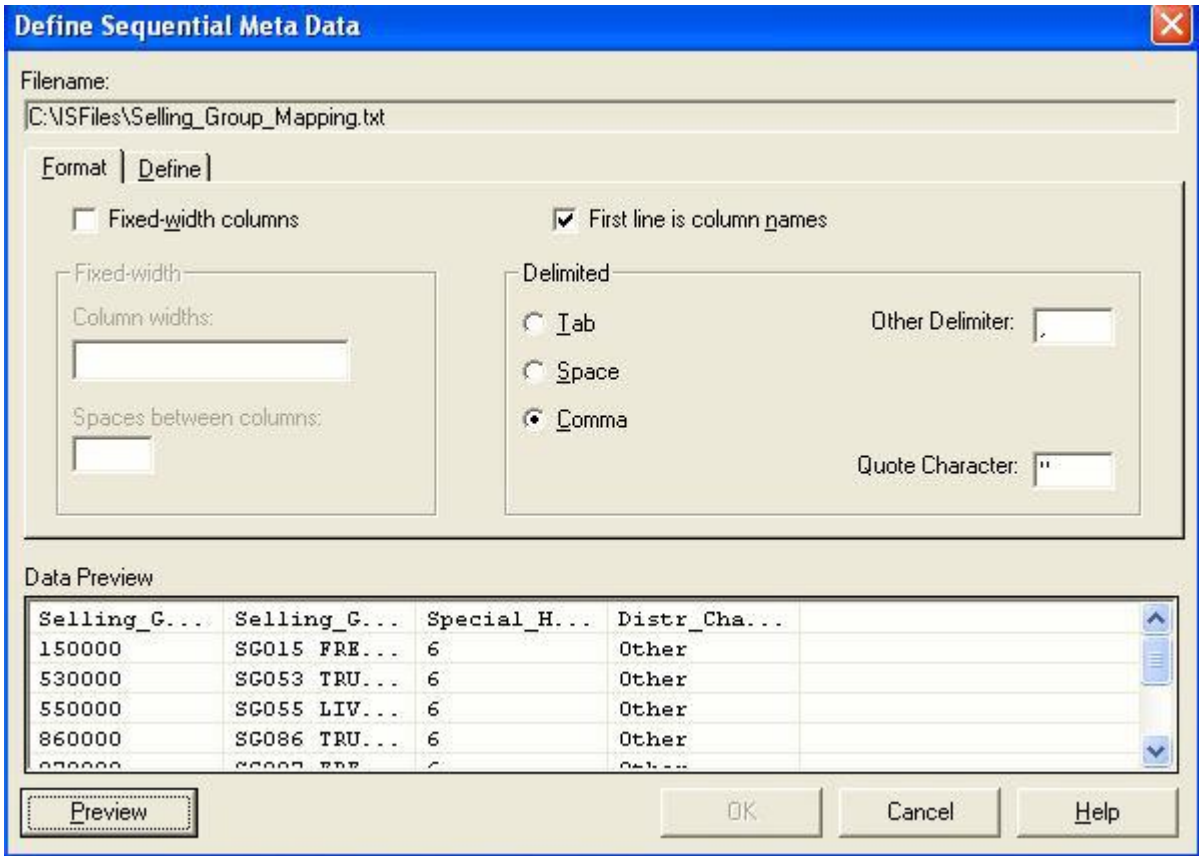
Task: Import a table definition from a sequential file

1. In a text editor, open up the Selling_Group_Mapping.txt file in your ISFiles directory and examine its format and contents. Some questions to consider:
    - Is the first row column names?
    - Are the columns delimited or fixed-width?
    - How many columns? What types are they?
2. In Designer, click Import>Table Definitions>Sequential File Definitions.
3. Select your ISFiles directory.
4. Specify _Training\Metadata as the “To category”.
5. Select your Selling_Group_Mapping.txt file.



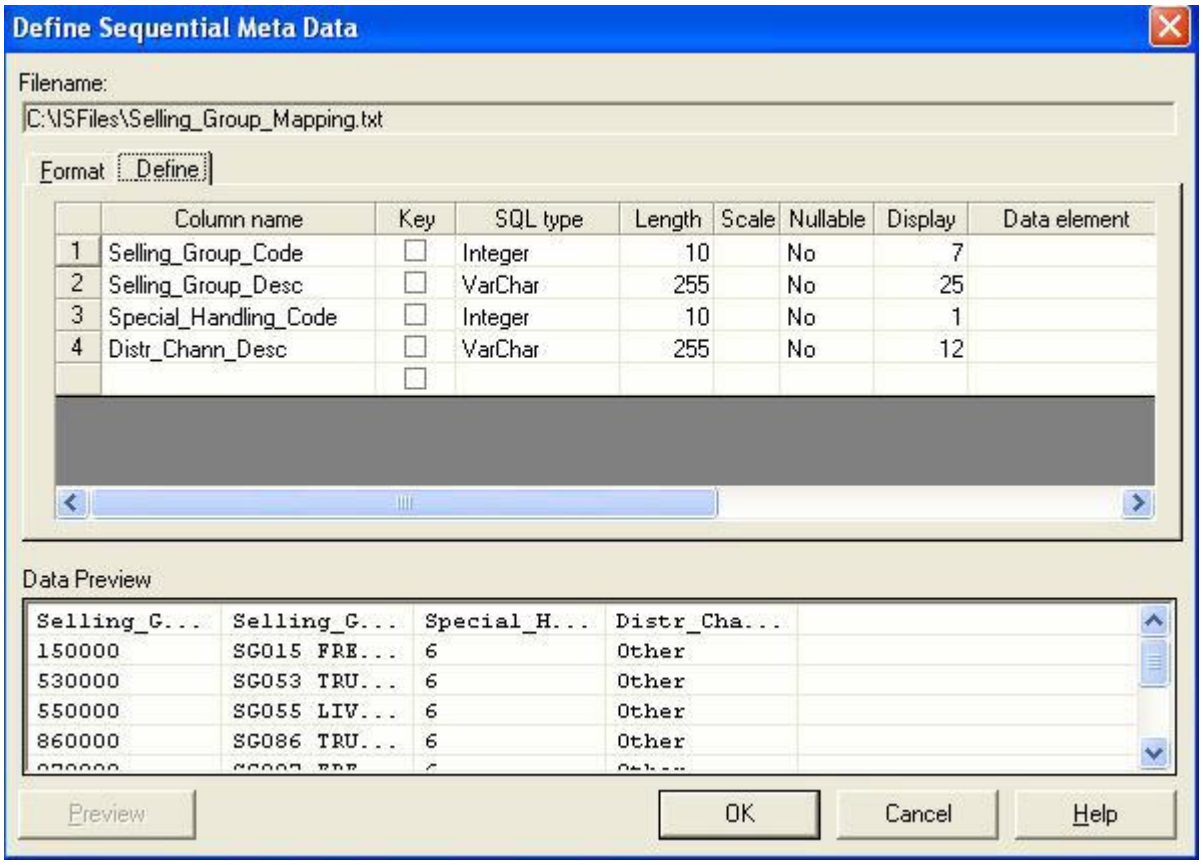
6. Click Import.

7. Specify the general format on the Format tab. Be sure to specify that the first line is column names, if this is the case. Then Data Stage can use these names in the column definitions.



8. Click Preview to view the data in your file in the specified format. This is a check whether you have defined the format correctly. If it looks like a mess, you haven't correctly specified the format.

9. Click the Define tab to examine the column definitions.



10. Click OK, to import your table definition.
11. After closing the import window, locate and examine your new table definition in the Repository window.

Lab 02: Creating Parallel Jobs

Assumptions

- All activities described in Chapter 01 of this lab book are completed.
- Completed PPT training session for Lession 1, 2 and 3

Task: Create a two-node configuration file

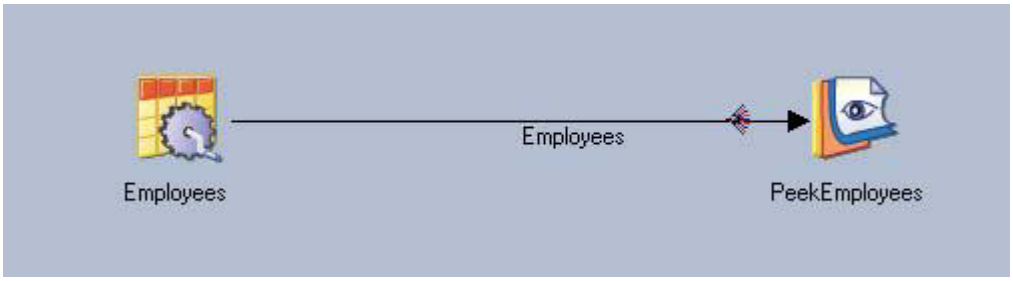
The lab exercises that follow in this and later modules are more instructive when the jobs are run
using a two-node (or more) configuration file. Configuration files are discussed in more detail in
a later module.

1. Click Tools>Configurations.
2. In the Configurations box, select the default configuration.
3. If only one node is listed, make a copy of node through the curly braces and change the name of the node to “node2”. When done, your file should now look like this:

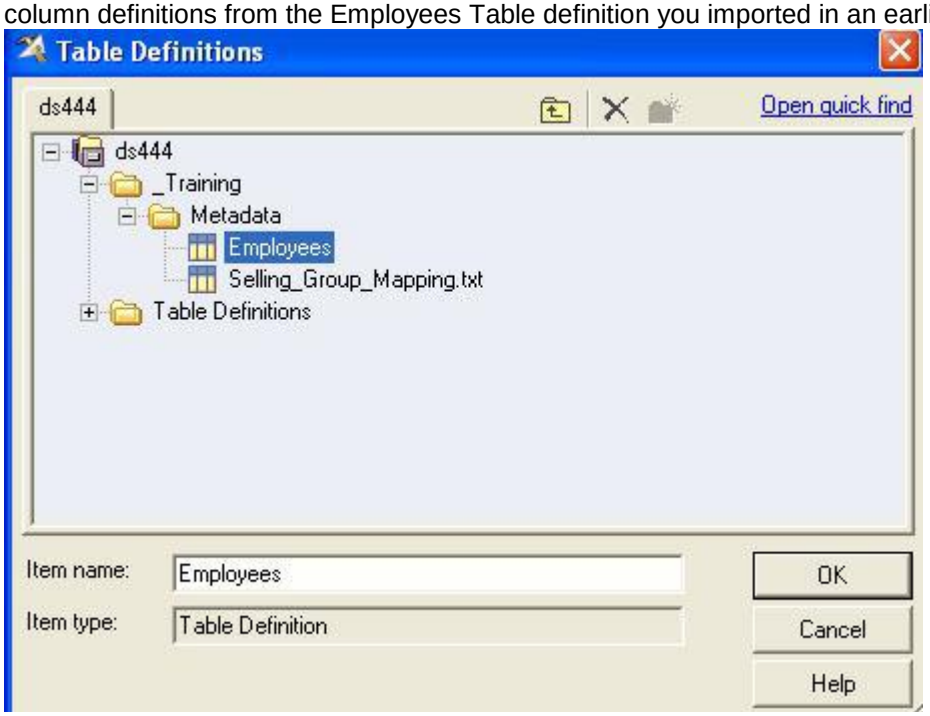
4. Save and close.

Task: Create a parallel job

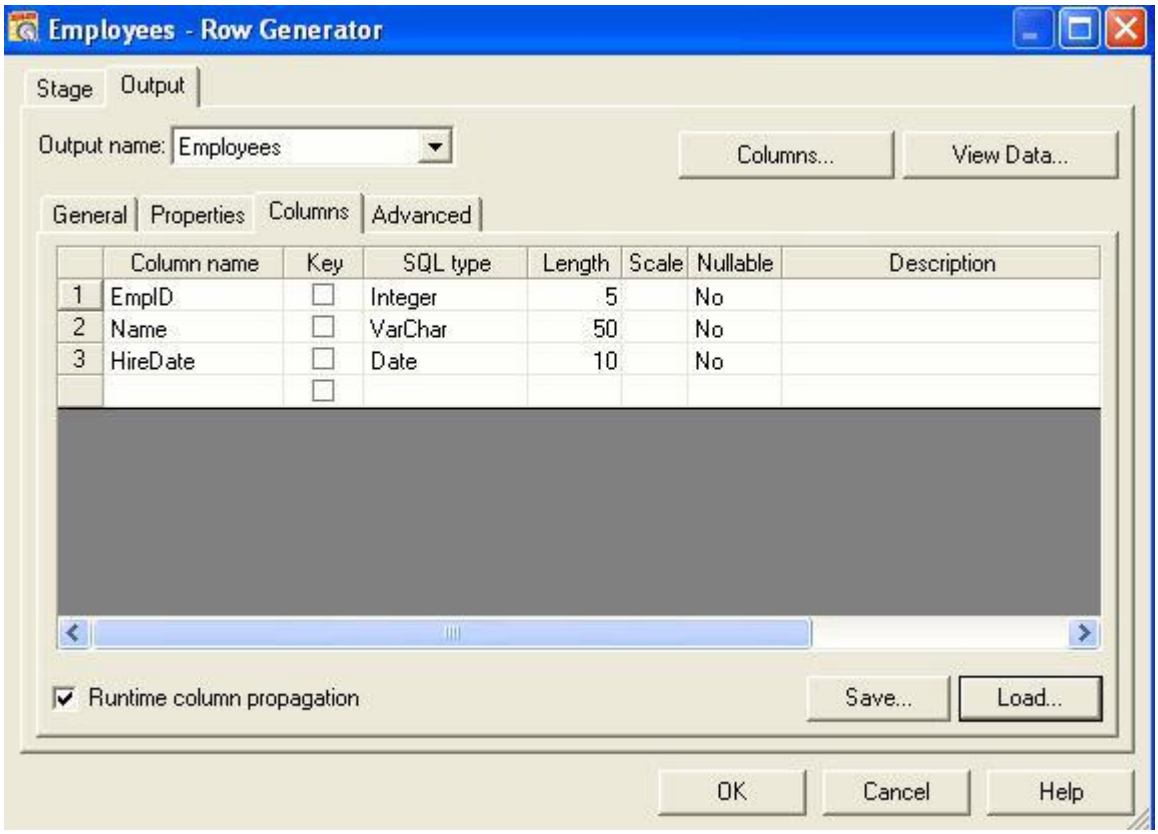
1. Open a new Parallel job and save it under the name GenDataJob. Save it your _Training>Jobs folder.
2. Add a Row Generator stage and a Peek stage.
3. Draw a link from the Row Generator stage to the Peek stage. Change the names of the stages and links as shown.



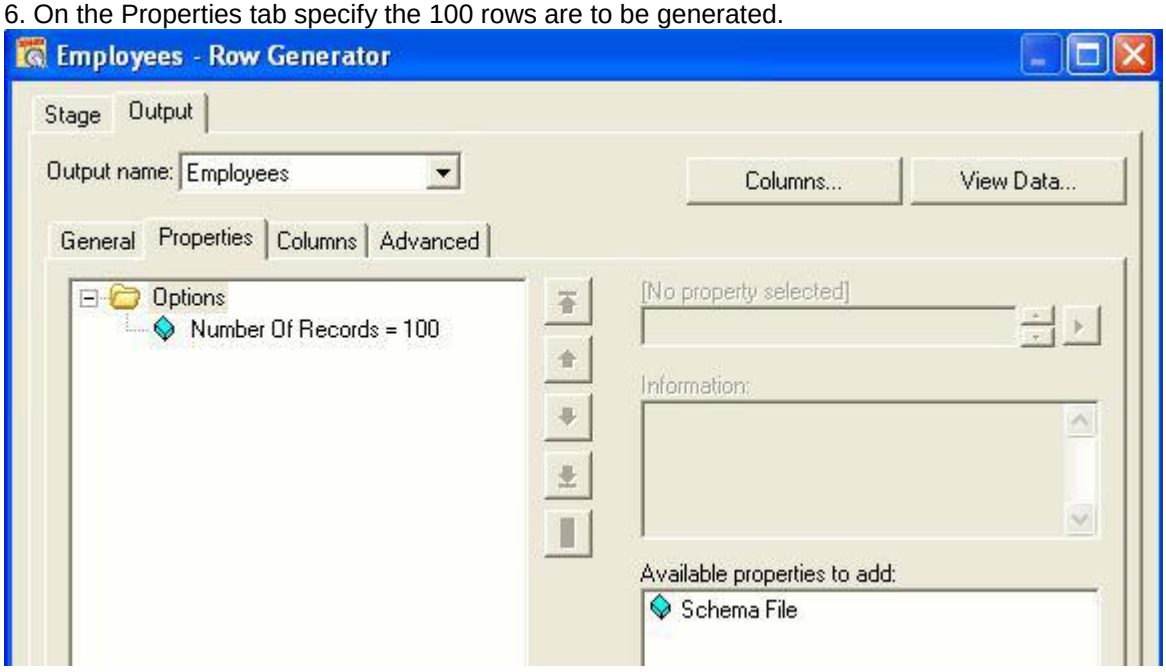
4. Open up the Row Generator stage to the Columns tab. Click the Load button to load the column definitions from the Employees Table definition you imported in an earlier lab.



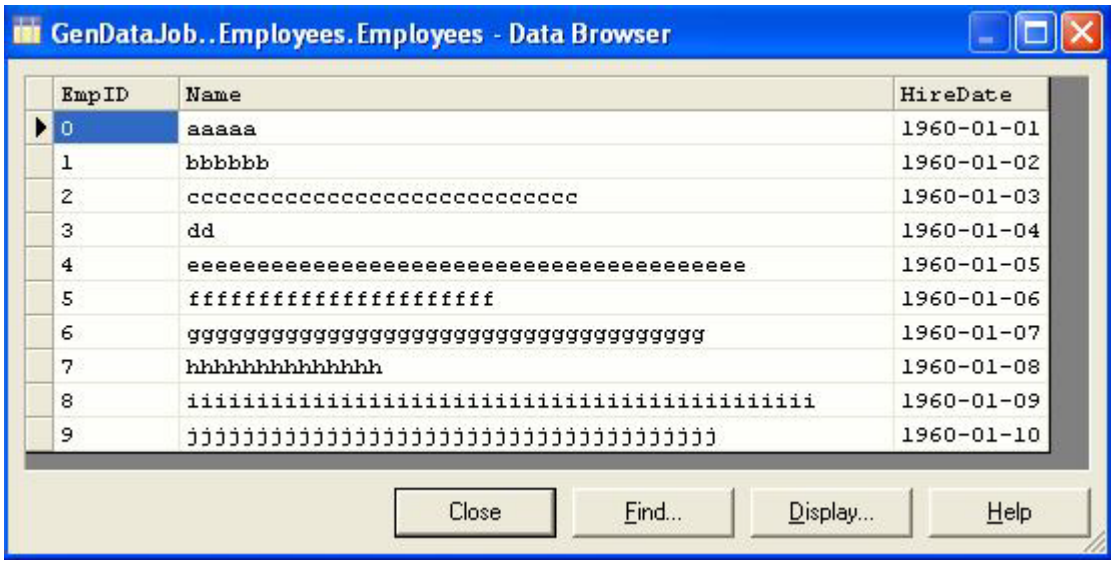
5. Verify your column definitions with the following.



6. On the Properties tab specify the 100 rows are to be generated.

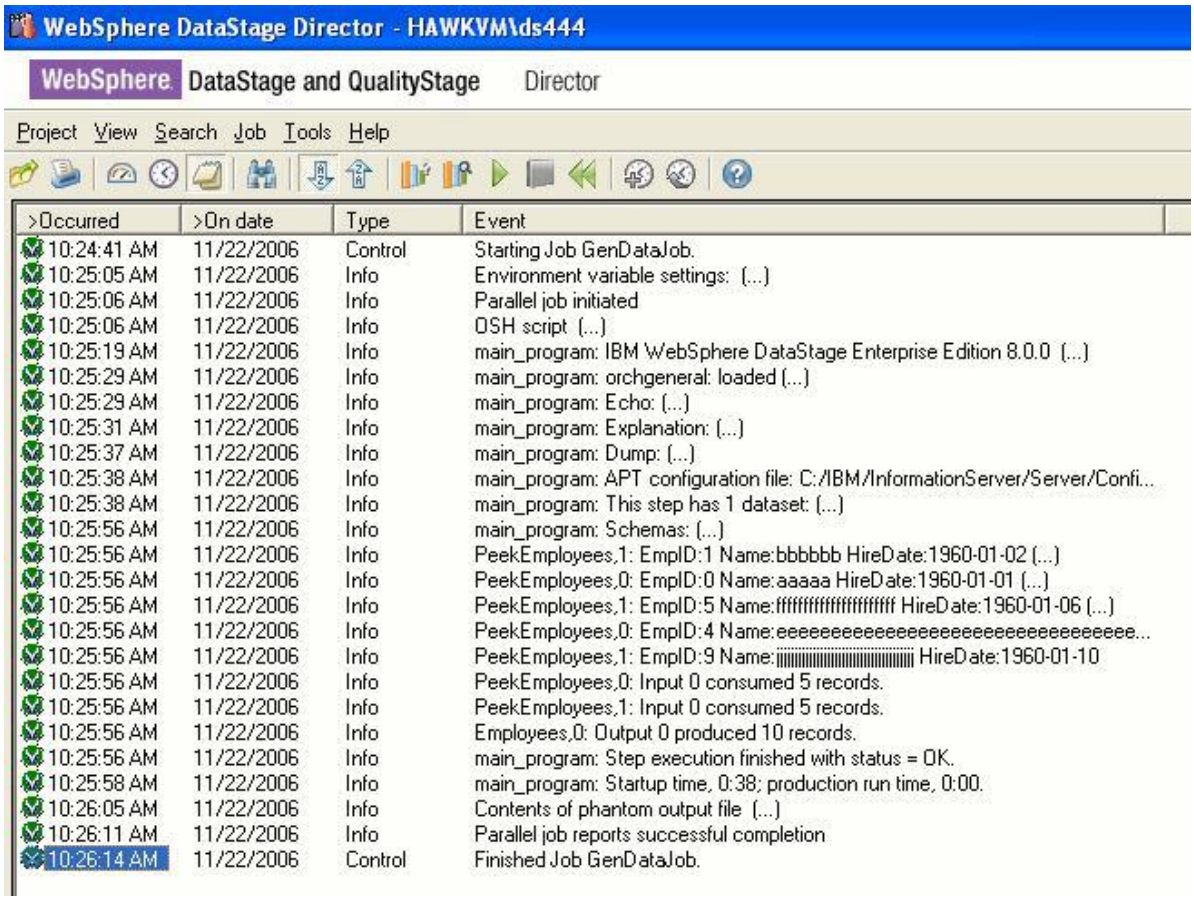


7. Click View Data to view the data that will be generated.

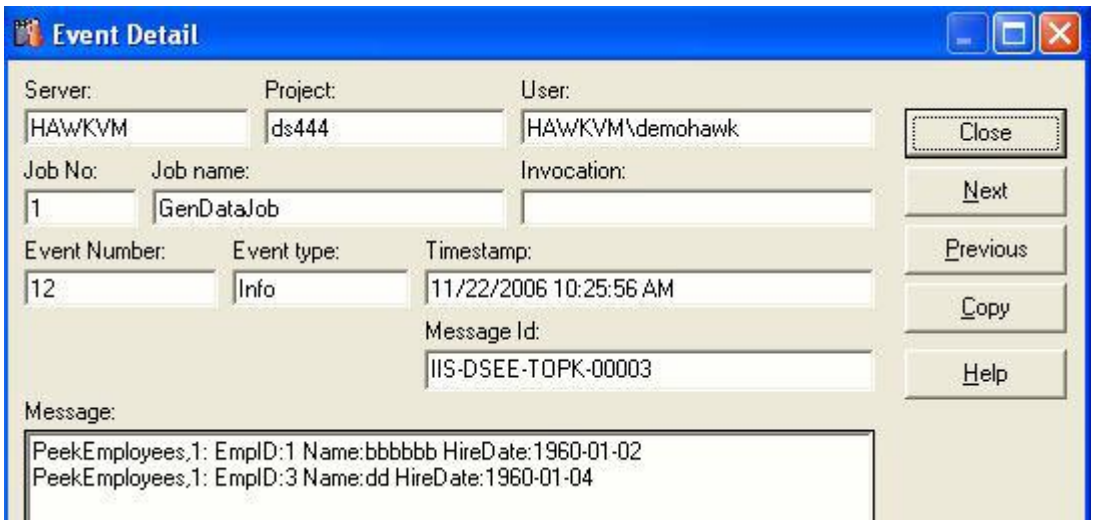


Task: Compile, run, and monitor the job

1. Compile your job.
2. Click your right mouse button over an empty part of the canvas. Select or verify that "Show performance statistics" is enabled.
3. Run your job.
4. Move to Director from within Designer by clicking Tools>Run Director.
5. In the Director Status window select your job.
6. Move to the job log.
7. Scroll through the messages in the log. There should be no warnings (yellow) or errors (red). If there are, double-click on the messages to examine their contents. Fix the problem and then recompile and run.

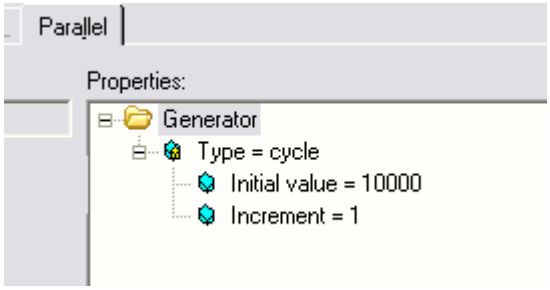


8. Notice that there are one or more log messages starting with “PeekEmployees,” the label on your Peek stage. Double click on one of these to open the message window.

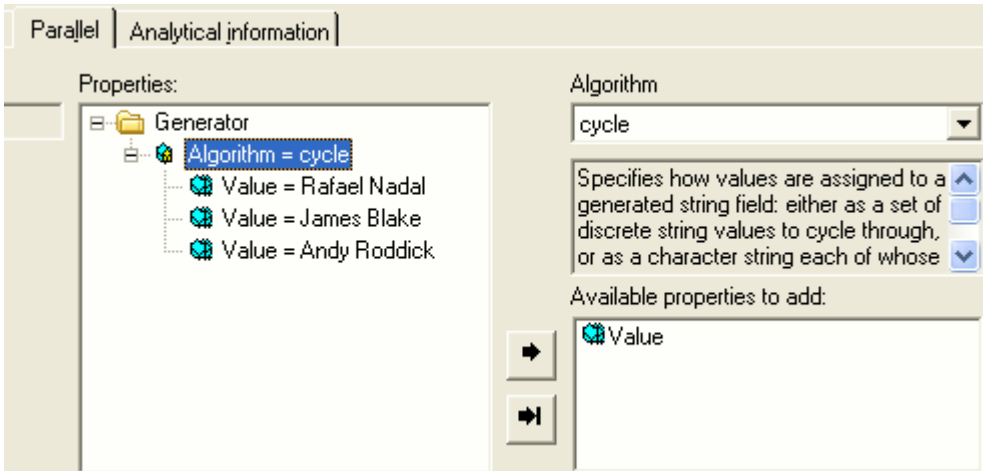


TASK: Specify Extended Properties

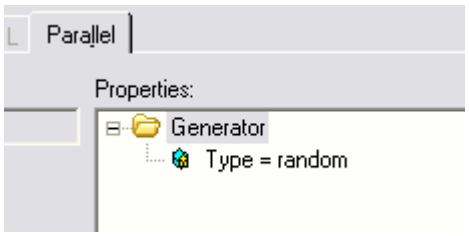
1. Save your job as GenDataJobAlgor in your _Training>Jobs folder.
2. Open up the Row Generator stage to the Columns tab. Double-click on the row number to the left of the first column name.
3. Specify the Extended Properties as shown.



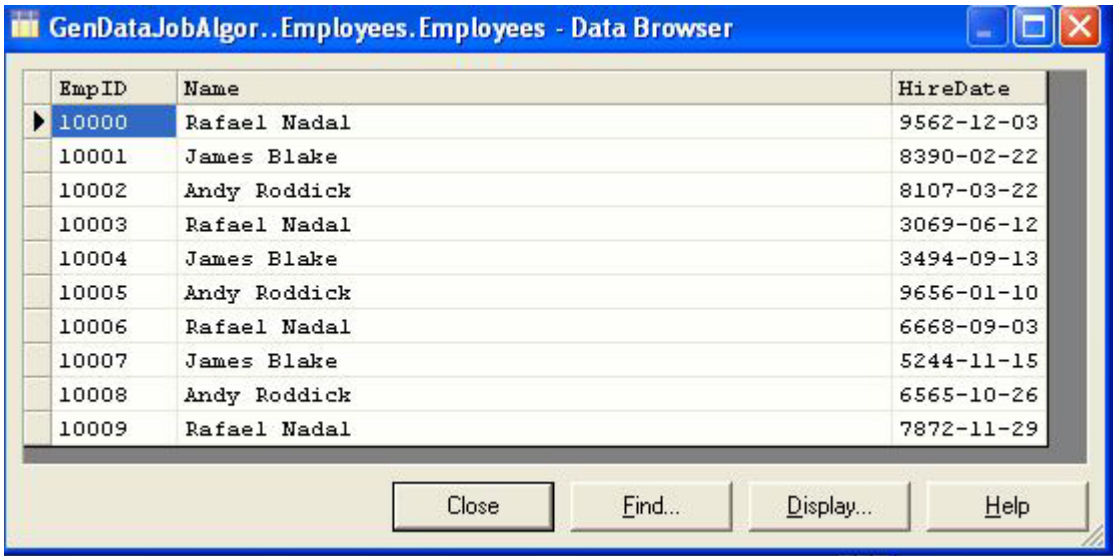
4. For the Name column specify that you want to cycle through 3 names, your choice.



5. For the HireDate column, specify that you want the dates generated randomly.

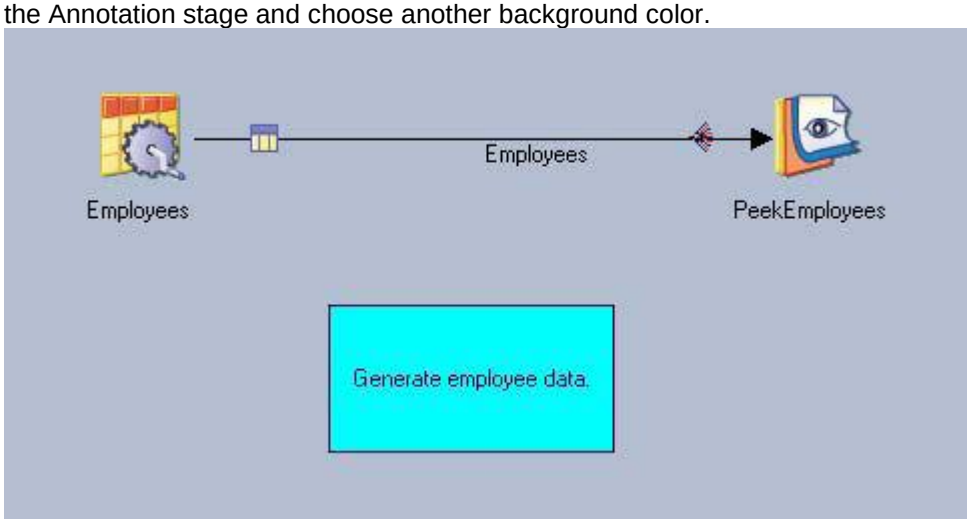


6. Click View data to see the data that will be generated.

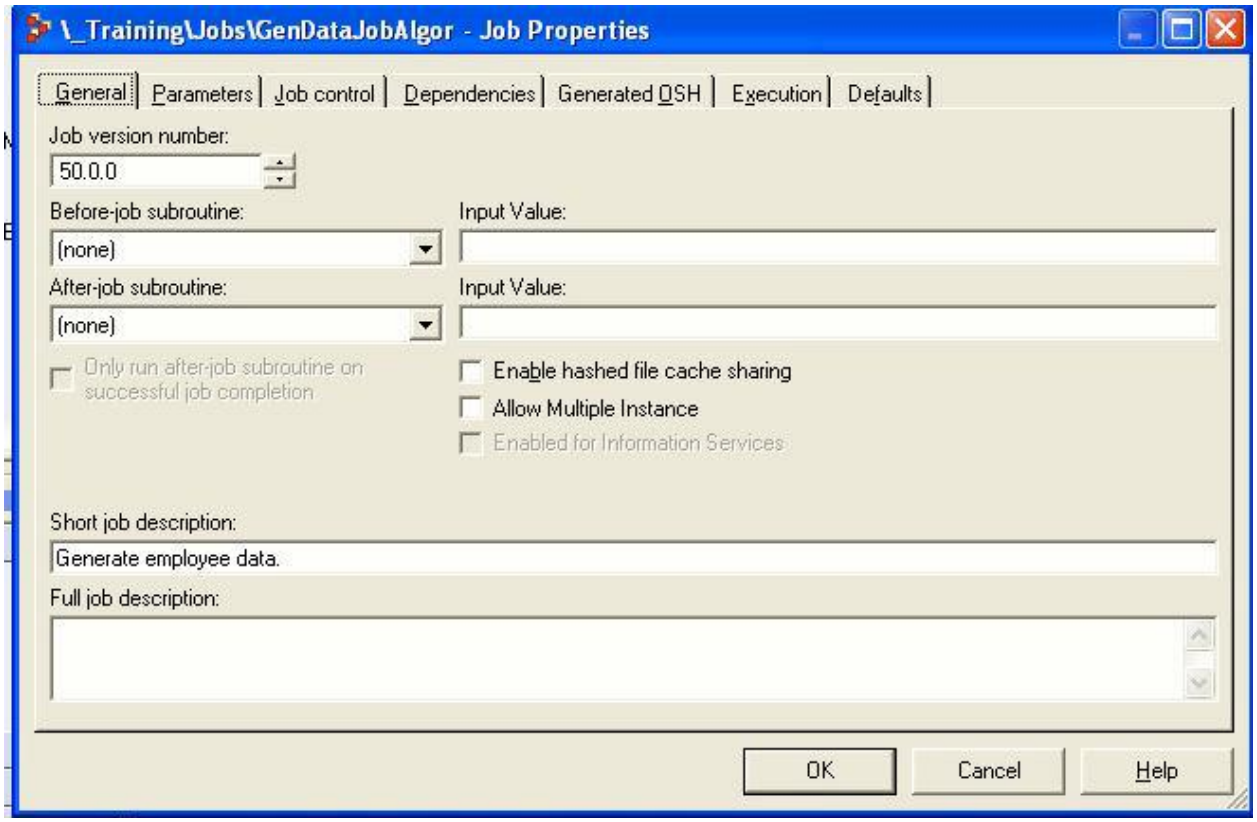


TASK: Document your job

1. Add an Annotation stage to your job diagram that describes what your job does. Open up the Annotation stage and choose another background color.



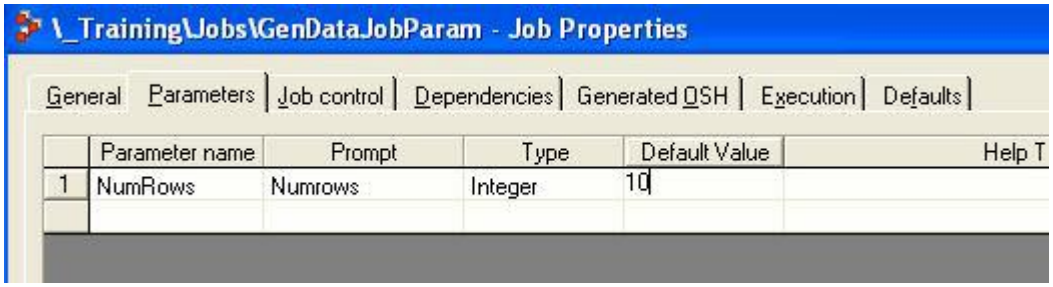
2. In the Job Properties window, short description, also briefly describe what your job does.



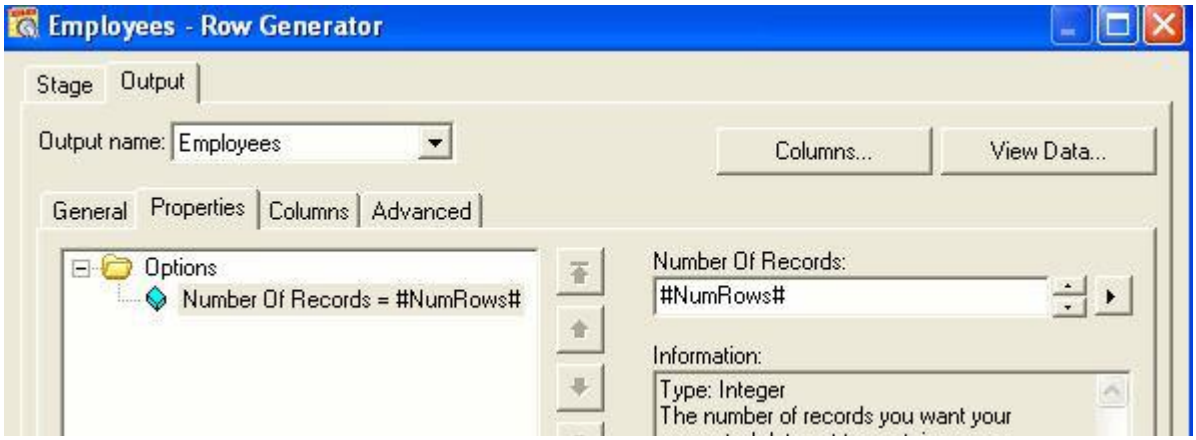
3. Compile and run your job. View the messages in the Director log. Fix any warnings or errors.
4. Verify the data by examining the Peek stage messages in the log.

Task: Add a job parameter

1. Save your job as GenDataJobParam in your _Training>Jobs folder.
2. Open up the Job Properties window. Click on the Parameters tab.
3. Define a new parameter named NumRows with a default value of 10. Its type is Integer.



4. Open up the Properties tab of the RowGenerator stage in your job. Use your NumRows job parameter to specify the number of rows to generate.



5. View the data.
6. Compile and run your job. Verify the results.

Task: Backup your project

1. Select your _Training folder.
2. Export the contents of your folder to your Training.dsx file

Lab 03: Accessing Sequential Data

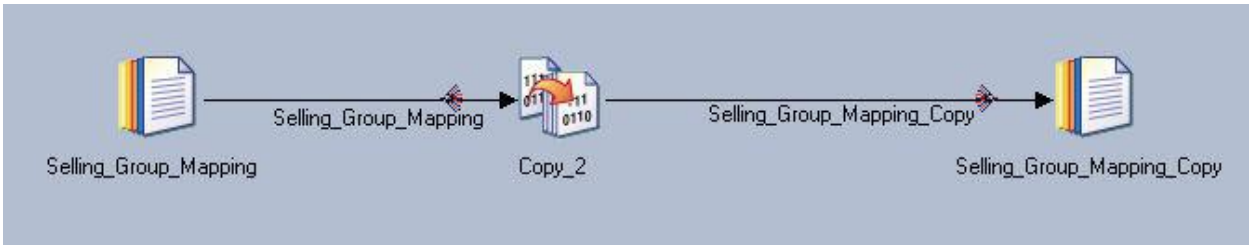
Assumptions:

- Completed PPT training session for Lession 4.1 to 4.5

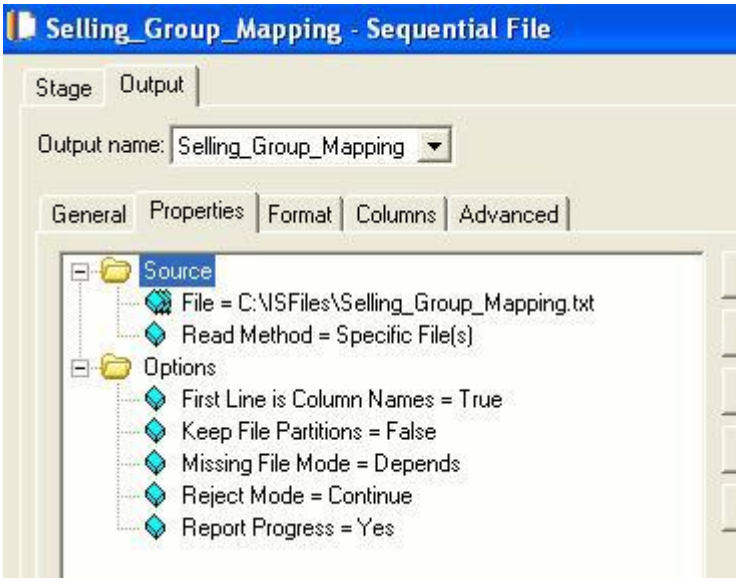
Task: Read and write to a sequential file

In this task, you design a job that reads data from the Selling_Group_Mapping.txt file, copies it
through a Copy stage, and then writes the data to a new file named Selling_Group_Mapping_Copy.txt.

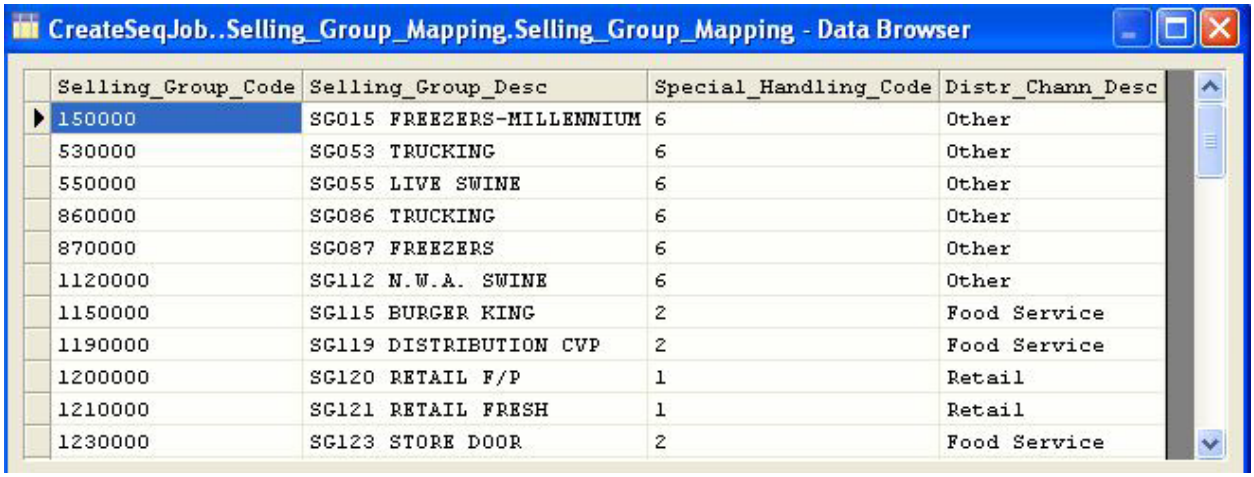
1. Open a new Parallel job and save it under the name CreateSeqJob. Save it into your _Training>Jobs folder.
2. Add a Sequential stage, a Copy stage, and a second Sequential stage. Draw links. Name the stages and links as shown.



3. In the Sequential source stage Columns and Format tabs, load the format and column definitions from the Selling_Group_Mapping.txt table definition you imported in a previous exercise.
4. On the Properties tab specify the file to read and other relevant properties. Here be sure to set the First Line is Column Names to True. If you don't your job will have trouble reading the first row and issue a warning message in the Director log.

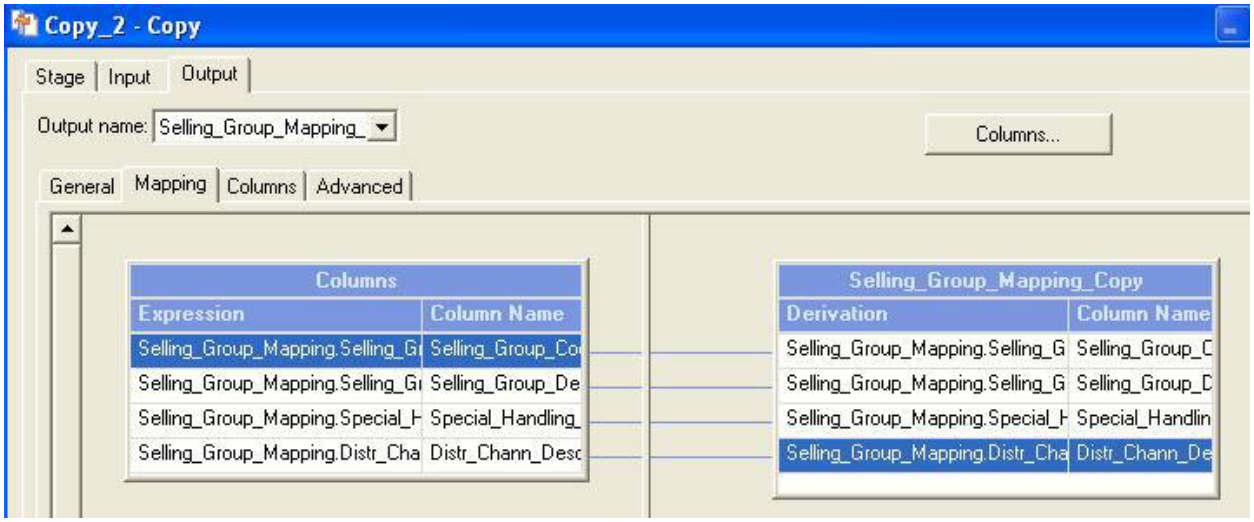


5. Click View data to view the data to verify that the metadata has been specified properly.

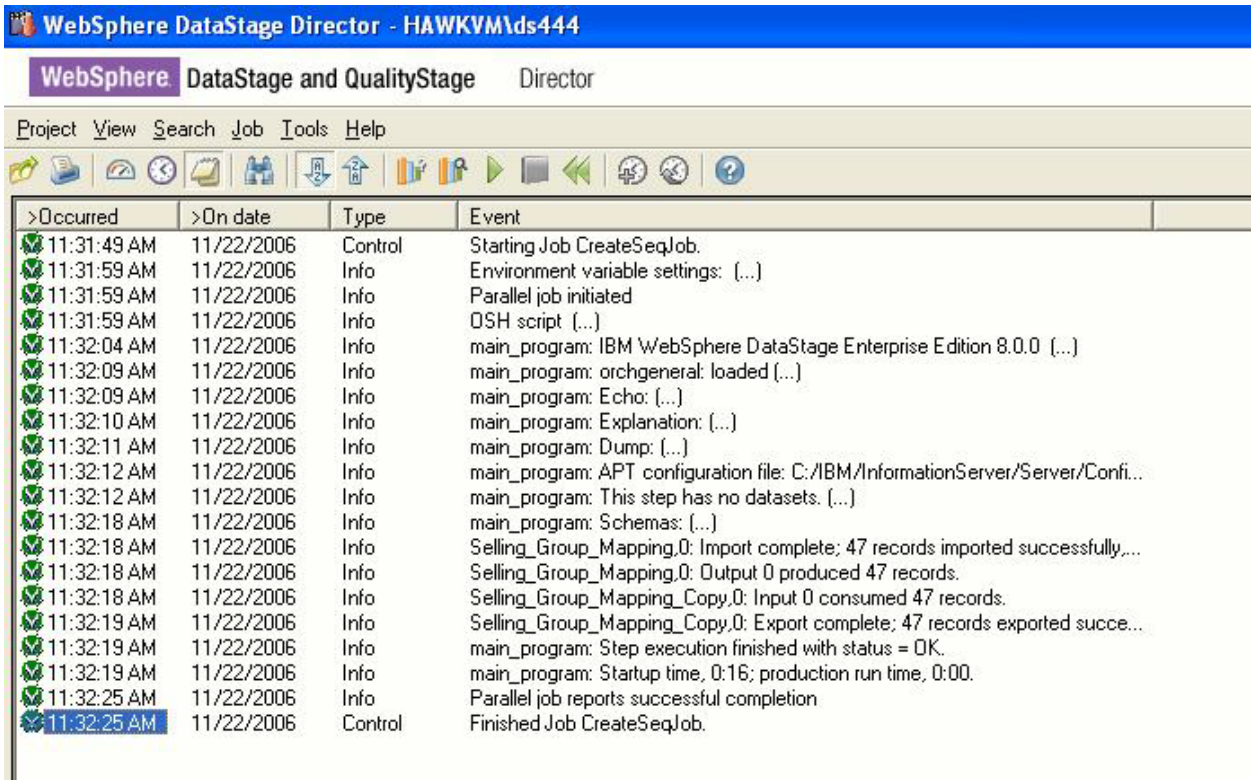


Selling_Group_Code	Selling_Group_Desc	Special_Handling_Code	Distr_Chann_Desc
150000	SG015 FREEZERS-MILLENNIUM	6	Other
530000	SG053 TRUCKING	6	Other
550000	SG055 LIVE SWINE	6	Other
860000	SG086 TRUCKING	6	Other
870000	SG087 FREEZERS	6	Other
1120000	SG112 N.W.A. SWINE	6	Other
1150000	SG115 BURGER KING	2	Food Service
1190000	SG119 DISTRIBUTION CVP	2	Food Service
1200000	SG120 RETAIL F/P	1	Retail
1210000	SG121 RETAIL FRESH	1	Retail
1230000	SG123 STORE DOOR	2	Food Service

6. In the Copy stage Output>Mapping tab, drag the columns across from the source to the target.

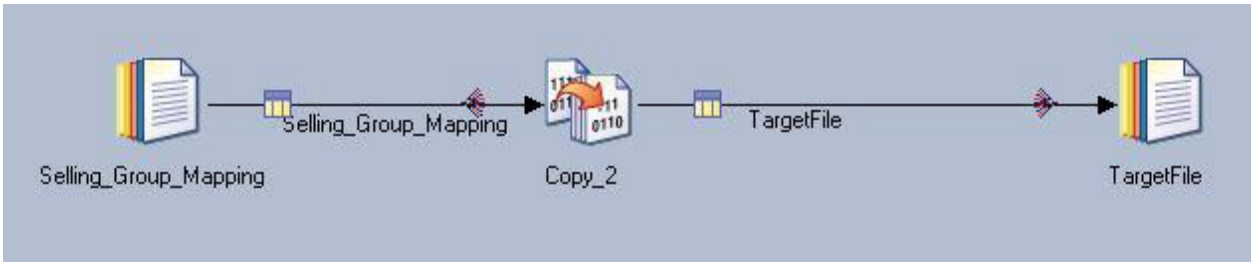


7. In the target Sequential stage, create a delimited file with the comma as the delimiter. Name the file the Selling_Group_Mapping_Copy.txt and write to your ISFiles>Temp directory. Create it with a first line of column names. It should overwrite any existing file with the same name.
8. Compile and run your job.
9. In Director, view the job log. Fix any errors.

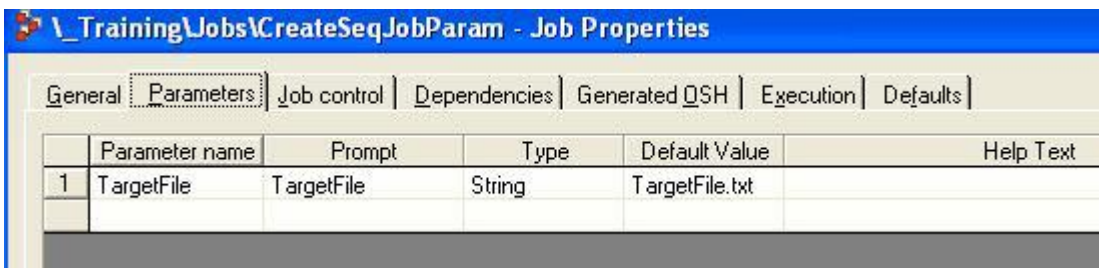


Task: Create and use a job parameter

1. Save your CreateSeqJob job as CreateSeqJobParam. Rename the last link and Sequential File stage to “TargetFile”.

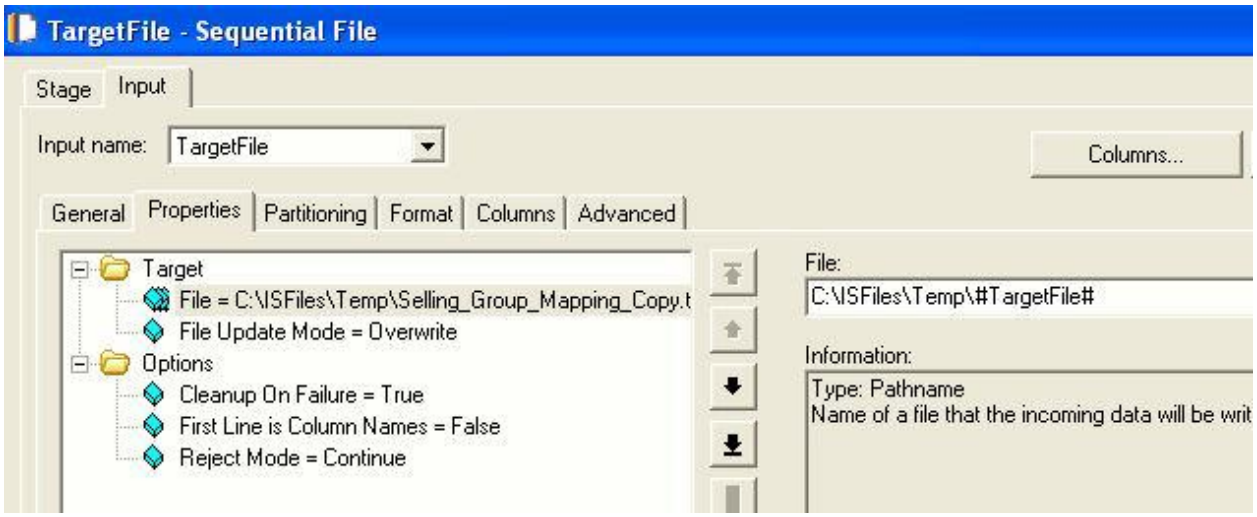


2. Open up the job properties window.
3. On the parameters tab, define a job parameter named TargetFile of type string. Create an appropriate default filename, e.g., TargetFile.txt.



4. Open up your target stage to the Properties tab. Select the File Property. In the File text box, replace the name of your file by your job parameter. Retain the rest of your file

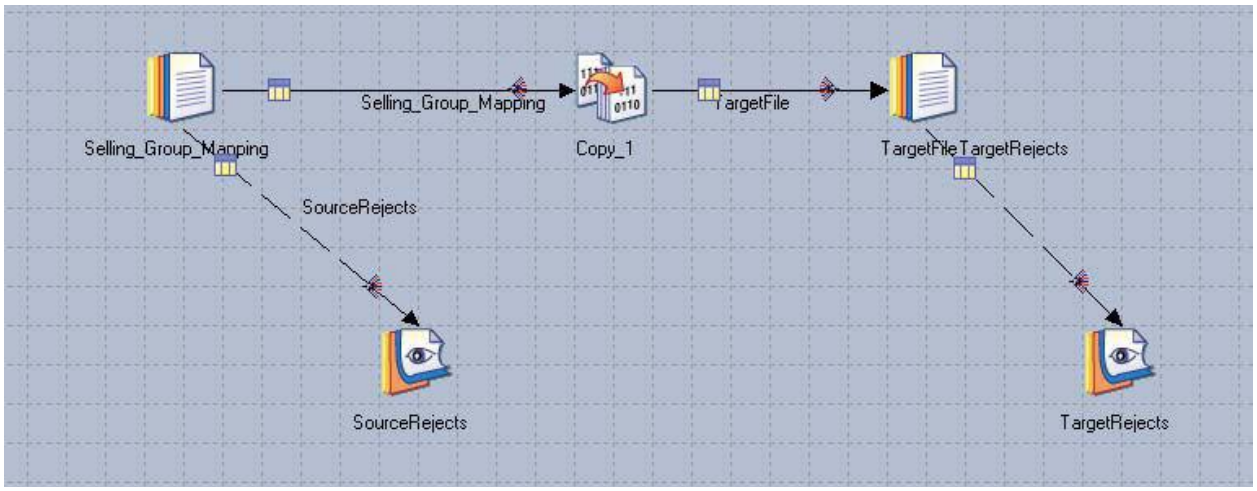
path. **(**IMPORTANT:** In some releases of DataStage there is a bug regarding the use of job parameters in Sequential File stages. The fix is to replace the back slashes in the path by forward slashes, e.g., C:/ISFiles/Temp/#TargetFile#.)



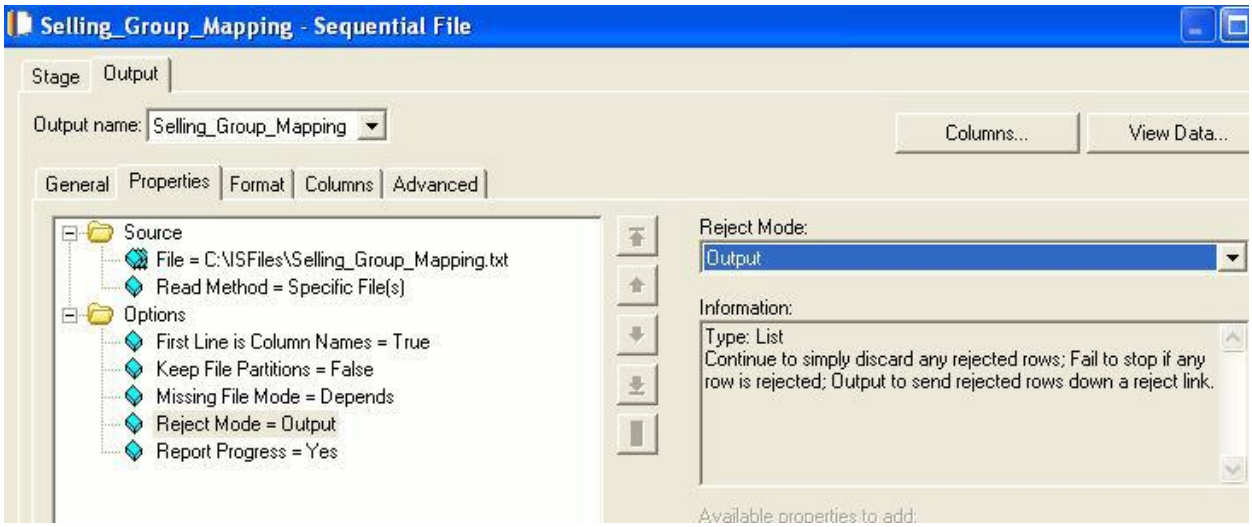
Task: Add Reject links

1. Add a second link (which will automatically become a reject link) from the source Sequential File stage to a Peek stage. Also add an output link from the target Sequential

File stage to a Peek stage. Give appropriate names to these new stages and links.

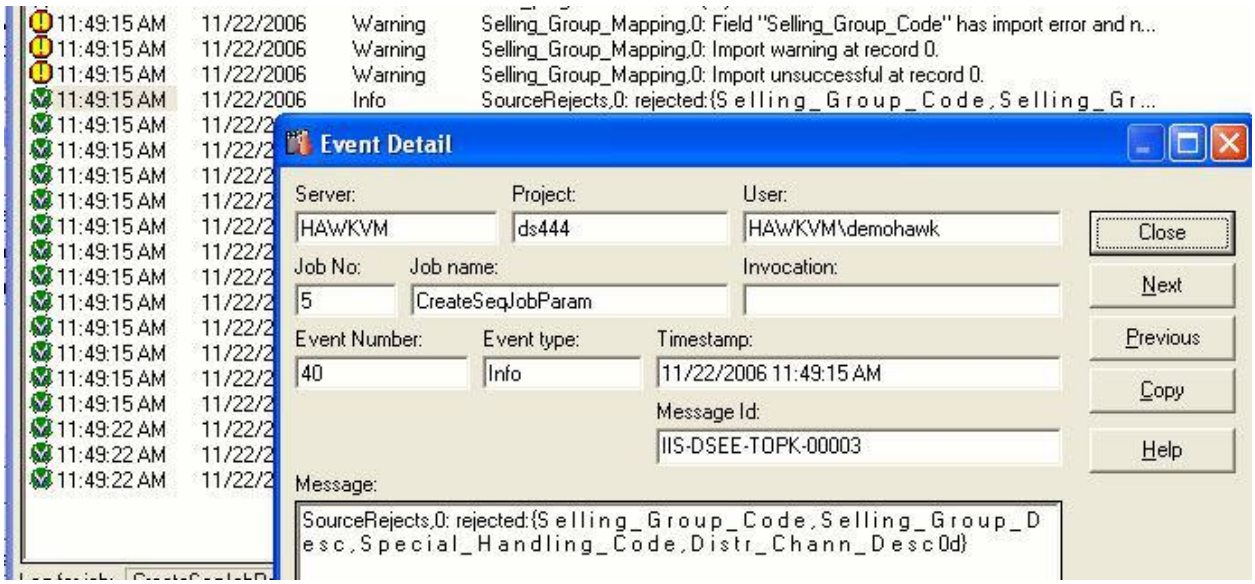


2. On the Properties tab of each Sequential File stage, change the Reject Mode property value to “Output”.

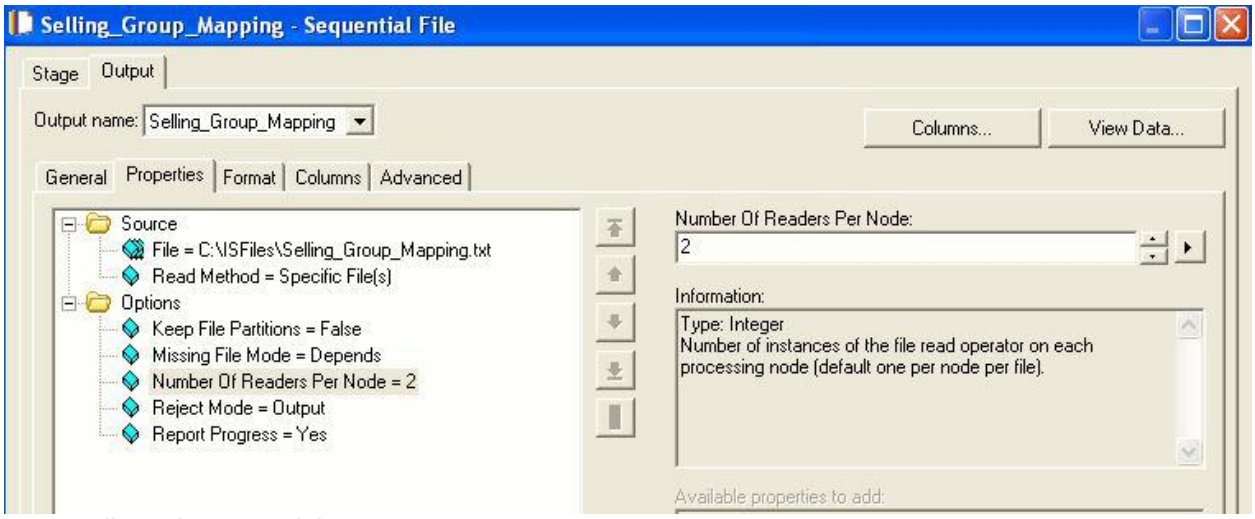


3. Compile and run. Verify that it's running correctly. You shouldn't have any rejects or errors or warnings.
4. To test the rejects link, temporarily change the property First Line is Column Names to False in the Source stage and then recompile and run. This will cause the first row to be rejected because the values in the first row, which are all strings, won't fit the column definitions, some of which are integers.

5. Examine the DataStage log. What row shows up in the Peek message? Examine the warning messages before the Peek. Note the number of rows that are successfully imported and how many are rejected.



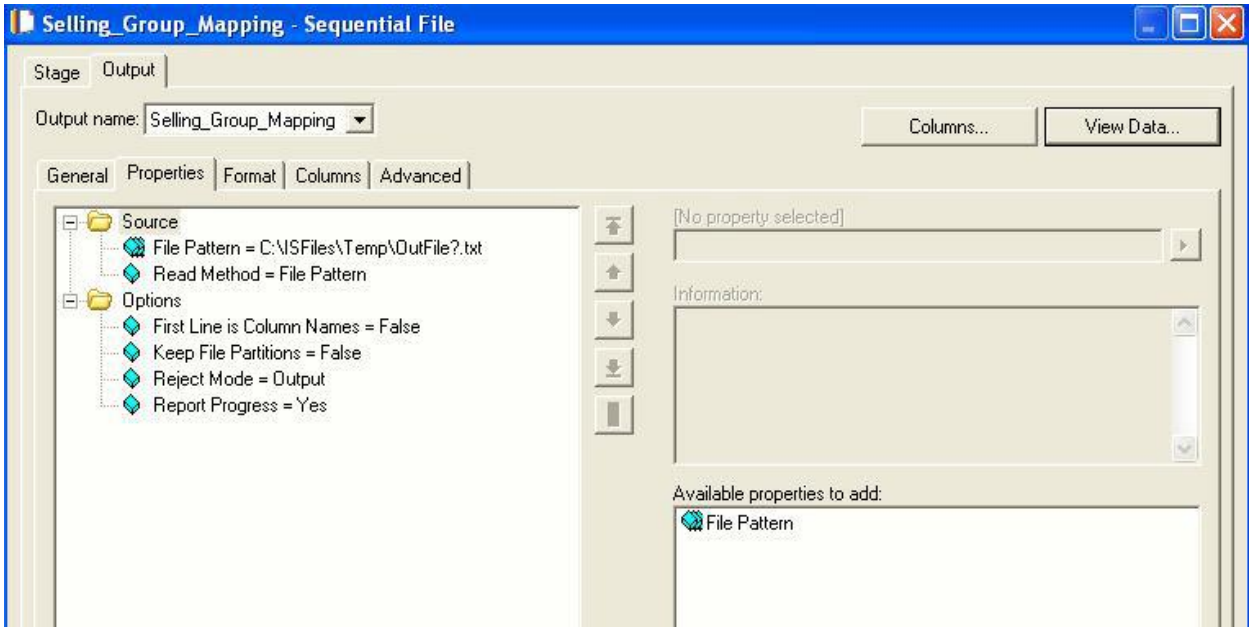
Task: Read a file using multiple readers
1. Save your job as CreateSeqJobMultiRead.
2. Click the Properties tab of your source stage.
3. Click the Options folder and add the "Number of Readers Per Node" property. Set this property to 2.



4. Compile and run your job.
5. View the job log. (**Note:** You will receive some warning messages related to the first Columns names row. And this row will be rejected. You can ignore these.)

Task: Create job reading multiple files

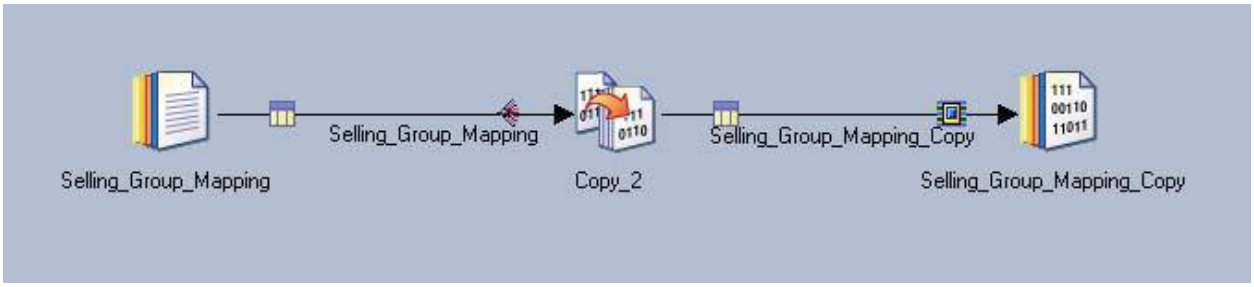
1. Save your job as CreateSeqJobPattern.
2. Run your job twice specifying the following file names in the job parameter for the target file: OutFileA.txt, OutFileB.txt.
3. Edit the source Sequential stage:
• Change read method to File Pattern
• Place a wildcard (?) in the last portion of the file name:
OutFile?.txt



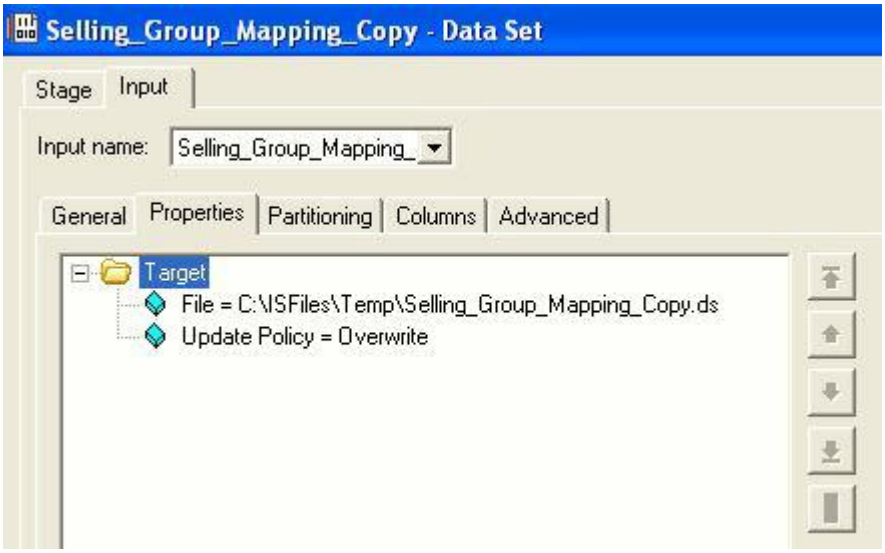
4. Click View Data to verify that you can read the files.
5. Compile and run the job. View the job log.
6. Verify the results. There should be two copies of each row, since you are now reading two identical files.

Working with Data Sets
Task: Create a data set

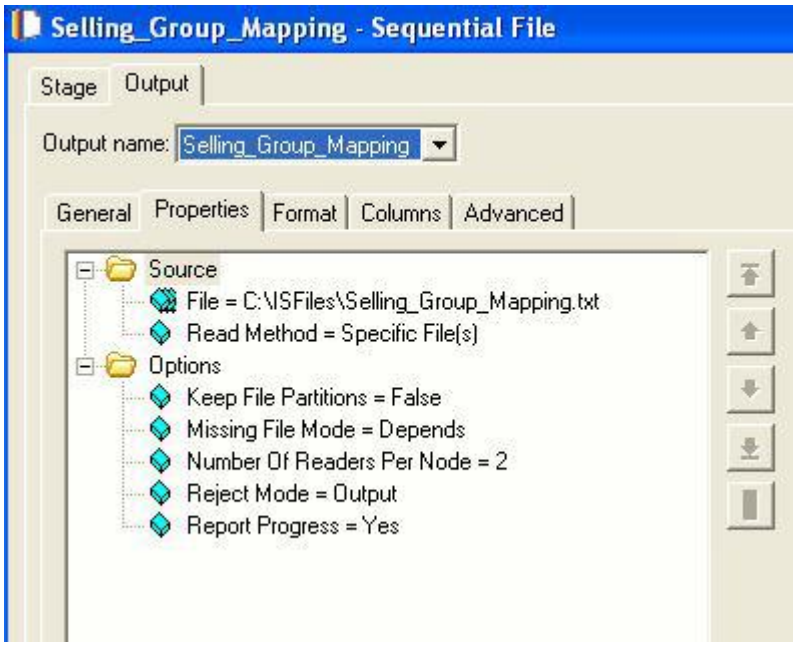
1. Open up your CreateSeqJob job and save it as CreateDataSetJob.
2. Delete the target sequential stage leaving a dangling link.
3. Add a DataSet stage and connect it to the dangling link. Change the name of the target stage as shown.



4. Edit the Data Set stage properties. Write to a file named Selling_Group_Mapping.ds in your ISFiles>Temp directory.



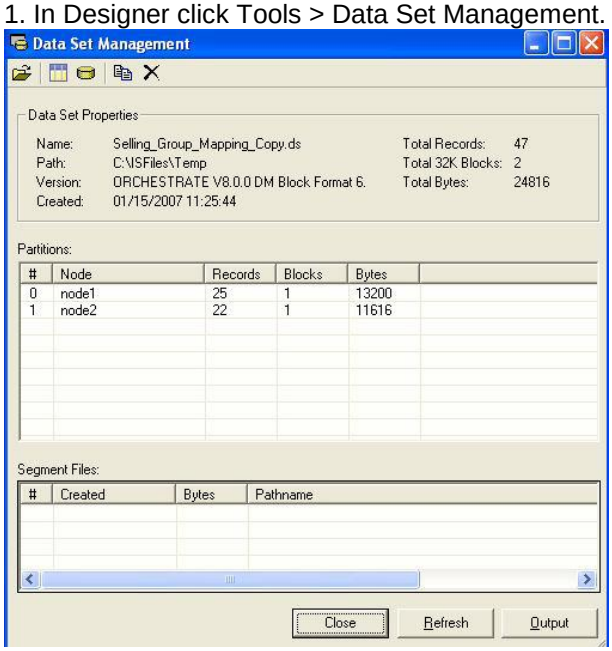
5. Open the source stage and add the optional property to read the file using multiple readers. Change the value of the property to 2. (This will insure that data is written to more than one partition.)



6. Check, and if necessary complete, the Copy stage columns mapping.
7. Compile your job.
8. Run your job. Check the Director log for errors.

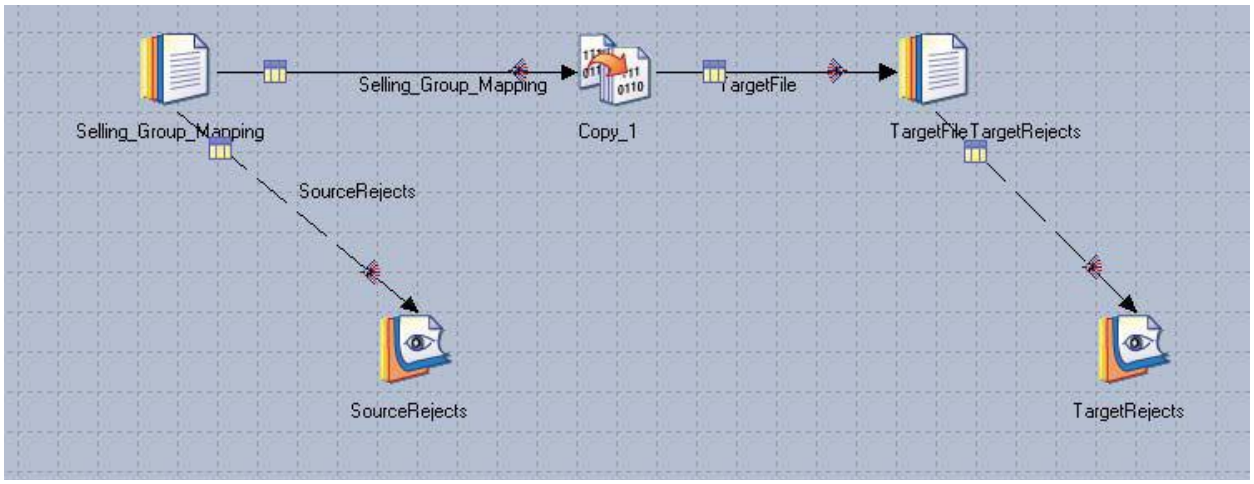
Task: View a dataset

1. In Designer click Tools > Data Set Management. Select your dataset.

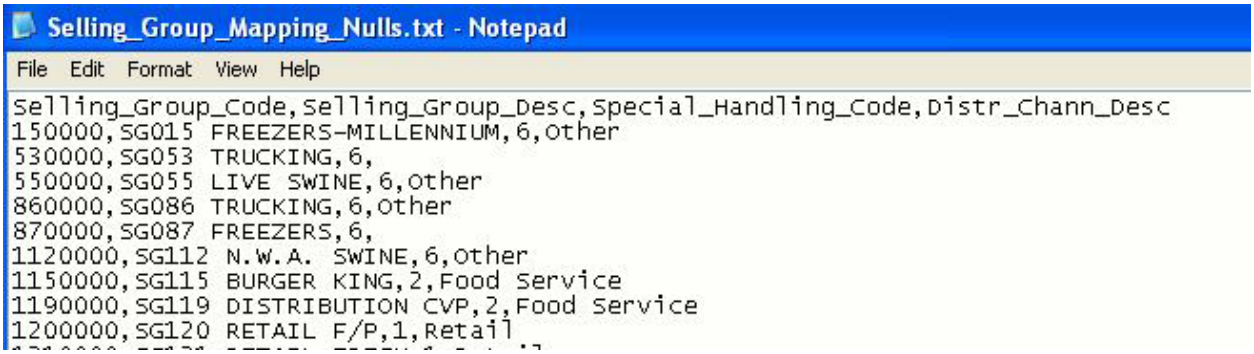


2. Click the Show Data Window icon at the top of the window to view the dataset data.
3. Click the Show Schema Window icon at the top of the window to view the dataset schema.

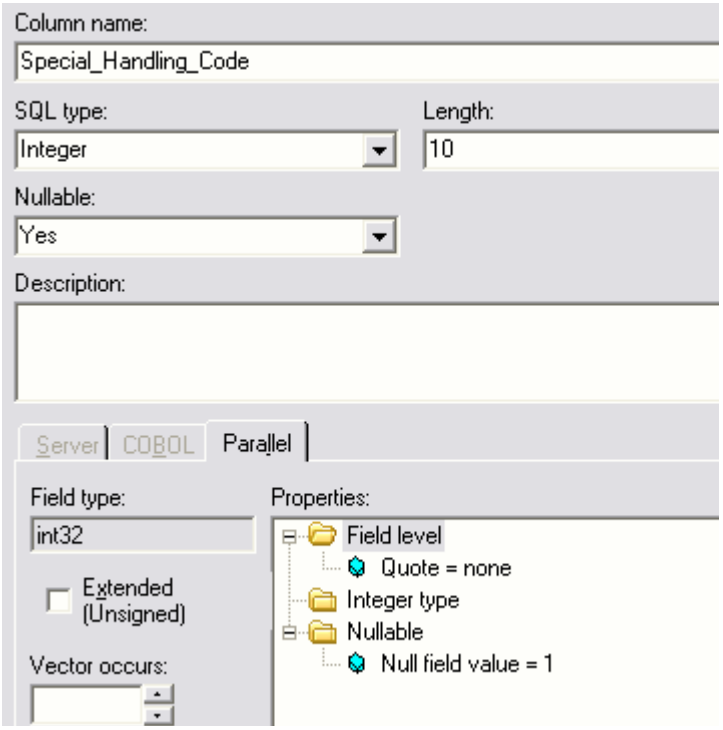
Reading and Writing NULLs
Task: Read values meaning NULL from a sequential file
1. Save your CreateSeqJobParam job as CreateSeqJobNULL



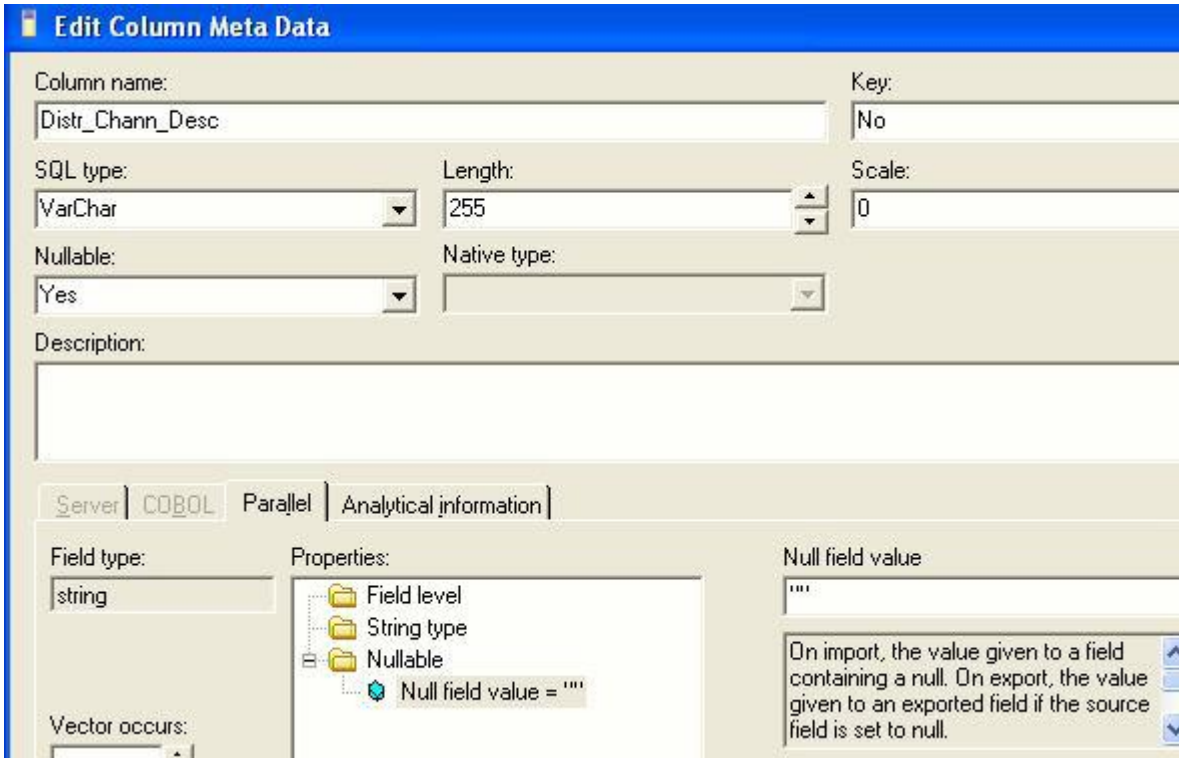
2. Open up the Selling_Group_Mapping_Nulls.txt file in your ISFiles directory.



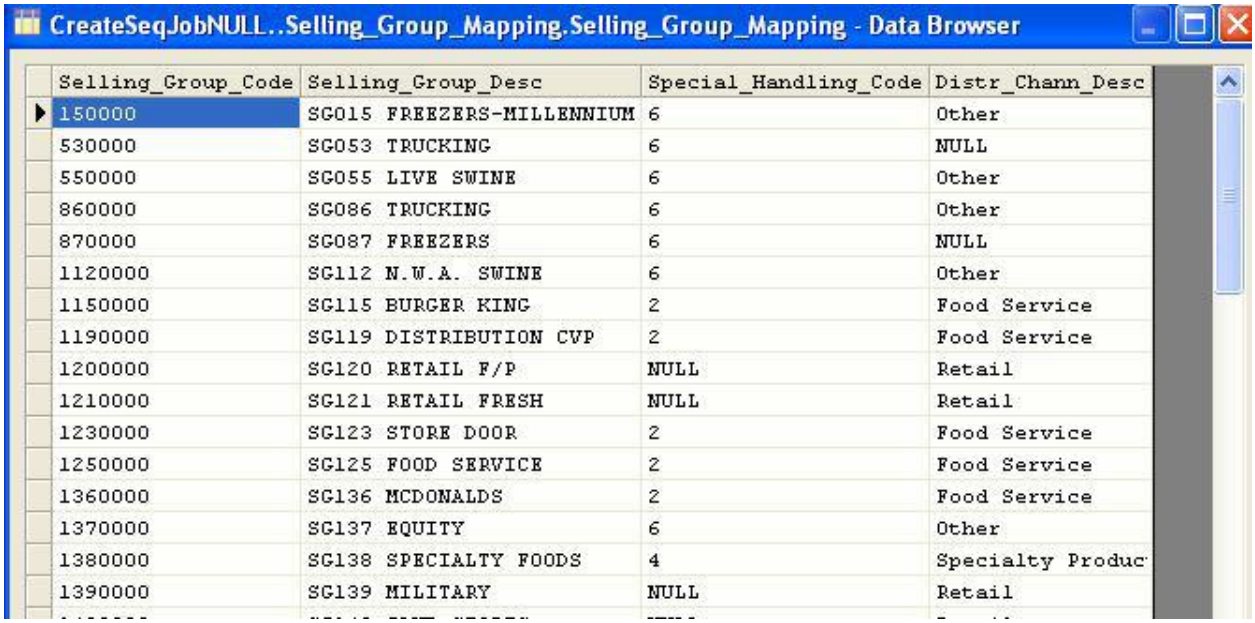
3. Notice in the data that the Special_Handling_Code column contains some integer values of 1. Notice also that the last column (Distr_Chann_Desc) is missing some values. To test how to read NULLs, let's assume that 1 in the third column means NULL and that nothing in the last column means NULL. In the next step we will specify this.
4. Open up the source Sequential stage to the columns tab. Double-click to the left of the Special_Handling_Code column to open up the Edit Column Meta Data window.
5. Change the field to nullable. Notice that the Nullable folder shows up in the Properties window. Select this folder and then add the Null field value property. Specify a value of 1 for it.



6. Click Apply.
7. Move to the Distr_Chann_Desc column. Set this field to nullable. Add the Null field value property. Here, we will treat the empty string as meaning NULL. To do this specify “” back-to-back quotes.

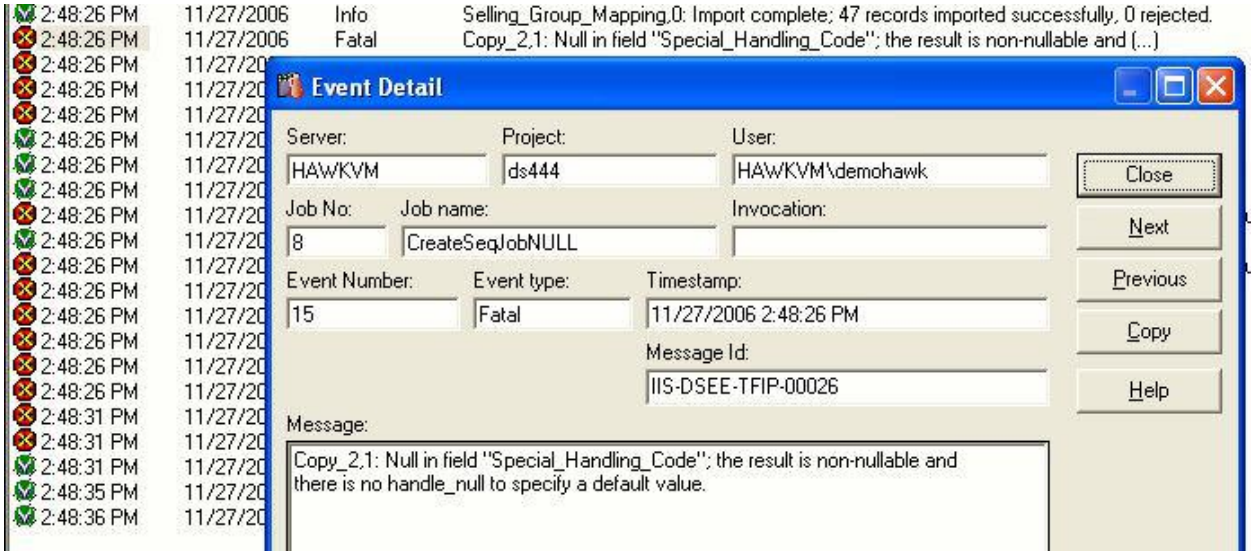


8. On the Propeties tab, select the Selling_Group_Mapping_Nulls.txt file to read.
9. Click the View Data button. Notice that values that are interpreted by DataStage as NULL show up as the word “NULL”.



Selling_Group_Code	Selling_Group_Desc	Special_Handling_Code	Distr_Chann_Desc
150000	SG015 FREEZERS-MILLENNIUM	6	Other
530000	SG053 TRUCKING	6	NULL
550000	SG055 LIVE SWINE	6	Other
860000	SG086 TRUCKING	6	Other
870000	SG087 FREEZERS	6	NULL
1120000	SG112 N.W.A. SWINE	6	Other
1150000	SG115 BURGER KING	2	Food Service
1190000	SG119 DISTRIBUTION CVP	2	Food Service
1200000	SG120 RETAIL F/P	NULL	Retail
1210000	SG121 RETAIL FRESH	NULL	Retail
1230000	SG123 STORE DOOR	2	Food Service
1250000	SG125 FOOD SERVICE	2	Food Service
1360000	SG136 MCDONALDS	2	Food Service
1370000	SG137 EQUITY	6	Other
1380000	SG138 SPECIALTY FOODS	4	Specialty Produc
1390000	SG139 MILITARY	NULL	Retail

10. Compile and run your job. It should abort since NULL values will be written to nonnullable columns on your target.

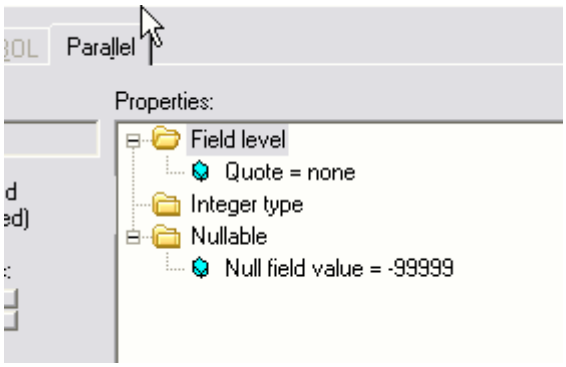


Task: Write values meaning NULL to a sequential file

1. Save your job as CreateSeqJobHandleNULL.
2. Open up your target stage to the Columns tab. Specify that the Special_Handling_Code column and the Distribution_Channel_Description column are nullable.
3. Compile and run your job. What happens?
4. In this case, the job doesn’t abort, since NULL values aren’t being written to a nonnullable column. But the rows with NULL values get rejected because the NULL values “aren’t being handled.” Notice that they are written to the Peek stage, where you can view them.

2:54:44 PM	11/27/2006	Warning	Selling_Group_Mapping_Copy,0: Export was unsuccessful at record 20; continuing.
2:54:44 PM	11/27/2006	Warning	Selling_Group_Mapping_Copy,0: Field "Special_Handling_Code" is null but no null export handling is defined
2:54:44 PM	11/27/2006	Warning	Selling_Group_Mapping_Copy,0: Export was unsuccessful at record 21; continuing.
2:54:44 PM	11/27/2006	Info	SourceRejects,0: Input 0 consumed 0 records.
2:54:44 PM	11/27/2006	Warning	Selling_Group_Mapping_Copy,0: Field "Special_Handling_Code" is null but no null export handling is defined
2:54:44 PM	11/27/2006	Warning	Selling_Group_Mapping_Copy,0: Export was unsuccessful at record 28; continuing.
2:54:44 PM	11/27/2006	Info	Selling_Group_Mapping_Copy,0: No further reports will be generated from this partition until a successful export.
2:54:44 PM	11/27/2006	Info	Selling_Group_Mapping,0: Output 0 produced 47 records. (...)
2:54:44 PM	11/27/2006	Info	SourceRejects,1: Input 0 consumed 0 records.
2:54:44 PM	11/27/2006	Info	Selling_Group_Mapping_Copy,0: Export complete; 38 records exported successfully, 9 rejected.
2:54:44 PM	11/27/2006	Info	TargetRejects,1: Selling_Group_Code:1390000 Selling_Group_Desc:SG139 MILITARY Special_Handling_Code:NULL
2:54:44 PM	11/27/2006	Info	TargetRejects,0: Selling_Group_Code:1210000 Selling_Group_Desc:SG121 RETAIL FRESH Special_Handling_Code:

5. Now, let's handle the NULL values. That is, we will specify values to be written to the target file that represent NULLs. For the Special_Handling_Code column we will specify a value of -99999. For the Distribution_Channel_Description column we will specify a value of "UNKNOWN".
6. Open up the target stage and specify these values. The procedure is the same as when the Sequential stage is used as a source.



7. Compile and run your job. View the job log. You should get no errors or warnings or rejects.
8. Verify the results by viewing your target file on the DataStage Server. (Note: if you view the data in DataStage, all you will see is the word "NULL", not the actual value that means NULL.)

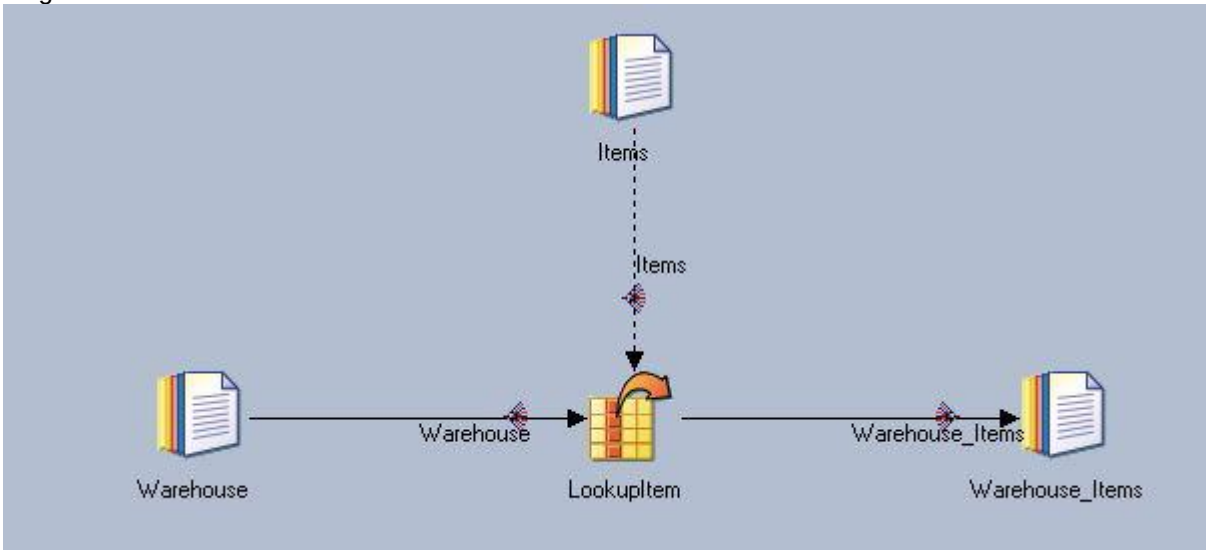
Lab 04: Combining data

Assumptions:

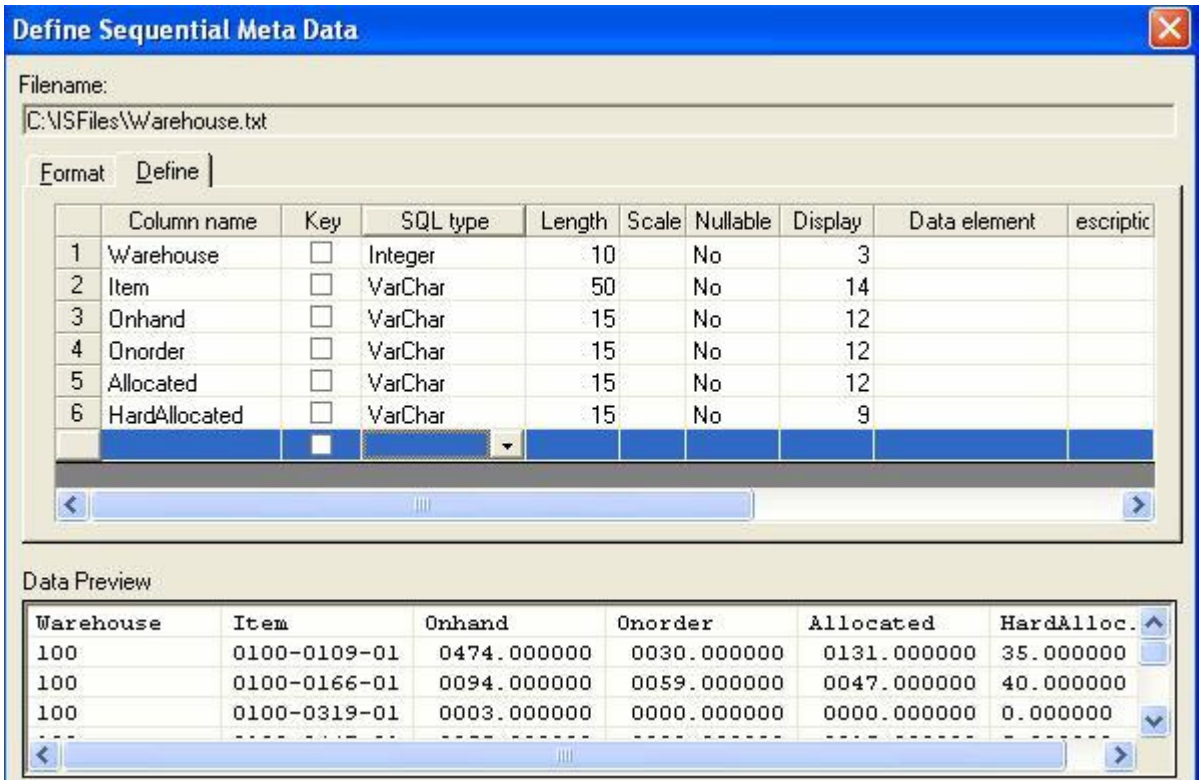
- Completed PPT training session for Lession 4

Task: Lookup Address

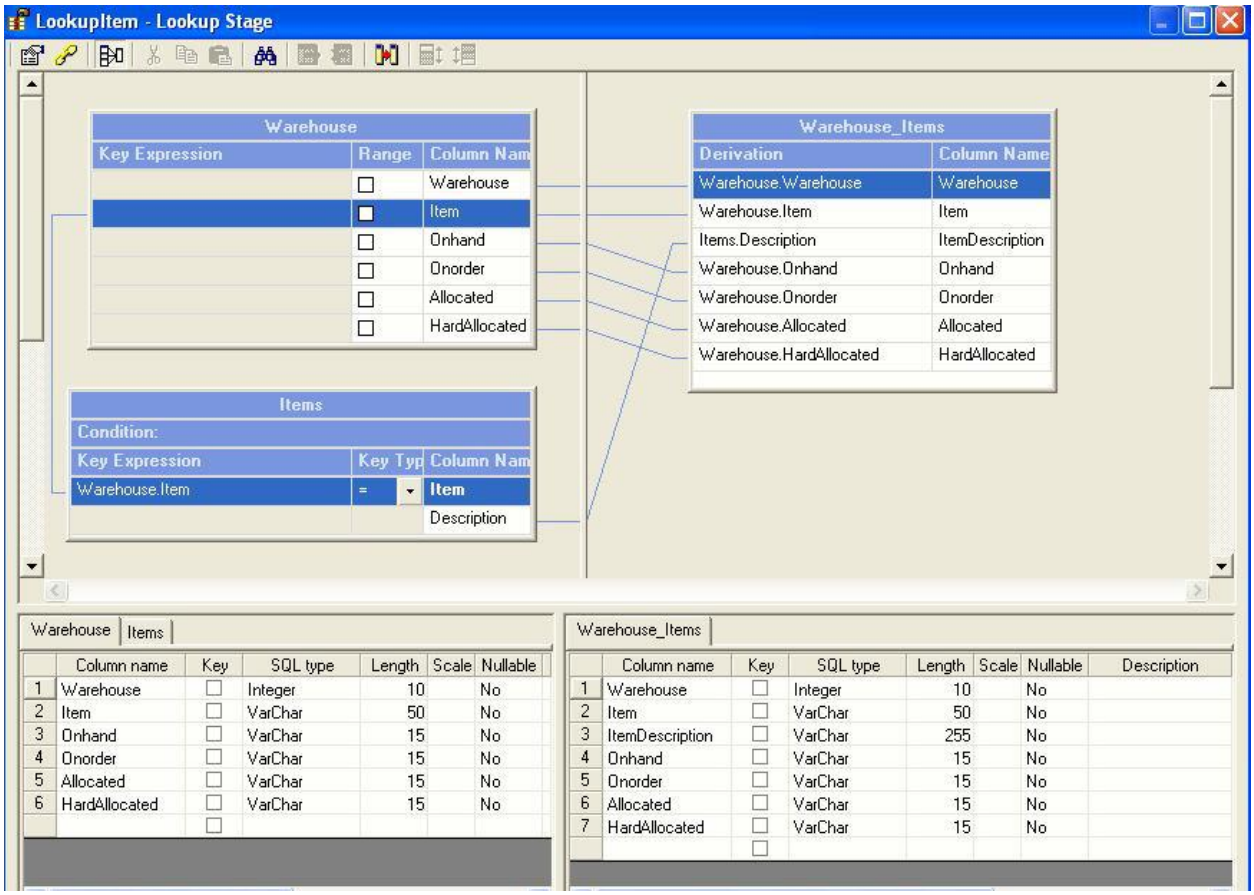
1. Open a new Parallel job and save it under the name LookupWarehouseItem. Add the stages and links and name them as shown.



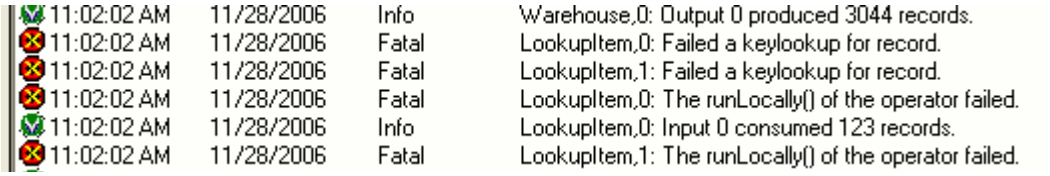
2. Import the Table Definition for the Warehouse.txt sequential file to your _Training>Metadata folder. Use the types shown below. (Otherwise, you will get errors when your job reads the file.)



3. In the Warehouse Sequential File stage extract data from the Warehouse.txt file. Be sure you can view the data.
4. In the Items Sequential File stage extract data from the Items.txt file. Import the Table Definition. The Item field should be defined like it is defined in the Warehouse stage, viz., as VarChar(50).
5. In the Items Sequential File stage, on the Format tab, change the Quote character to the single quote (‘). This is because some of the data contains double quotes as part of the data.
6. Open the Lookup stage. Map the Item field in the top left pane to the lookup Item key field in the bottom left pane by dragging the one to the other.
7. Drag all the Warehouse columns to the target link.
8. Drag the Description field in the lower left pane to target link placing it just below the Item field. Change its name to ItemDescription.



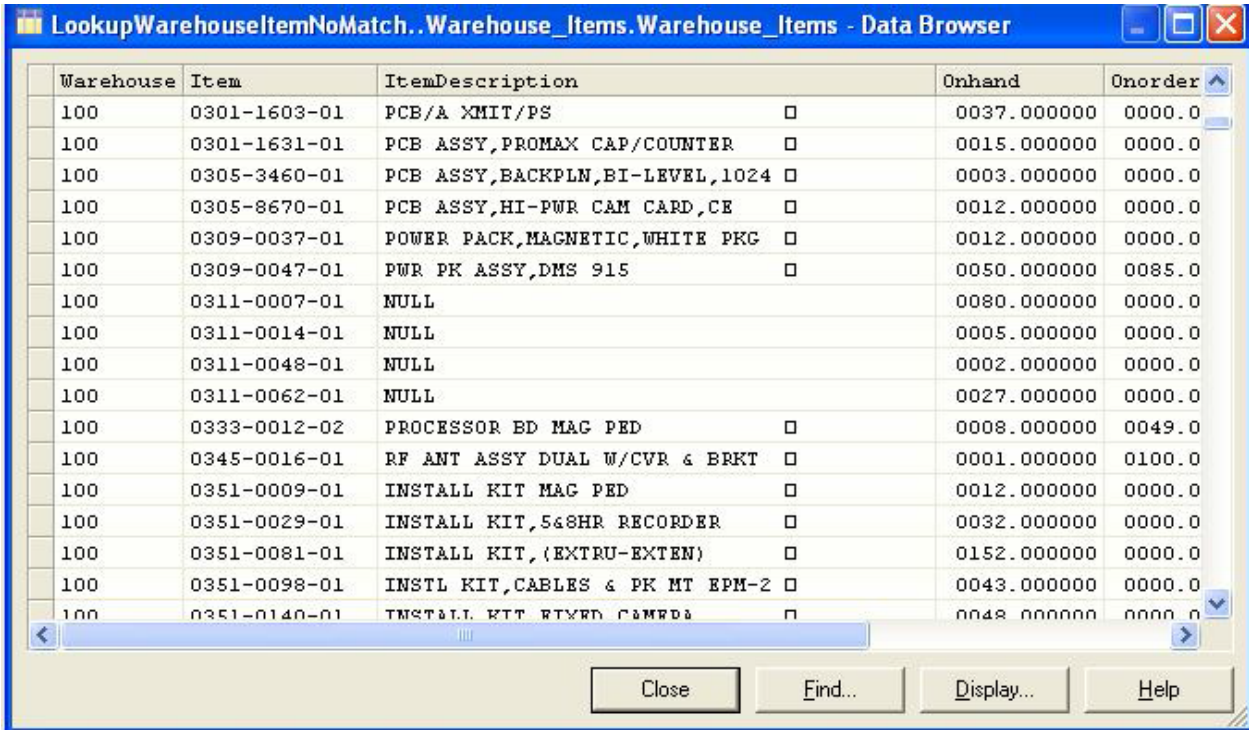
9. Edit your target Sequential stage as needed.
10. Compile and run. Examine the job log. Your job probably failed. Try to determine why it failed



Task: Handle lookup failures

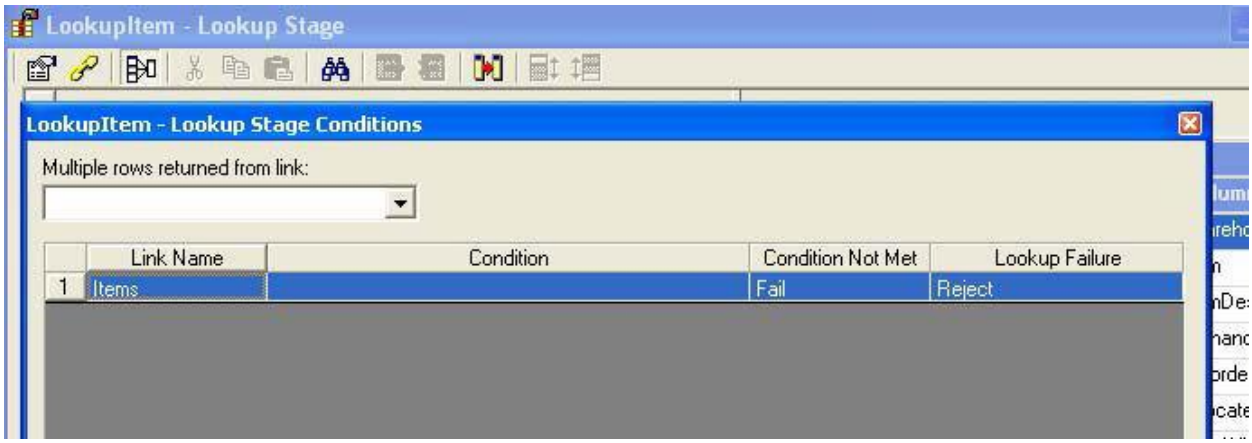
1. Save your job as LookupWarehouseItemHandleNoMatch.
2. Open up Lookup stage. Click the constraints icon (top, second from left). When the lookup fails, specify that the job is to continue.

3. Compile and run. Examine the log. You should not get any fatal errors this time.
4. View the data. Do you find any rows in the target file in which the lookup failed? (These would be rows with missing item descriptions.) What happened to these rows? This difficult question will be answered in the next task.
5. By default, when a there is a lookup failure with Continue, DataStage outputs NULLs to the lookup columns. Since these columns are not nullable, these rows get rejected.
6. Open up the Lookup stage. Make both the Description column on the left side and the ItemDescription column on the right side nullable.
7. Open up the target sequential stage. Replace NULLs by the string “NOMATCH”.
8. Compile and run.
9. View the data in the target Sequential File stage. Notice the NULL values.

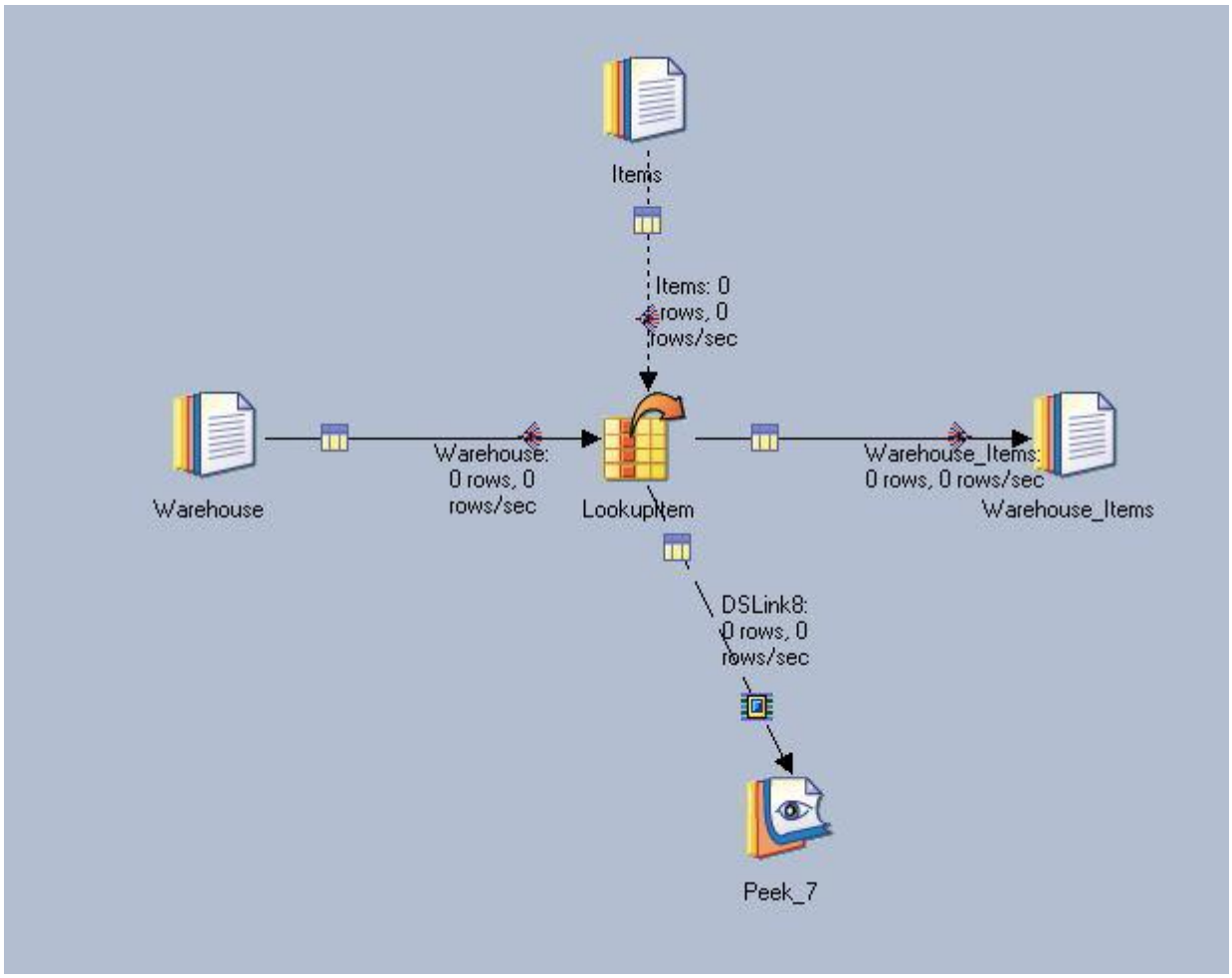


Task: Add a Reject link

1. Save your job as LookupWarehouseItemReject.
2. Open up Lookup stage and specify that Lookup Failures are to be rejected.



3. Add a Rejects link to a Peek stage to capture the lookup failures.

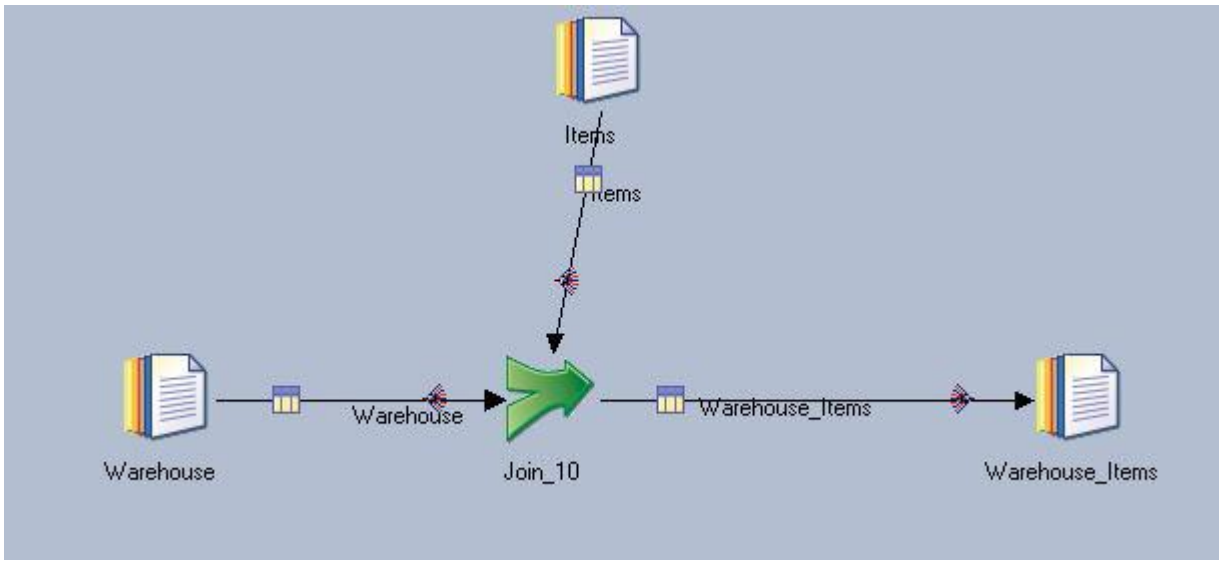


4. Compile and run. Examine the Peeks in the job log to see what rows were lookup failures.
5. Examine the job log. Notice in the Peek messages that a number of rows were rejected.

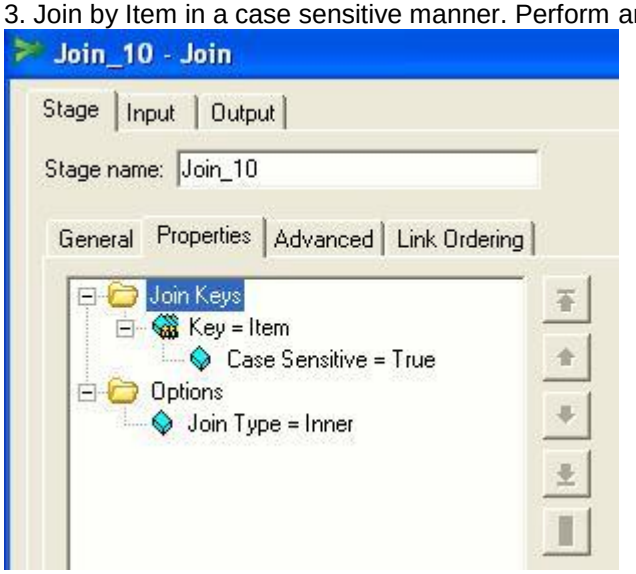
2:02:09 PM	1/15/2007	Info	Items,0: Progress: 90 percent.
2:02:09 PM	1/15/2007	Info	Warehouse,0: Import complete; 3044 records imported successfully, 0 rejected.
2:02:09 PM	1/15/2007	Info	Items,0: Import complete; 3281 records imported successfully, 0 rejected.
2:02:09 PM	1/15/2007	Info	Items,0: Output 0 produced 3281 records.
2:02:09 PM	1/15/2007	Info	Peek_7,0: Warehouse:100 Item:0311-0007-01 Onhand: 0080.000000 Onorder: 0000.000000 Allocat...
2:02:09 PM	1/15/2007	Info	Peek_7,1: Warehouse:100 Item:0311-0002-01 Onhand: 0010.000000 Onorder: 0000.000000 Allocat...
2:02:09 PM	1/15/2007	Info	Peek_7,0: Warehouse:100 Item:0311-0014-01 Onhand: 0005.000000 Onorder: 0000.000000 Allocat...
2:02:09 PM	1/15/2007	Info	Peek_7,1: Warehouse:100 Item:0311-0010-01 Onhand: 0085.000000 Onorder: 0000.000000 Allocat...
2:02:09 PM	1/15/2007	Info	Warehouse,0: Output 0 produced 3044 records.
2:02:09 PM	1/15/2007	Info	Peek_7,1: Warehouse:100 Item:0311-0056-01 Onhand: 0015.000000 Onorder: 0000.000000 Allocat...
2:02:09 PM	1/15/2007	Info	Peek_7,0: Warehouse:100 Item:0311-0062-01 Onhand: 0027.000000 Onorder: 0000.000000 Allocat...
2:02:09 PM	1/15/2007	Info	Warehouse_Items,0: Export complete; 3036 records exported successfully, 0 rejected.

Task: Use Join stage

1. Open your LookupWarehouseItemNoMatch job. Save it as LookupWarehouseItemJoin.
2. Replace the Lookup stage by the Join stage.

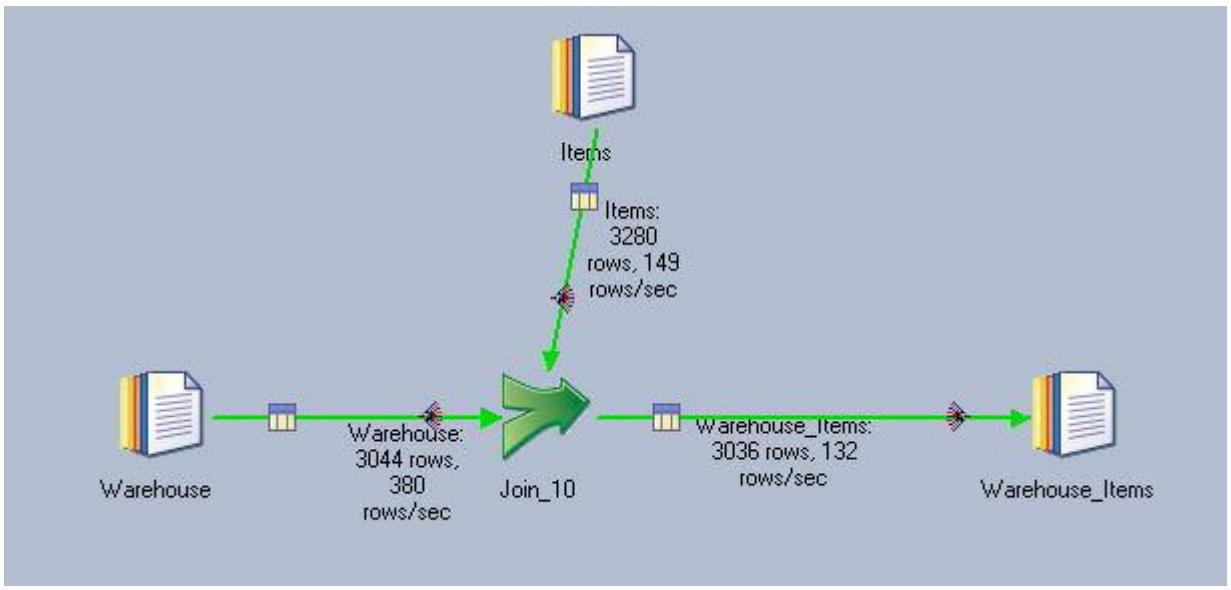


3. Join by Item in a case sensitive manner. Perform an inner join.



4. Click the Link Ordering tab. Make Items the Left link.

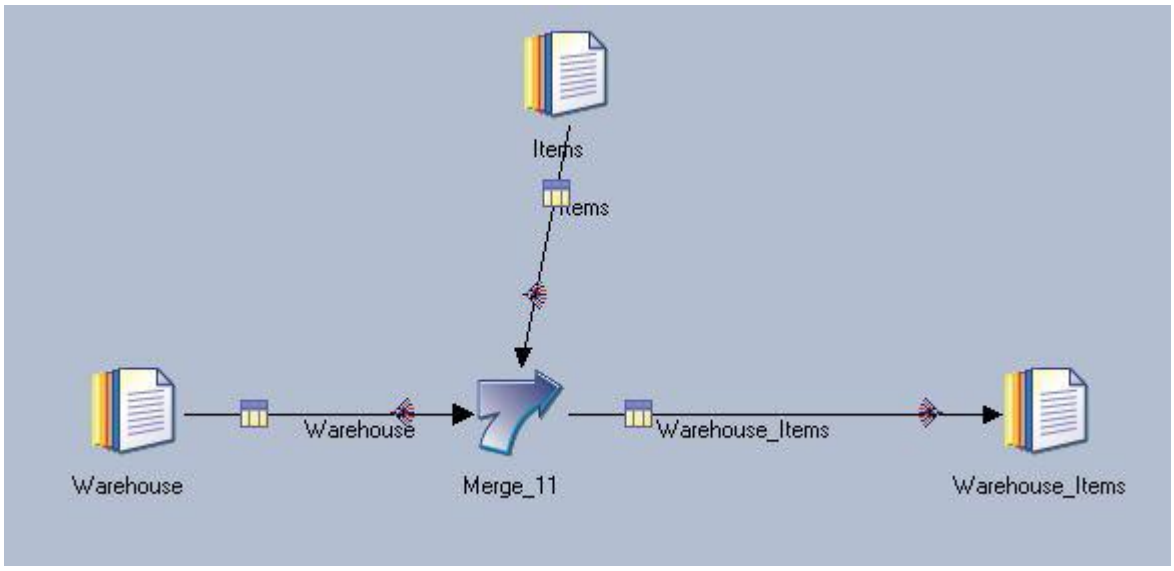
5. Check the Output>Mappings tab to make sure everything is mapped correctly.
6. Compile and run. Verify the results and verify that the number of records written to the target sequential file is the same for the Lookup job. You can verify this by examining the performance statistics or by seeing how many records are exported by the Warehouse_Items stage in the job log



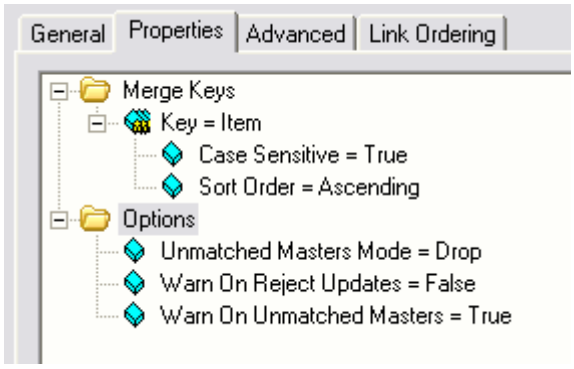
.
7. Recompile and run your job after selecting Right Outer Join as the join type. View the data. You should see NULLs in the eight no-matches.

Task: Use Merge stage

1. In this task, we will see if the Merge stage can be used in place of the Join stage. We will see that it cannot be successfully used.
2. Save your job as LookupWarehouseItemMerge. Replace the Join stage by the Merge stage.

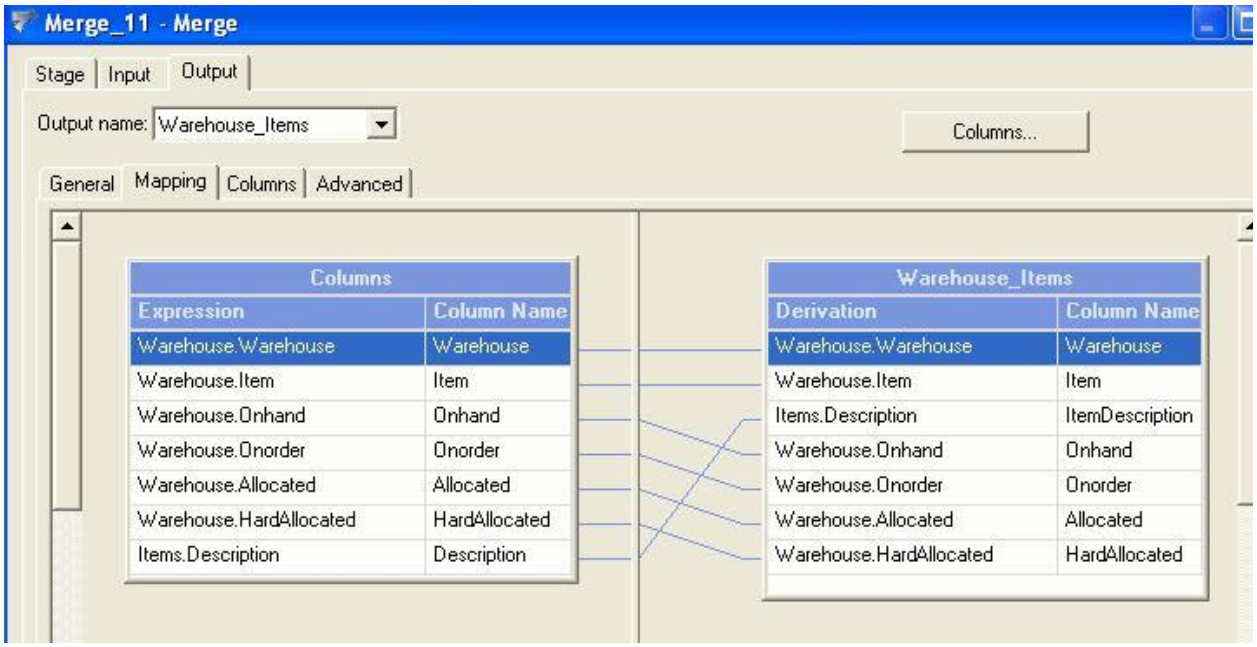


3. In the Merge stage, specify that data is to merged, with case sensitivity, by the key (Item). Assume that the data is sorted in ascending order. Also specify that unmatched records from Warehouse (the Master) are to be dropped.

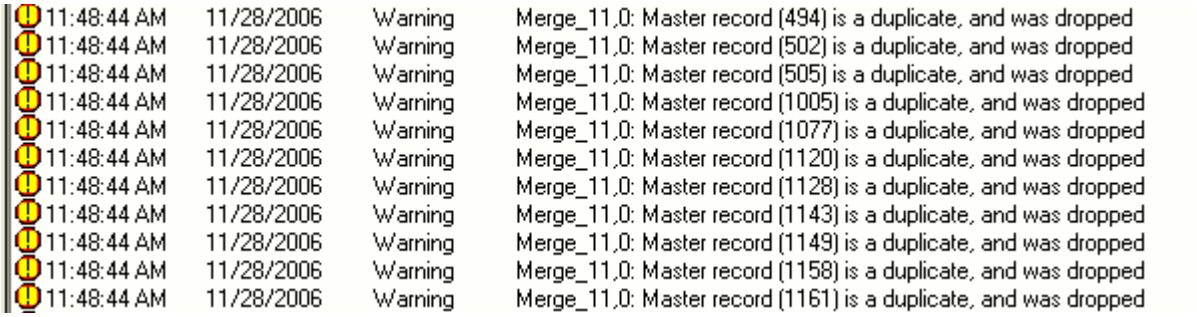


4. Insure that the Warehouse_Items link is the Master link.

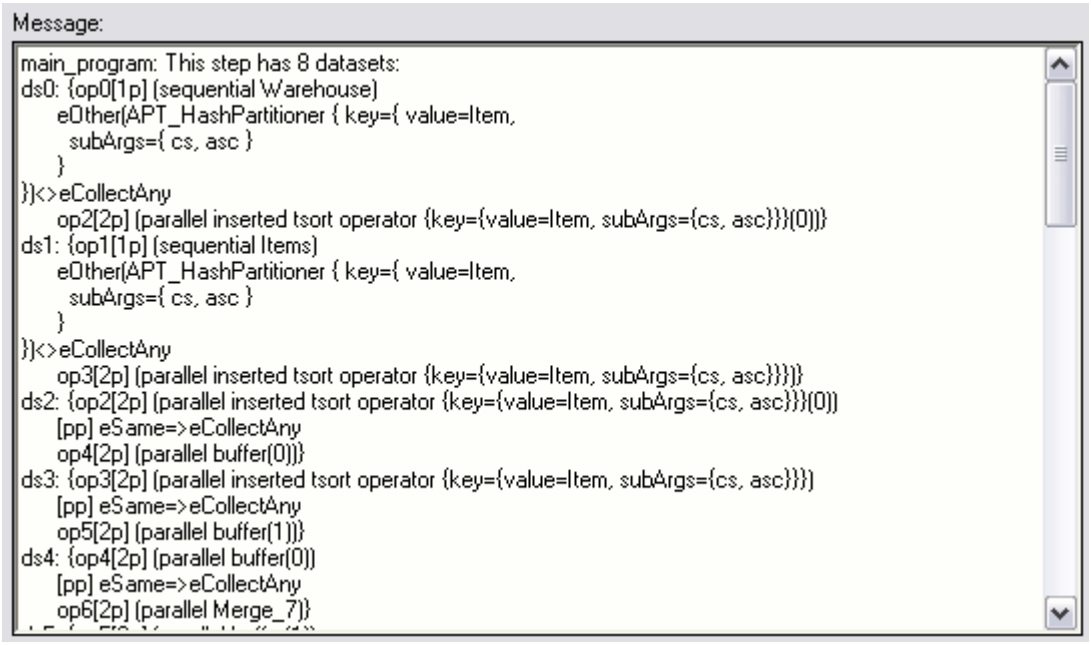
5. Map input columns to the appropriate output columns.



6. Compile and run. View the data.
7. Examine the performance statistics and/or job log to determine how many records were written to the target sequential file.
8. Notice that over 80 fewer rows are written to the target sequential file. Why? Examine the job log. Notice that a bunch of records from the Warehouse.txt file have been dropped because they have duplicate key values.



9. The moral here is that you cannot use the Merge stage if your Master source has duplicates. None of the duplicate records will match with update records.
10. Recall that another requirement of the Merge stage (and Join stage) is that the data is hashed and sorted by the key. We did not do this explicitly, so why didn't our job fail? Let's examine the job log for clues. Open up the Score message. Notice that sorts (tsort operators to be precise) have been added by DataStage.

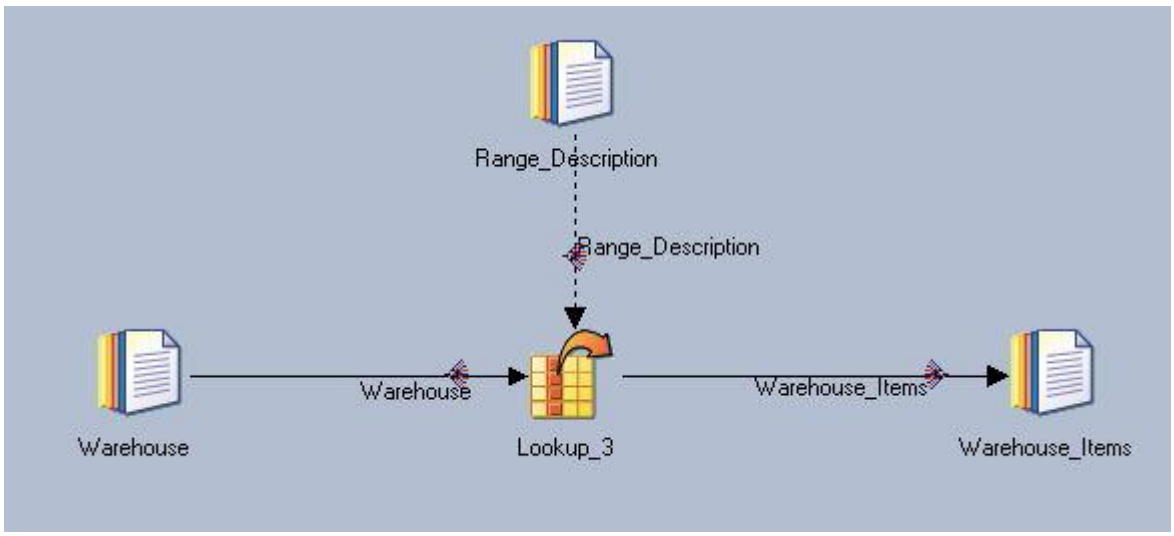


Range lookups

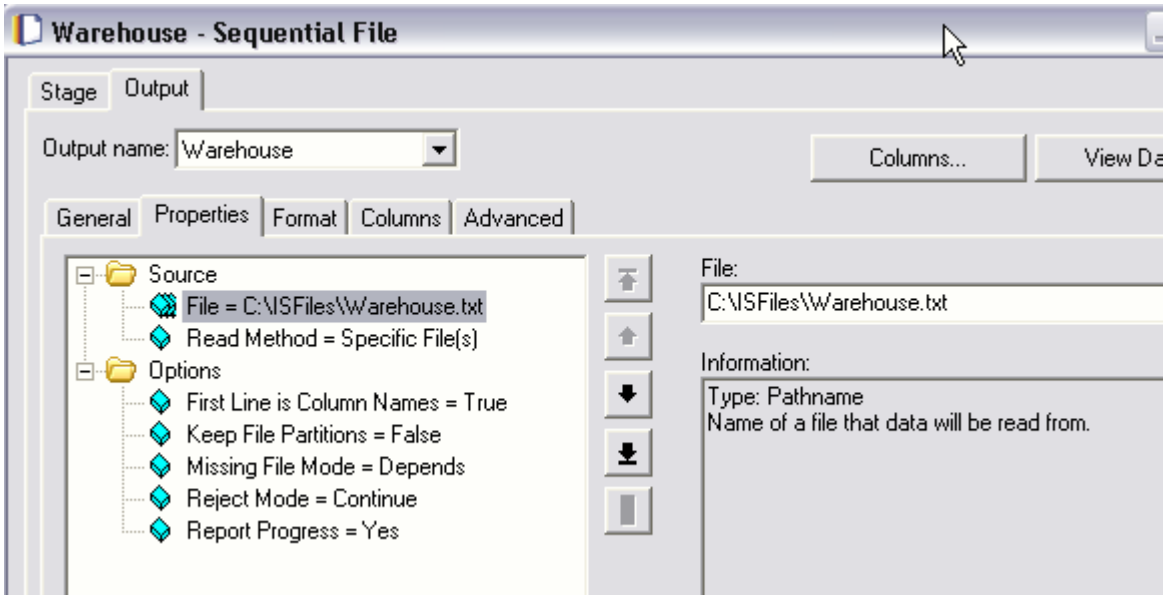
Task: Design a job with a reference link range lookup

This job reads Warehouse item records from the source file. The lookup file contains start and end item numbers with descriptions that apply to items within the specified range. The appropriate description is added to each record which is then written out to a sequential file.

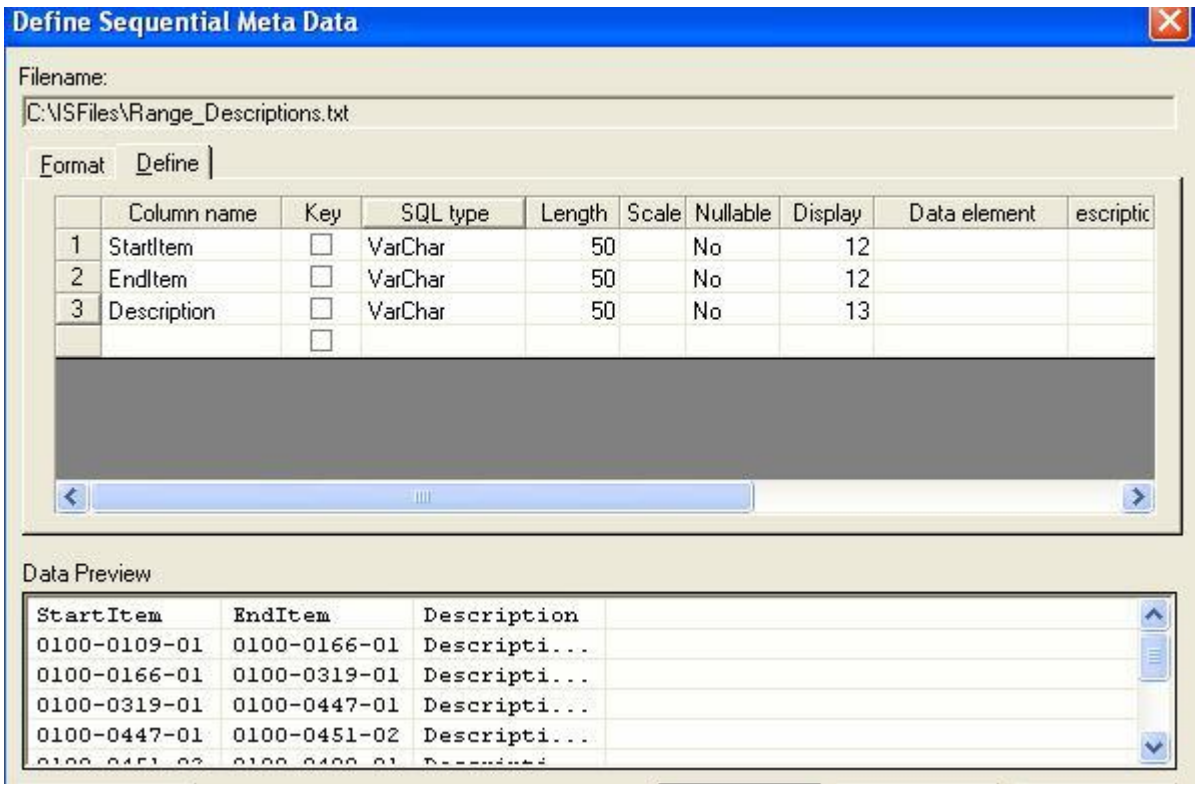
1. Open a new Parallel job and save it under the name LookupItemsRangeRef. Save in the _Training>Jobs folder. Add the stages and links and name them as shown.



2. Edit the Warehouse sequential stage to read from the Warehouse.txt file. Verify that you can view the data.

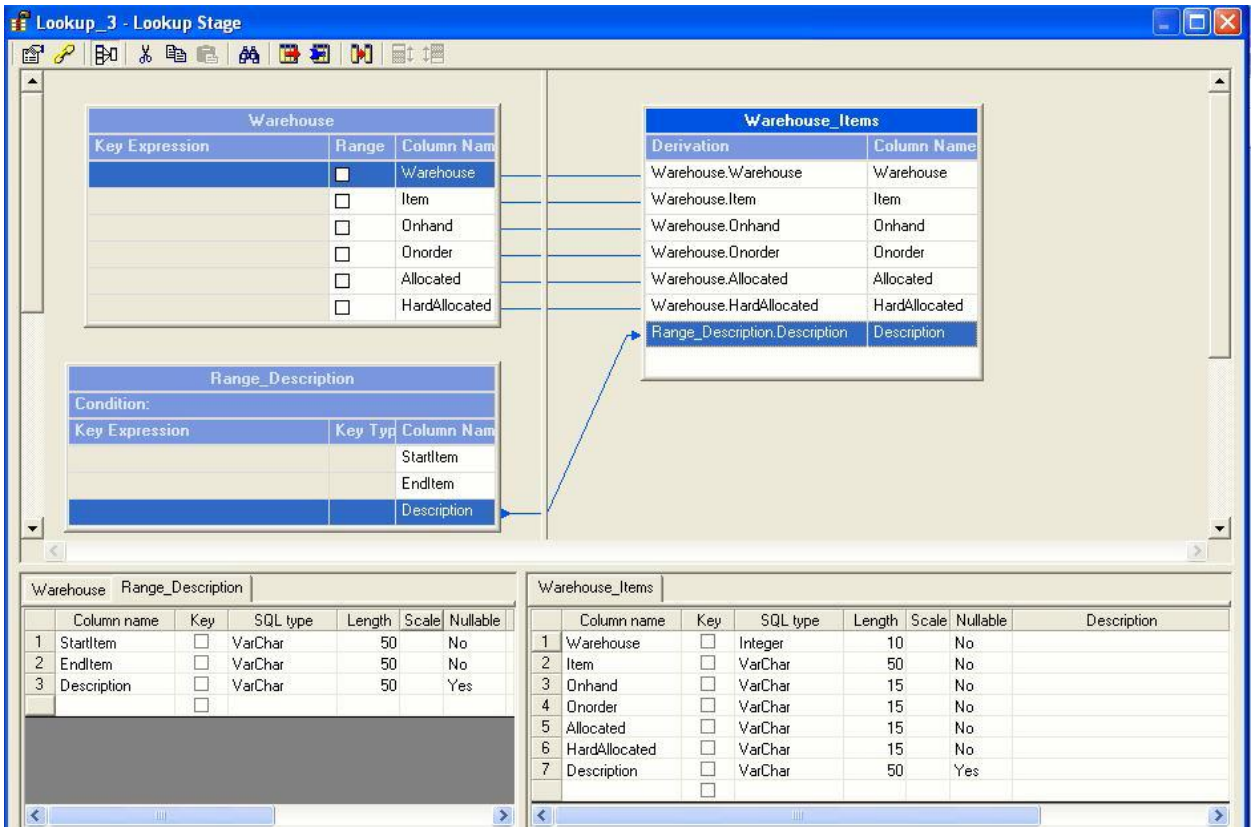


3. Import the Table Definition for the Range_Descriptions.txt sequential file. The StartItem and EndItem fields should be defined like the Item field is defined in the Warehouse stage, viz., as VarChar(50).

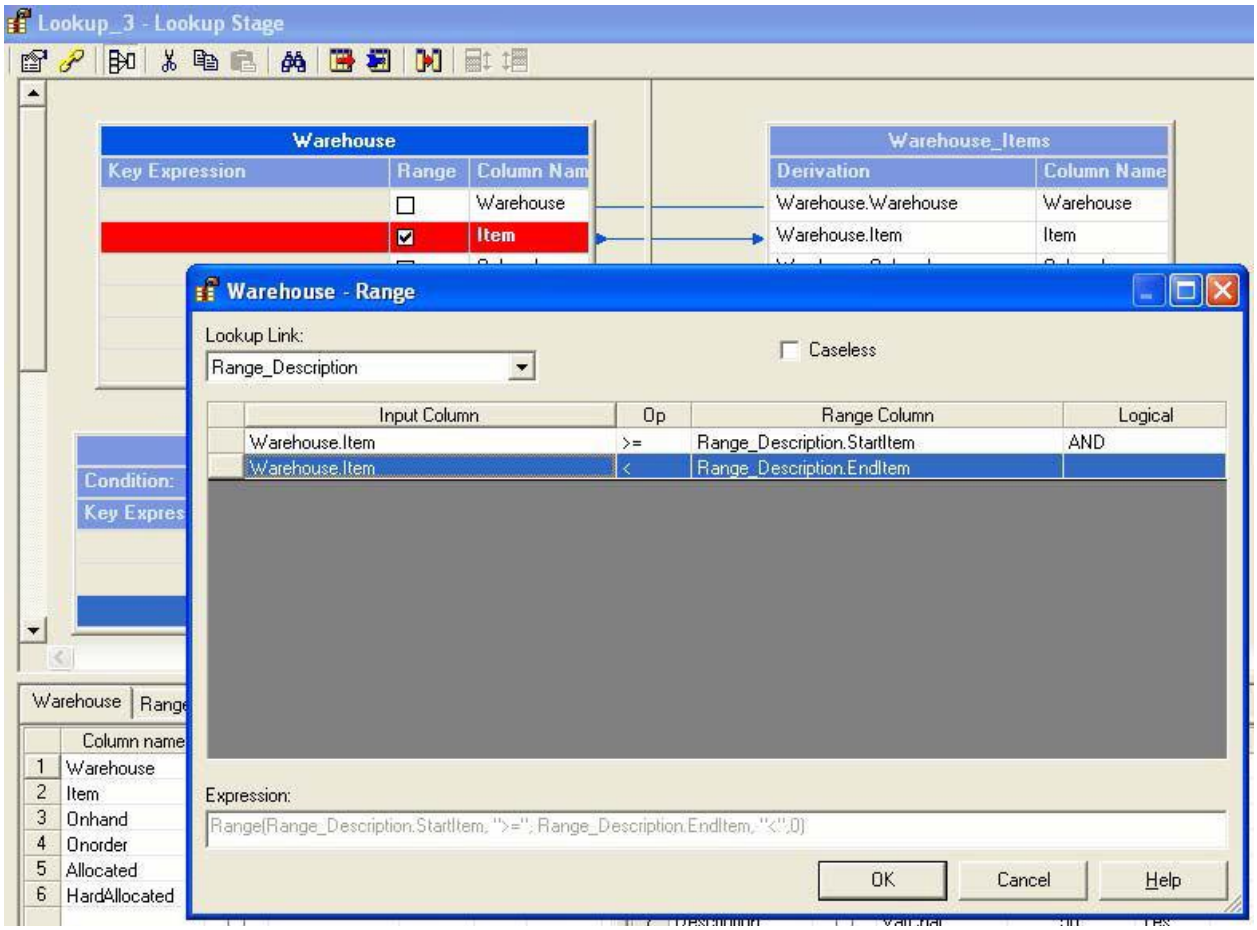


4. Edit the Range_Description sequential stage to read from the Range_Descriptions.txt file. Verify that you can view the data.
5. Open the Lookup stage. Drag all the Warehouse columns across. **Important!!!** Set

the Description column to nullable. Then drag the Description column from the Range_Description link across. (The Description column on the left and the Description column on the right should both be nullable.)



6. Select the Range checkbox to the left of the Item field in the Warehouse table.
7. Double-click on the Key Expression cell for the Item column to open the Range Expression editor. Specify that the Warehouse.Item column value is to be greater than or equal to the StartItem column value and less than the EndItem column value.

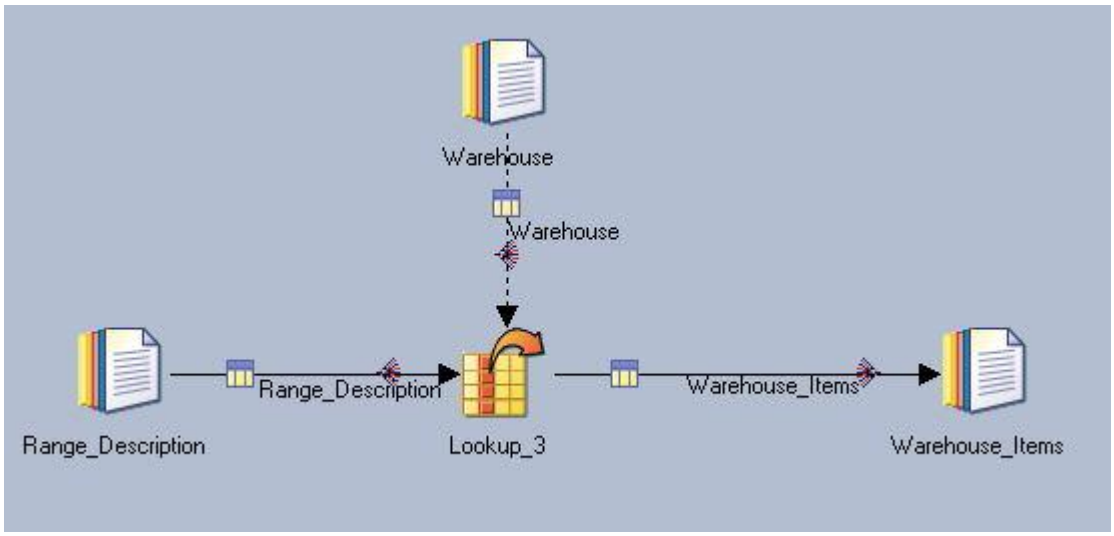


8. Open the constraints window and specify that the job is to continue if a lookup failure occurs.
9. Edit the target Sequential stage. Write to a file named WarehouseItems.txt. Create the file with column headings.
10. The Description column in the Sequential file stage is nullable. Go to the extended properties window for this column. Replace NULL values by NO_DESCRIPTION.
11. Compile and run your job.
12. Open the target file in a text editor to view and verify the data.

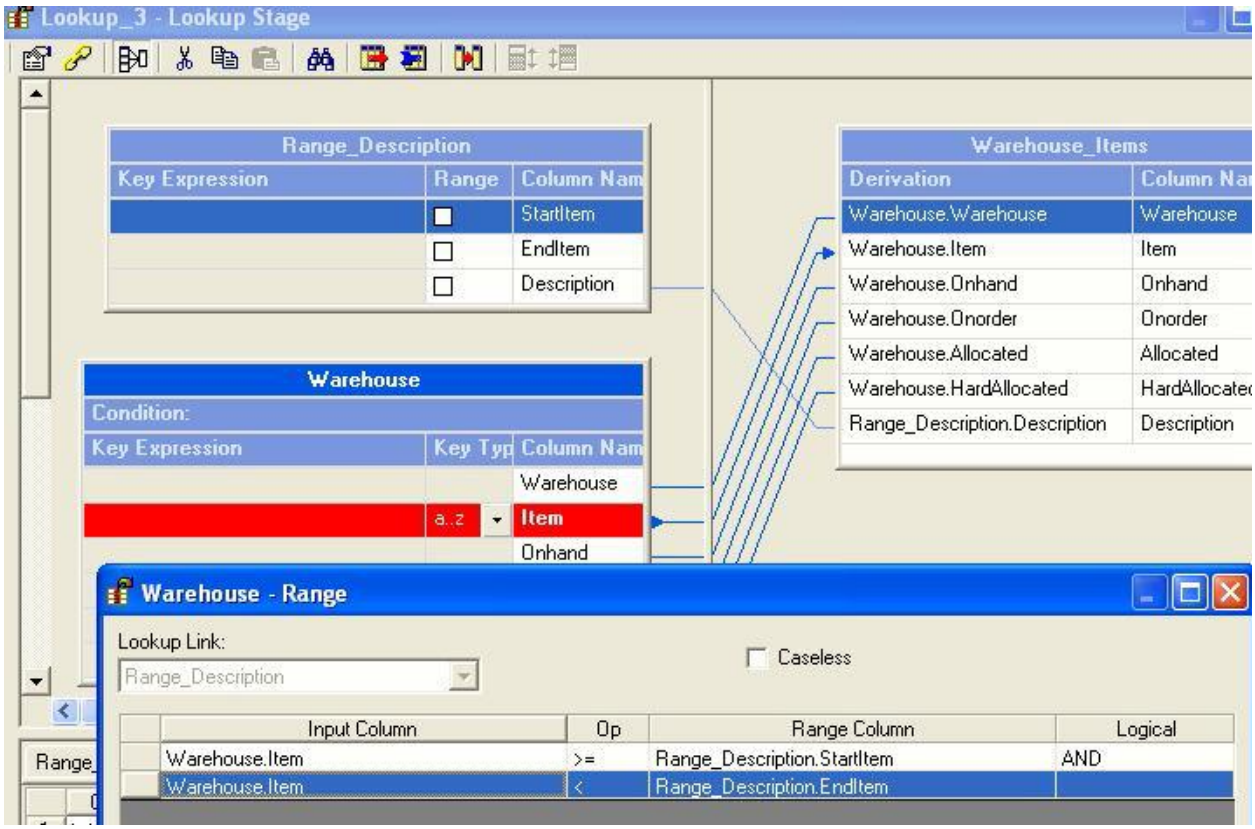
Task: Design a job with a stream range lookup

This job reads from the RangeDescription.txt file. It then does a lookup into the Warehouse.txt file. For each row read, it selects all the records from the Warehouse.txt file with items within the range. The appropriate description is added to each record which is then written out to the DB2 table.

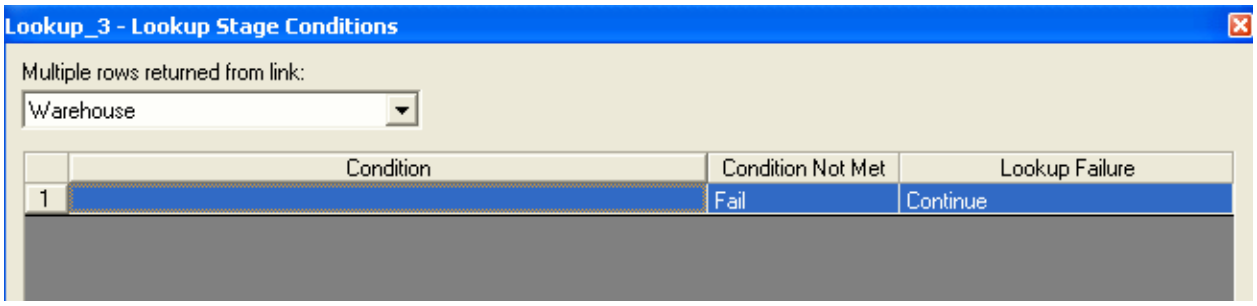
1. Save your job as LookupItemsRangeStream in your _Training>Jobs folder.
2. Reverse the source and lookup links. First make the source link a reference link. Click the right mouse button and click Convert to reference. Then make the lookup link a stream link.



3. Open up your Lookup stage. Select Item in the Warehouse link as the key. Specify the Key type as Range.
4. Double-click on the Key Expression cell. Specify the range expression.



5. Click the Constraints icon. Specify that multiple rows are to be returned from the Warehouse link. Also specify that the job is to continue if there is a lookup failure.

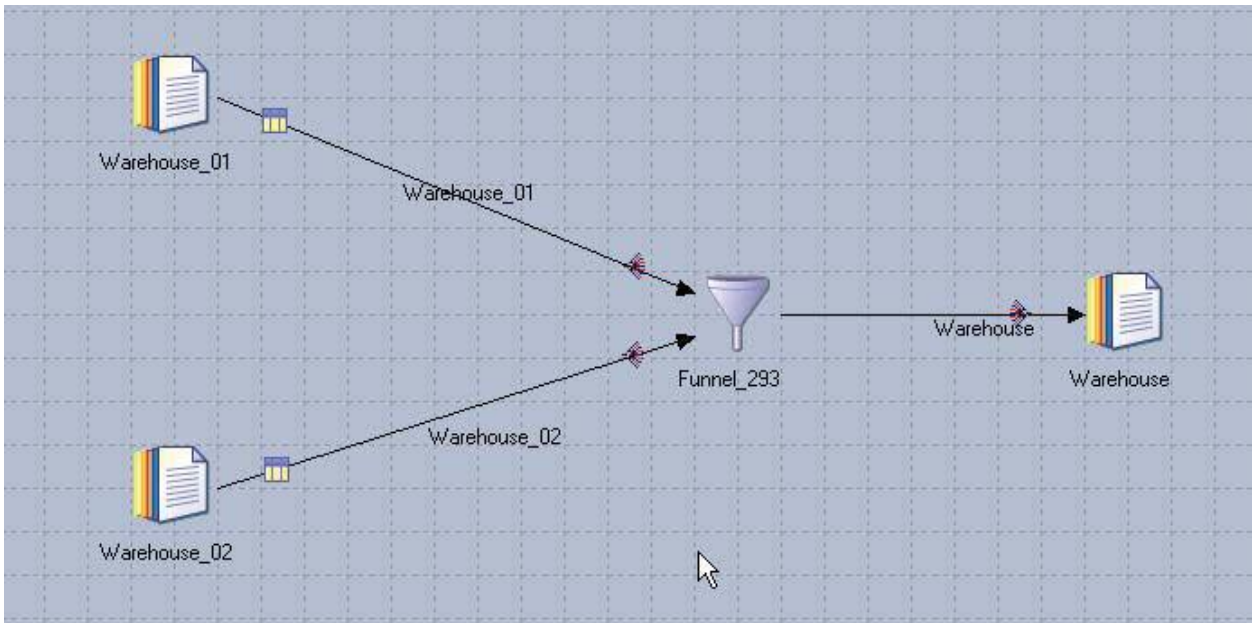


6. Compile
7. View the data.

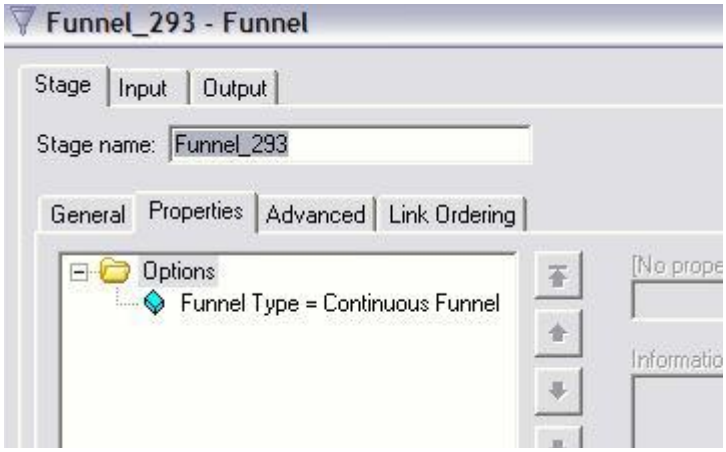
LookupItemsRangeStream..Warehouse_Items.Warehouse_Items - Data Browser

Warehouse	Item	Onhand	Onorder	Allocated	HardAllocated	Description
100	0100-0109-01	0474.000000	0030.000000	0131.000000	35.000000	Description A
100	0100-0166-01	0094.000000	0059.000000	0047.000000	40.000000	Description B
100	0100-0319-01	0003.000000	0000.000000	0000.000000	0.000000	Description C
100	0100-0447-01	0055.000000	0000.000000	0015.000000	5.000000	Description D
100	0100-0451-02	0024.000000	0000.000000	0000.000000	0.000000	Description E
100	0100-0490-01	0004.000000	0000.000000	0006.000000	4.000000	Description F
109	0100-0490-01	0002.000000	0000.000000	0019.000000	2.000000	Description F
100	0100-0535-01	0003.000000	0000.000000	0000.000000	0.000000	Description G
100	0100-0595-01	0418.000000	0227.000000	0074.000000	73.000000	Description H
100	0100-0739-01	0004.000000	0000.000000	0000.000000	0.000000	Description I
100	0100-0743-01	0007.000000	0000.000000	0000.000000	0.000000	Description J
100	0100-0743-02	0002.000000	0000.000000	0000.000000	0.000000	Description J
100	0100-0749-06	0003.000000	0000.000000	0000.000000	0.000000	Description J
100	0100-0770-03	0005.000000	0000.000000	0001.000000	1.000000	Description J

Combining data using the Funnel Stage
Task: Build a job with a Funnel stage
In this task, you'll combine data from two of the Warehouse.txt files into a single file.
1. Open a new Parallel job and save it under the name FunnelWarehouse. Add links and stages and name them as shown.

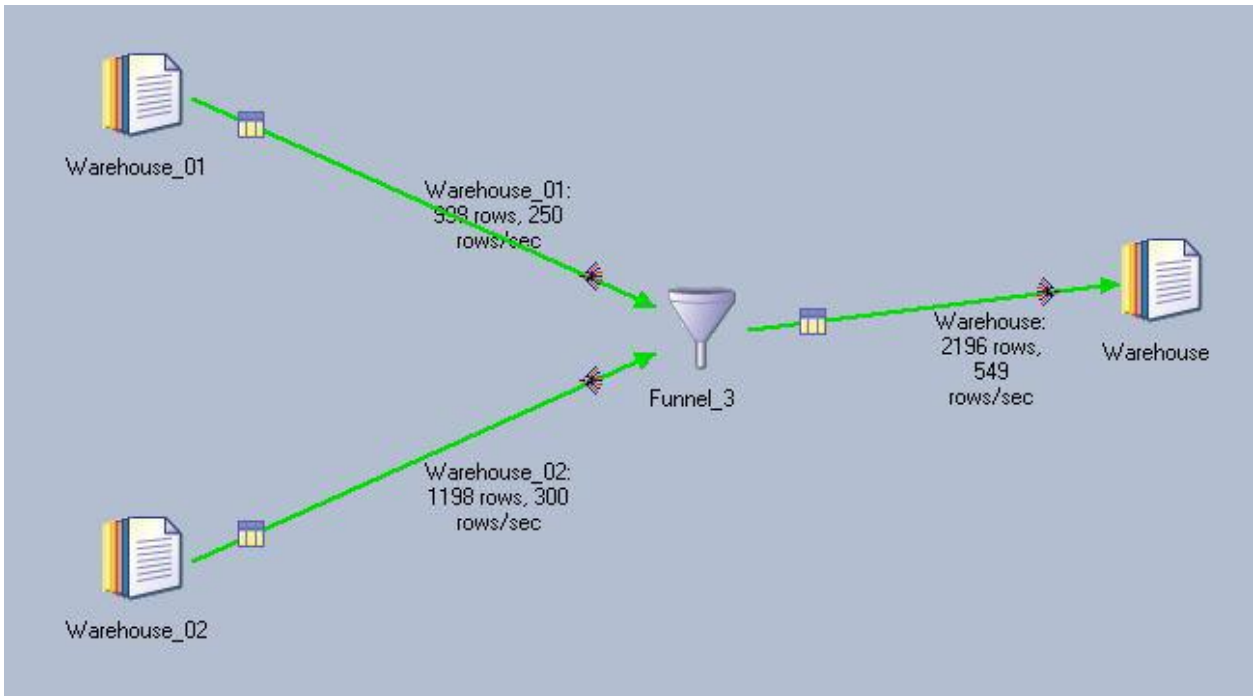


2. Edit the two source Sequential stages to extract data from the twoWarehouse files, Warehouse_031005_01.txt and Warehouse_031005_02.txt. They have the same metadata as the Warehouse.txt file.
3. Edit the Funnel stage to combine data from the two files in Continuous mode.



4. On the Output>Mapping tab map all columns through the stage.
5. Write to a file named Warehouse_031005.txt.

6. Compile and run. Verify that the number of rows going into the target is the sum of the number of rows coming from the source. And verify the data.



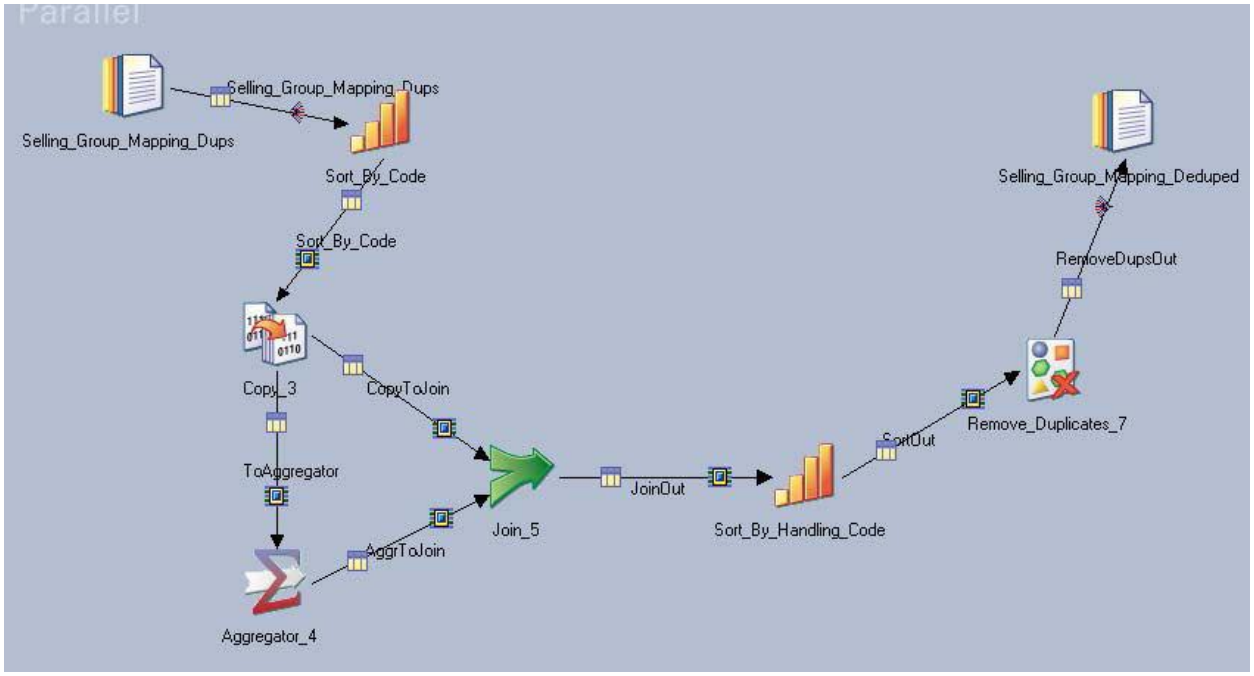
Lab 05: Sorting and Aggregating Data

Assumptions:

- Completed PPT training session for Lession 4

Task: Create the job design

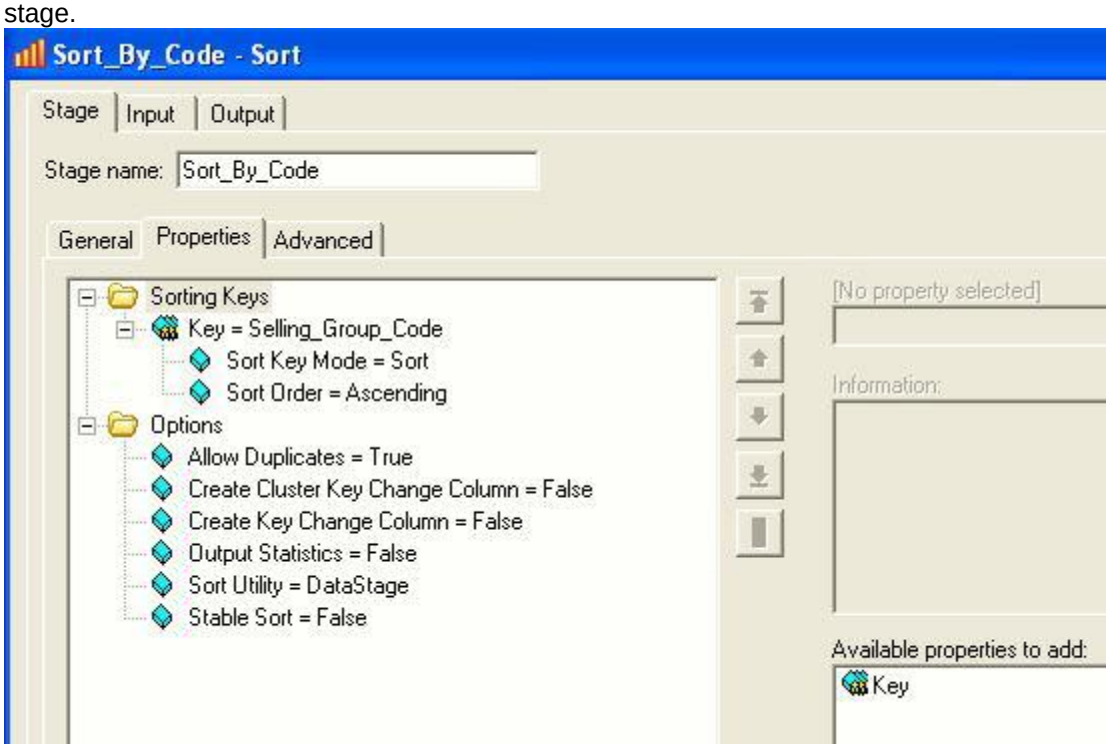
1. Open a new parallel job and save is as ForkJoin. Add stages and links and name them as shown.



2. Edit the Selling_Group_Mapping_Dups Sequential stage to read from the Selling_Group_Mapping_Dups.txt file. This file has the same format as the Selling_Group_Mapping.txt file.

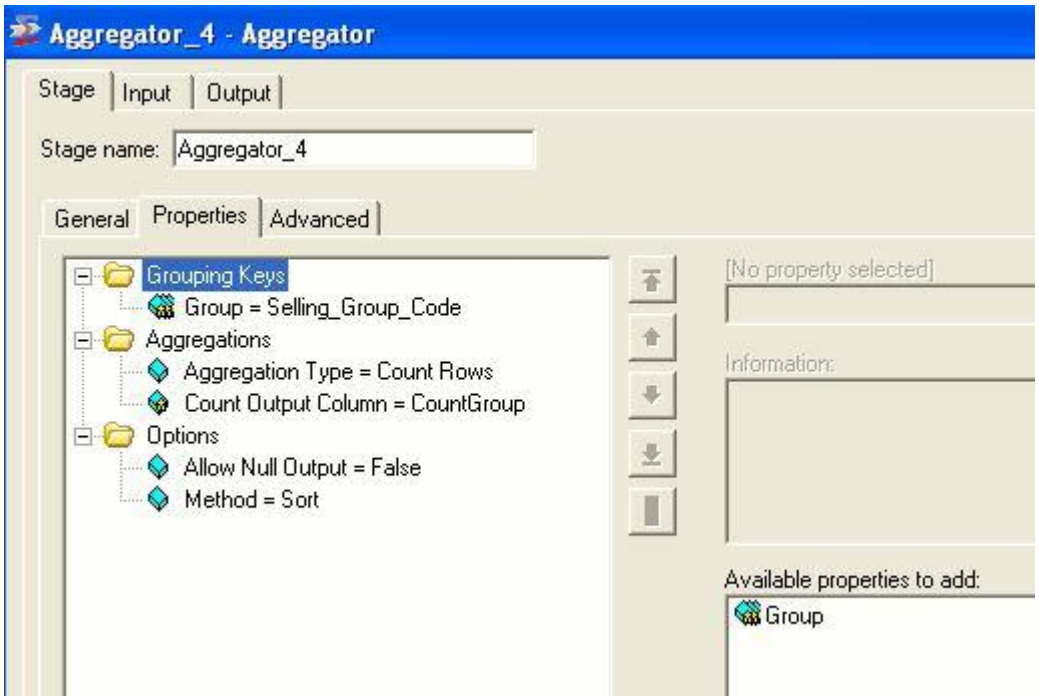
3. Edit the Sort_By_Code Sort stage. Perform an ascending sort by Selling_Group_Code.

The sort should not be a stable sort. Send all columns through the stage.

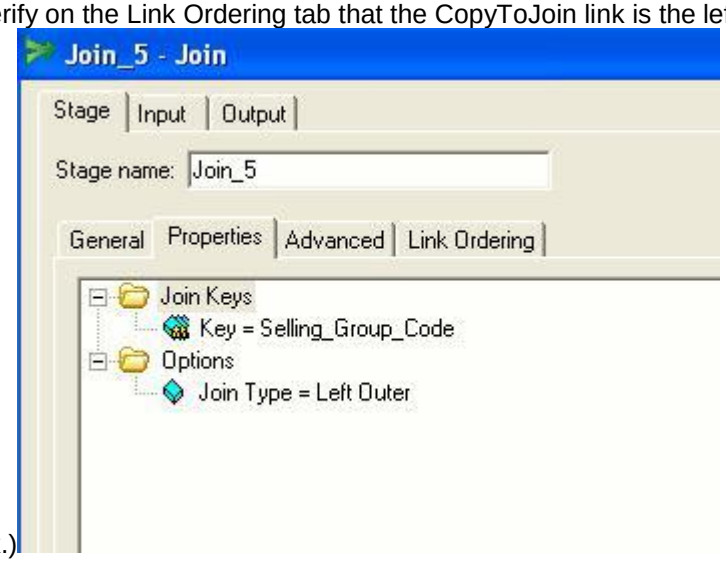


4. In the Copy stage, specify that all columns move through the stage to the output link going to the Join stage.
5. Specify that only the Selling_Group_Code column moves through the Copy stage to the Aggregator stage.
6. Edit the Aggregator stage. Specify that records are to be grouped by Selling_Group_Code.
7. Specify that the type of aggregation is to count the rows.
8. Specify that the aggregation amount is to go into a column named CountGroup. Define this column on the Outputs>Columns tab as an integer, length 10.

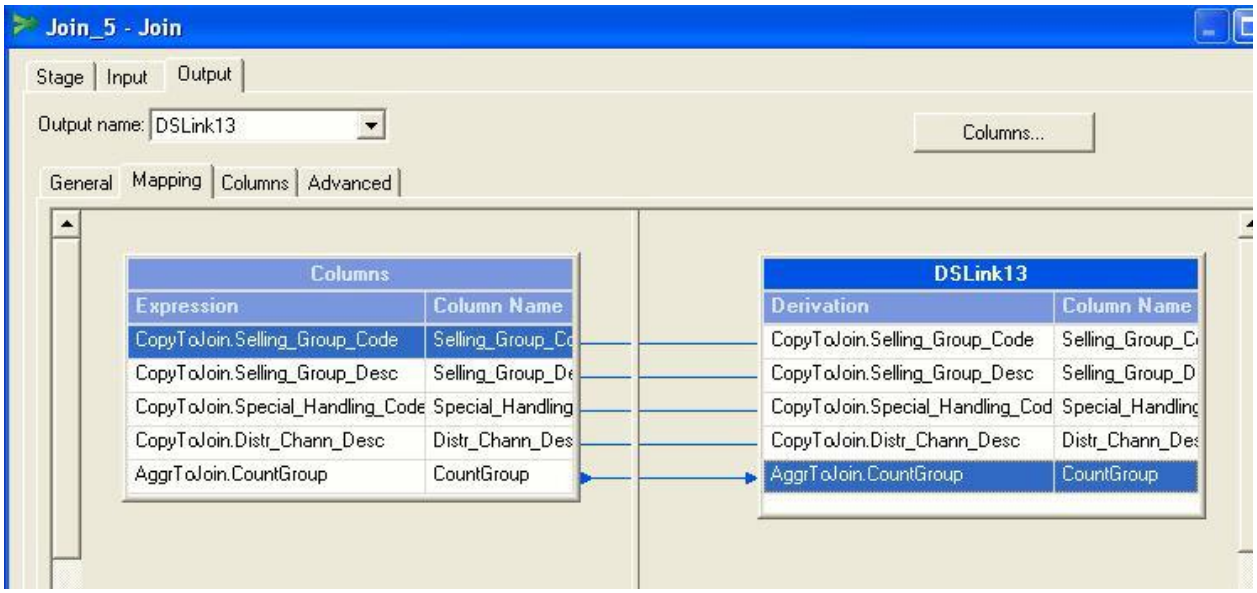
9. Select Sort as the aggregation method, because the data has been sorted by the grouping



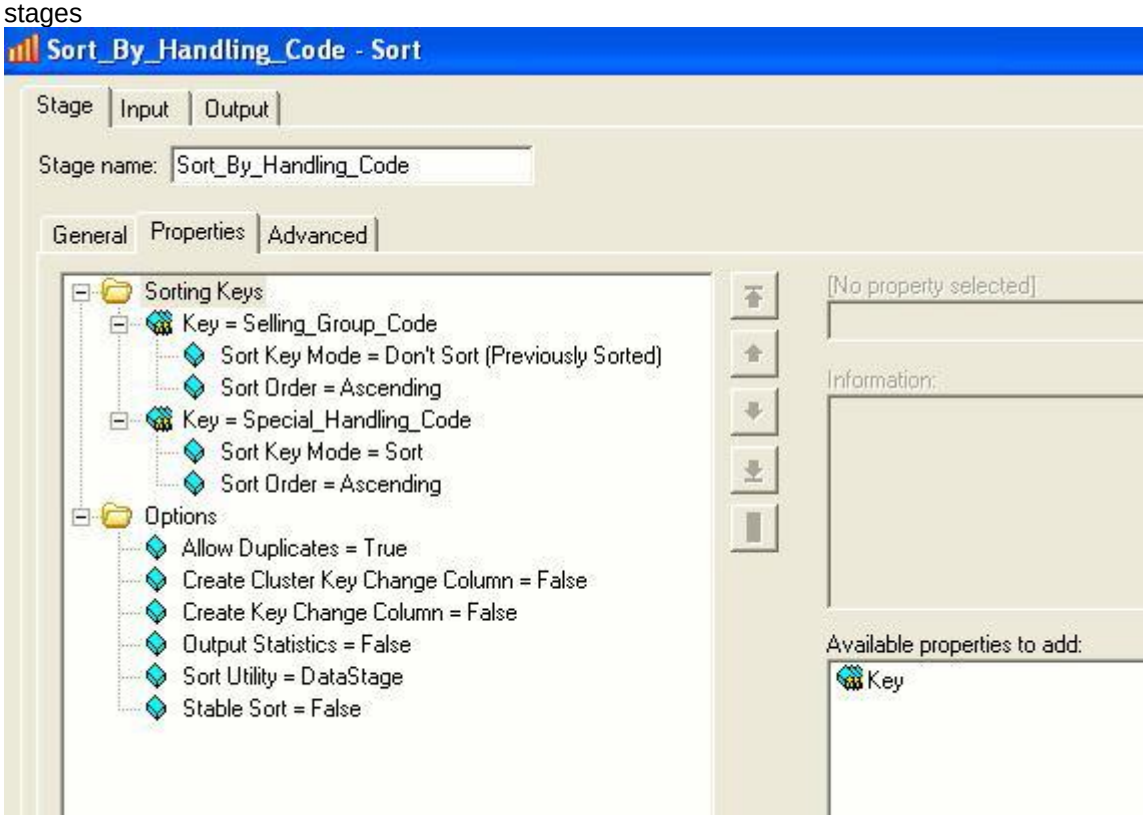
10. Edit the Join stage. The join key is Selling_Group_Code. The Join Type is Left Outer. (Verify on the Link Ordering tab that the CopyToJoin link is the left link.)



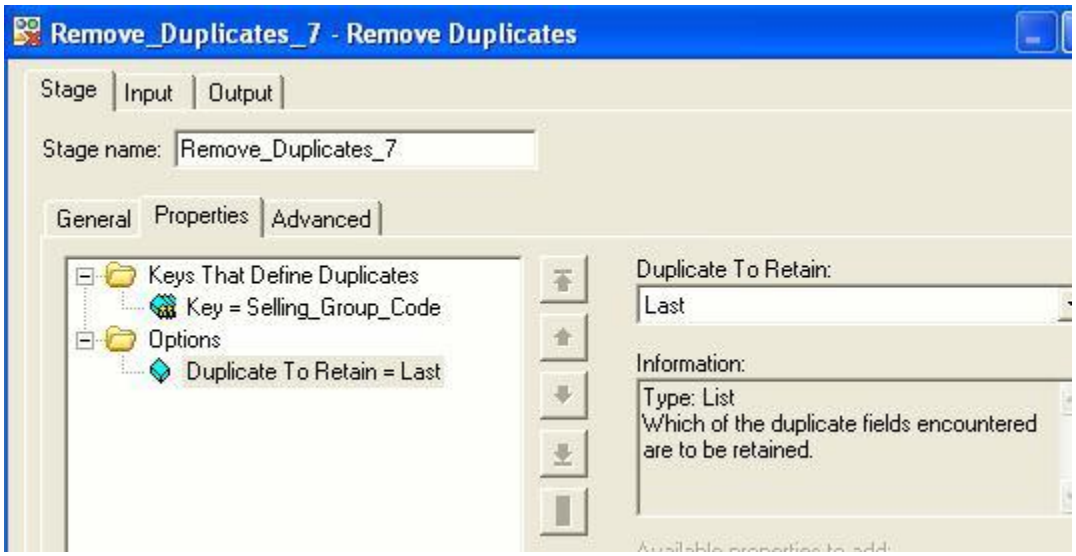
11. On the Outputs>Mapping tab, map all columns across.



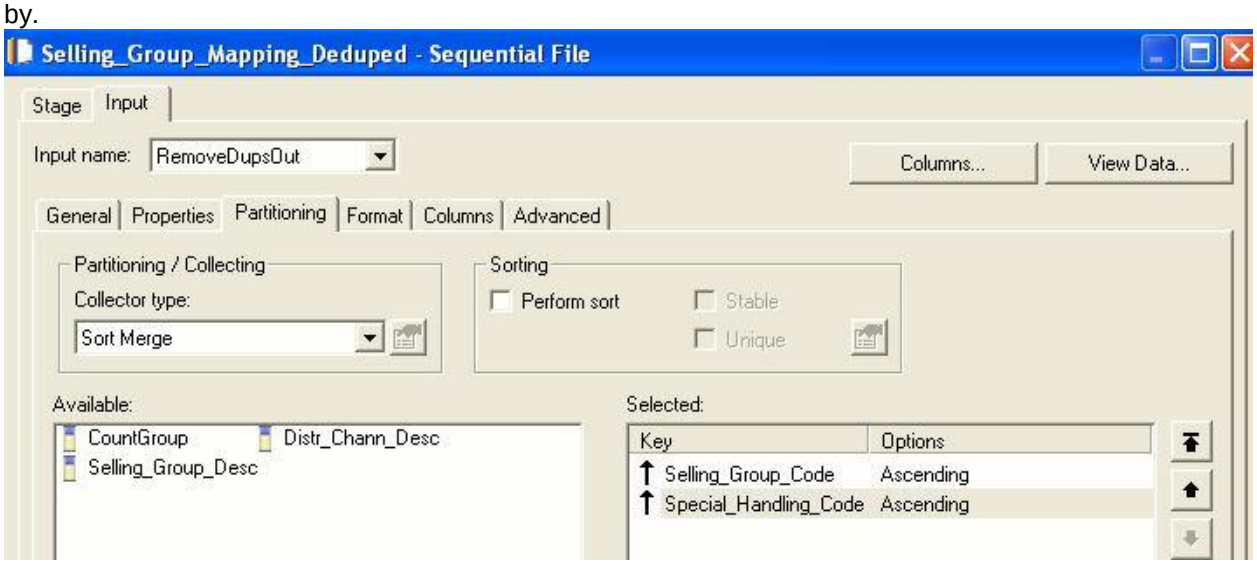
12. Edit the Sort_By_Handling_Code stage. Within each Selling_Group_Code sorted group sort by Special_Handling_Code. Specify that the data has already been sorted by Selling_Group_Code, so that the stage doesn't repeat this sort. Turn off Stable Sort. Move all columns through the
stages



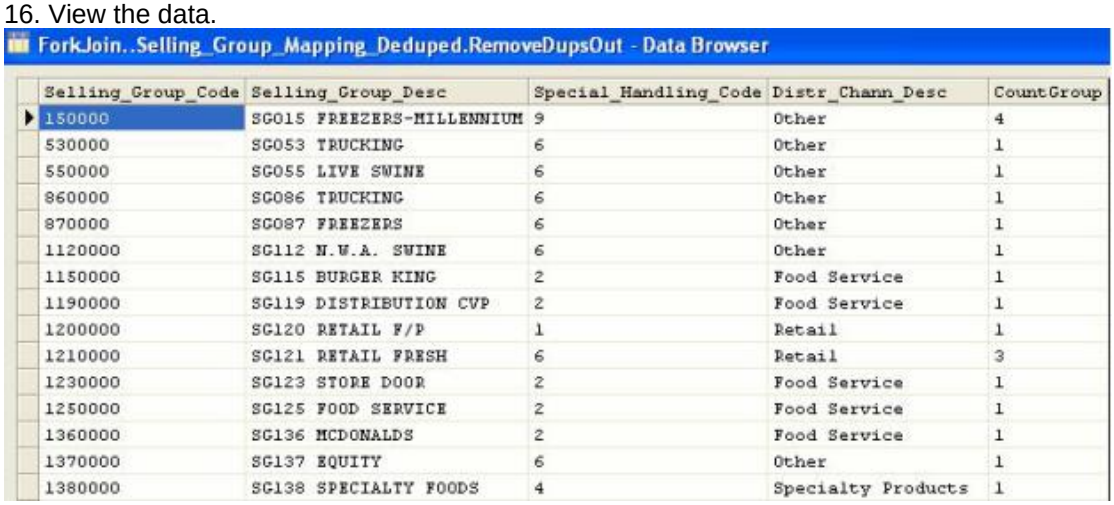
13. Edit the Remove Duplicates stage. Group by Selling_Group_Code. Retain the last record in each group.



14. Edit the target Sequential stage. Write to a file named Selling_Group_Code_Deduped.txt. On the Partitioning tab, collect the data using Sort Merge based on the two columns the data has been sorted
by.



16. View the data.



Selling_Group_Code	Selling_Group_Desc	Special_Handling_Code	Distr_Chann_Desc	CountGroup
150000	SG015 FREEZERS-MILLENNIUM	9	Other	4
530000	SG053 TRUCKING	6	Other	1
550000	SG055 LIVE SWINE	6	Other	1
860000	SG086 TRUCKING	6	Other	1
870000	SG087 FREEZERS	6	Other	1
1120000	SG112 N.W.A. SWINE	6	Other	1
1150000	SG115 BURGER KING	2	Food Service	1
1190000	SG119 DISTRIBUTION CVP	2	Food Service	1
1200000	SG120 RETAIL F/P	1	Retail	1
1210000	SG121 RETAIL FRESH	6	Retail	3
1230000	SG123 STORE DOOR	2	Food Service	1
1250000	SG125 FOOD SERVICE	2	Food Service	1
1360000	SG136 MCDONALDS	2	Food Service	1
1370000	SG137 EQUITY	6	Other	1
1380000	SG138 SPECIALTY FOODS	4	Specialty Products	1

Lab 06: Transforming Data

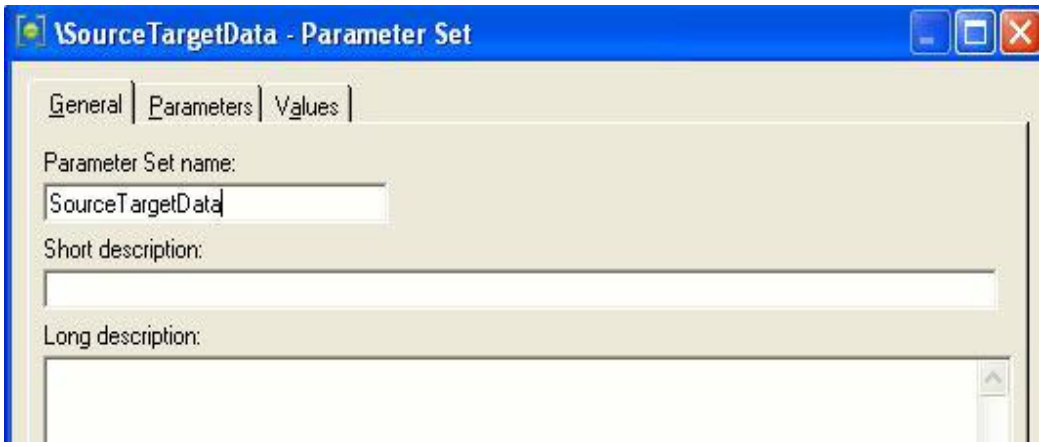
Assumptions:

- Completed PPT training session for Lession 4

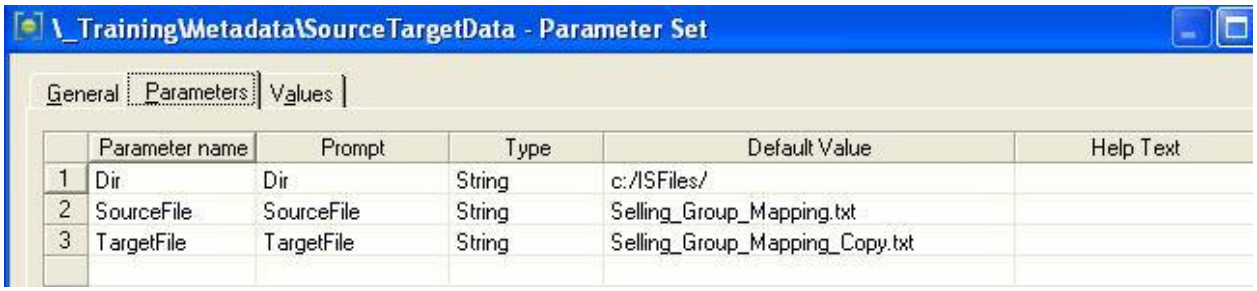
Task: Create a parameter set

1. Click the New button on the Designer toolbar and then open the Other folder.

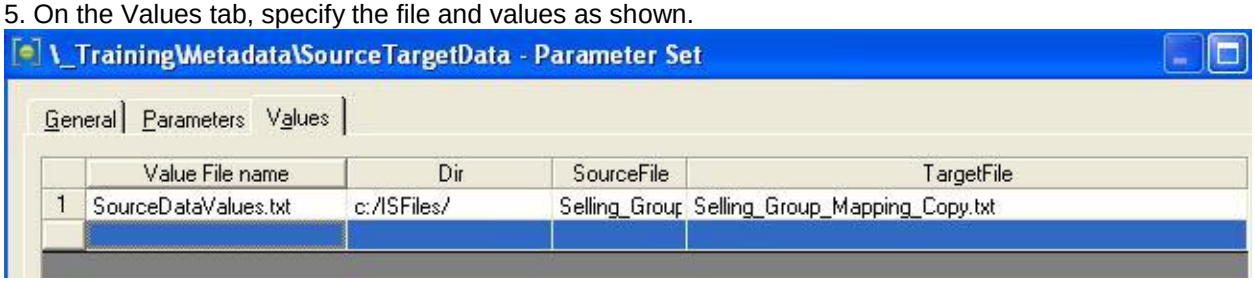
2. Double-click on the Parameter Set icon.
3. On the General tab, name your parameter set SourceTargetData.



4. On the Parameters tab, define the parameters shown.



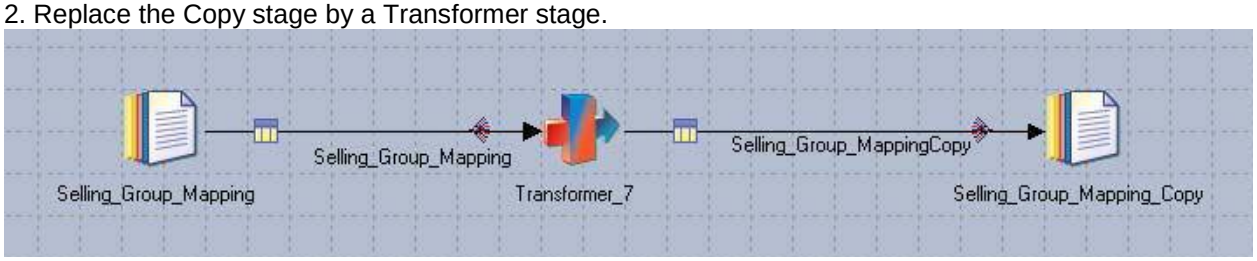
5. On the Values tab, specify the file and values as shown.



6. Save your new parameter set in _Training>Metadata in a folder.

Task: Add a Transformer stage to a job

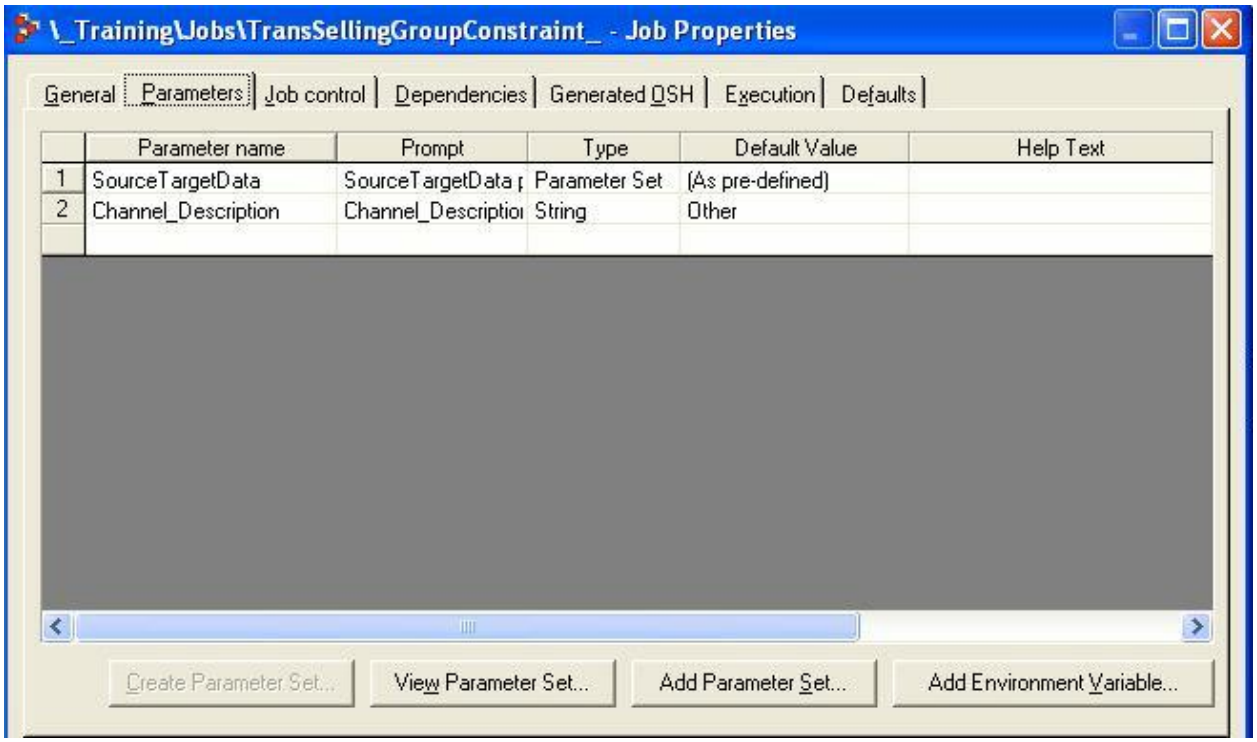
1. Open up your CreateSeqJobParam job and save it as TransSellingGroup.
2. Replace the Copy stage by a Transformer stage.



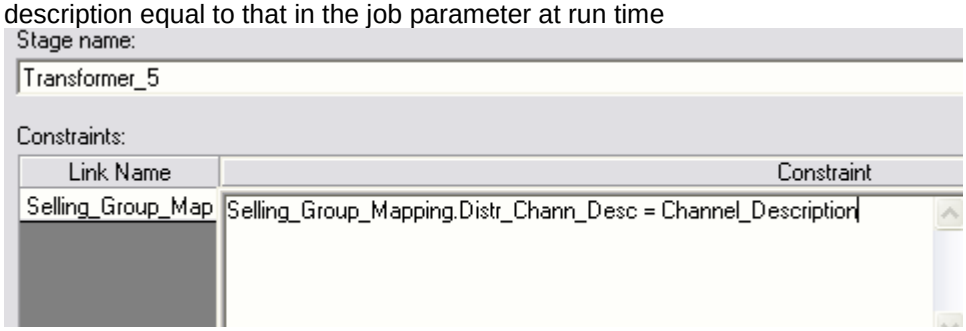
3. Open up the Transformer and map the columns across, just as was done in the Copy stage.
4. Compile and run.
5. Check the job log.

Task: Define a constraint

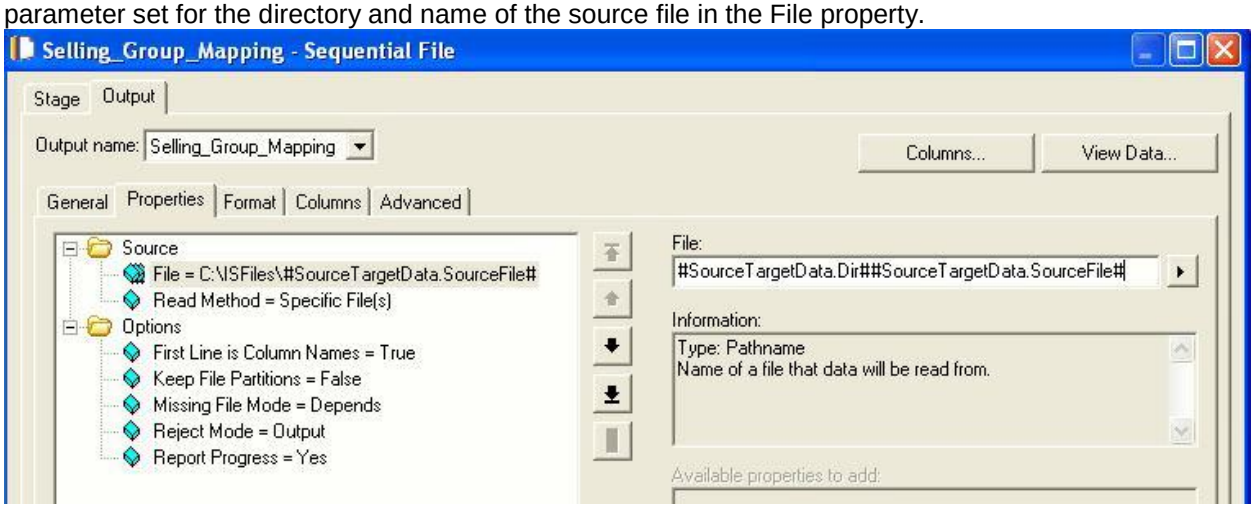
1. Save your job as TransSellingGroupConstraint.
2. Open up your Job Properties to the Parameters window. Delete any existing parameters.
3. Click Add Parameter Set. Select your SourceTargetData parameter set and click OK.
4. Add a new string, job parameter named Channel_Description with a default of “Other” (without the quotes).



5. Open up the Transformer and create a constraint that selects just records with a channel description equal to that in the job parameter at run time

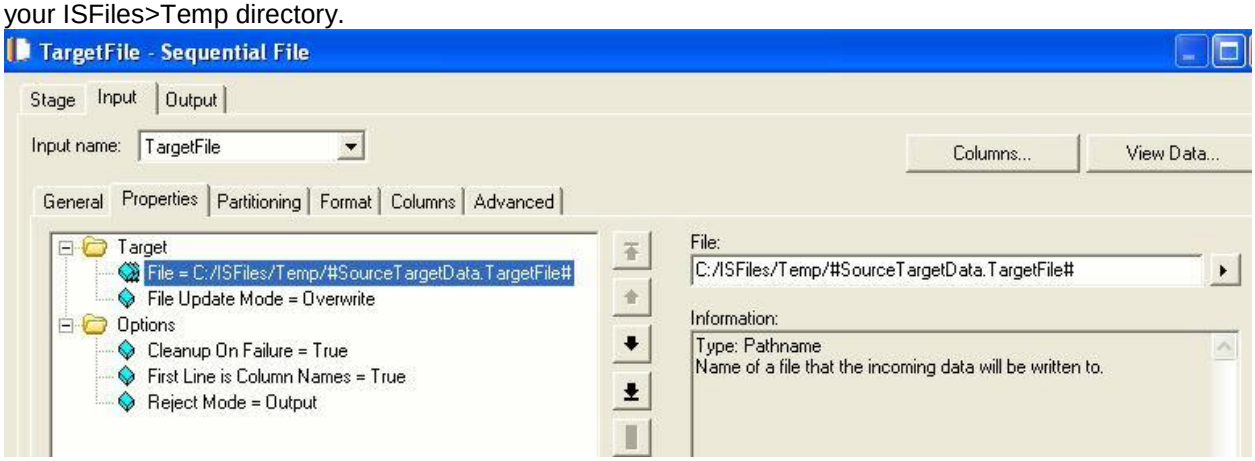


6. Open up your source Sequential stage. Use the Dir and SourceFile parameters from your parameter set for the directory and name of the source file in the File property.



7. Open up your target Sequential stage. Use the TargetFile parameter from your parameter

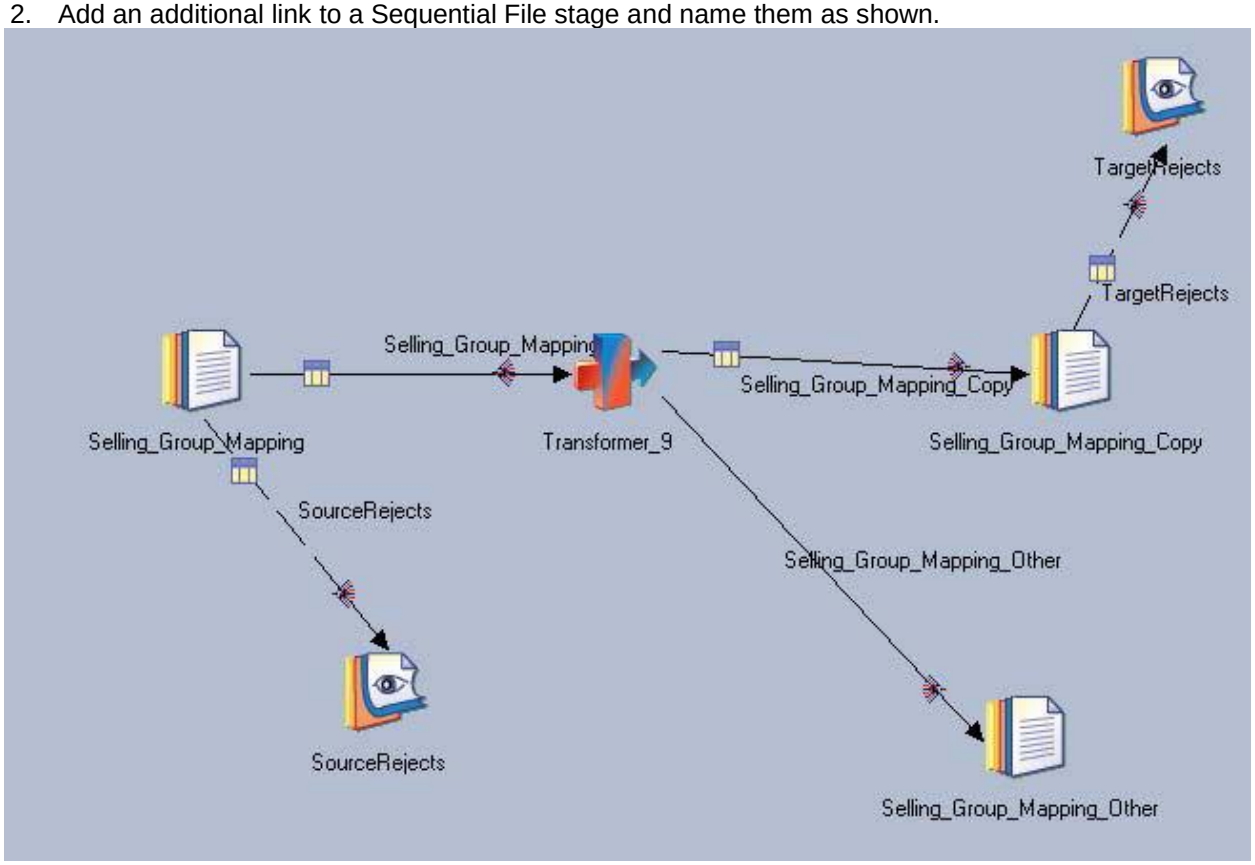
set for the name of the target file in the File property. Hard-code the directory path to your ISFiles>Temp directory.



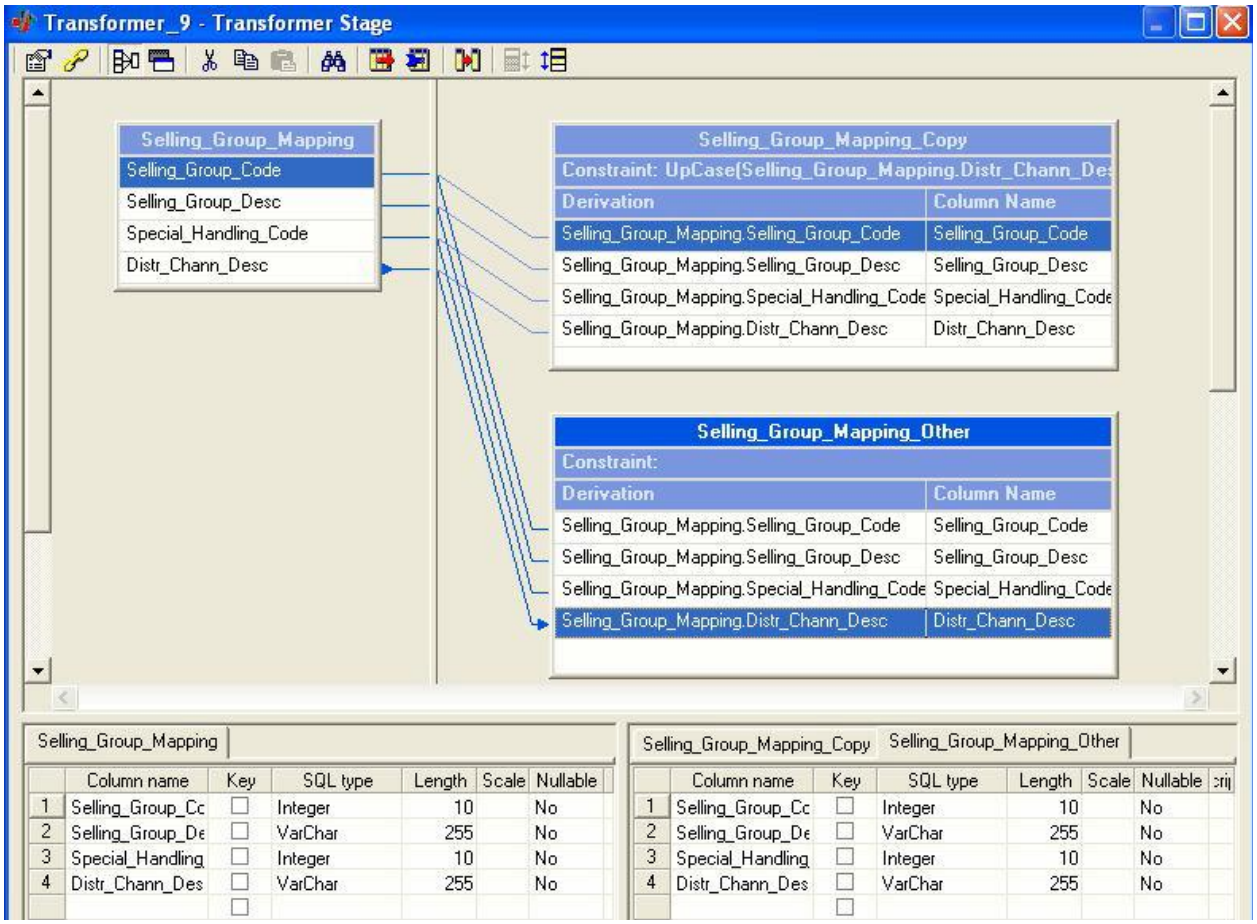
8. Compile and run your job.
9. View the data in the target file to verify that it correctly selects the right rows.
10. Modify your constraint in the Transformer so that descriptions can be entered in upper, lower, or mixed case. Hint: Use Upcase() function.
11. Compile, run, and test your job.

Task: Define an Otherwise Link

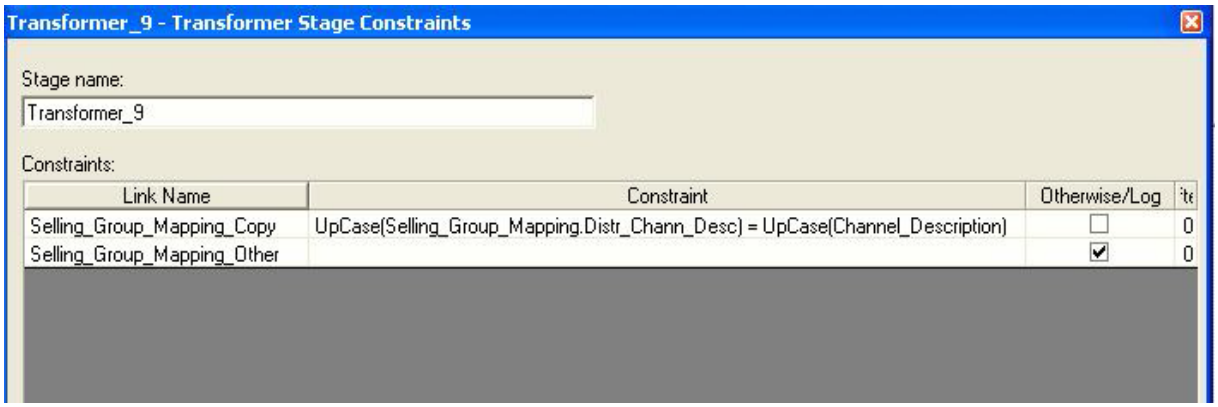
1. Save your job as TransSellingGroupOtherwise.
2. Add an additional link to a Sequential File stage and name them as shown.



3.In the Transformer, map all input columns across to the new target link



4. In the Transformer, reorder the links so that the Selling_Group_Mapping_Other link is last in output link ordering. Use the icon at the top right of the Transformer to accomplish this. (Note: Depending on how you drew your links, this link may already be last.)
5. Open the Constraints window. Select the Otherwise box on your Selling_Group_Mapping_Other link.



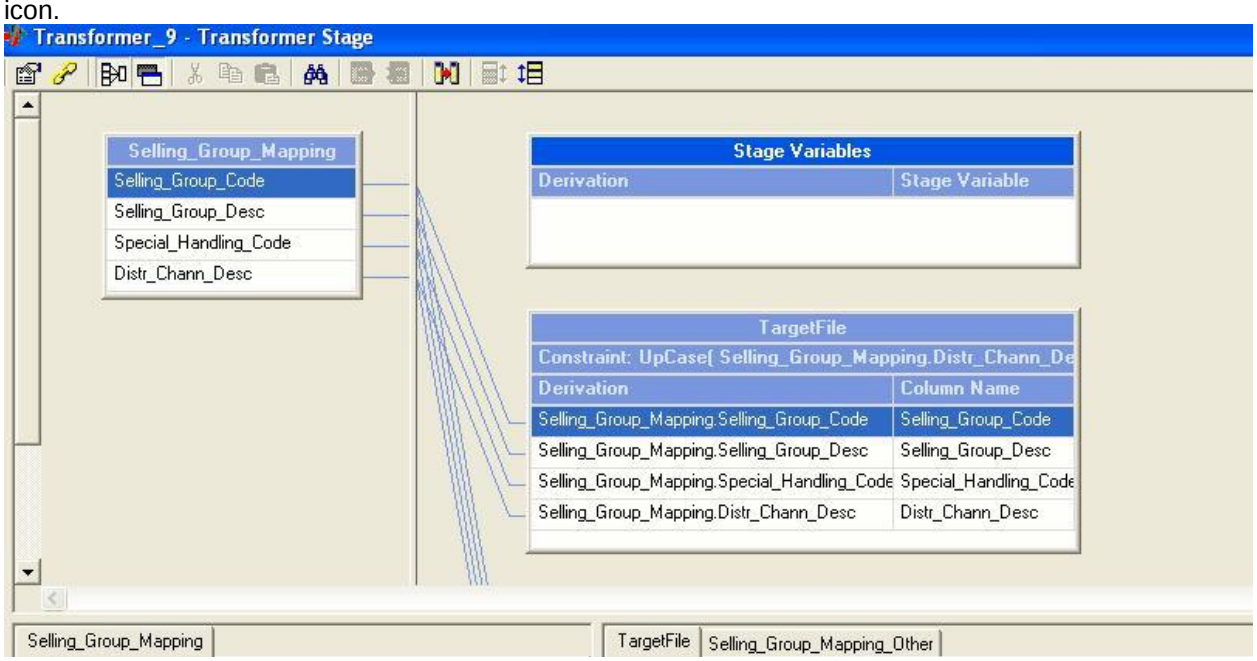
6. Edit the Selling_Group_Mapping_Other Sequential File stage as needed.

7. Compile, run, and test your job. The rows going down the Selling_Group_Mapping_Other link should be all the rows that do not satisfy the constraint on the first link.

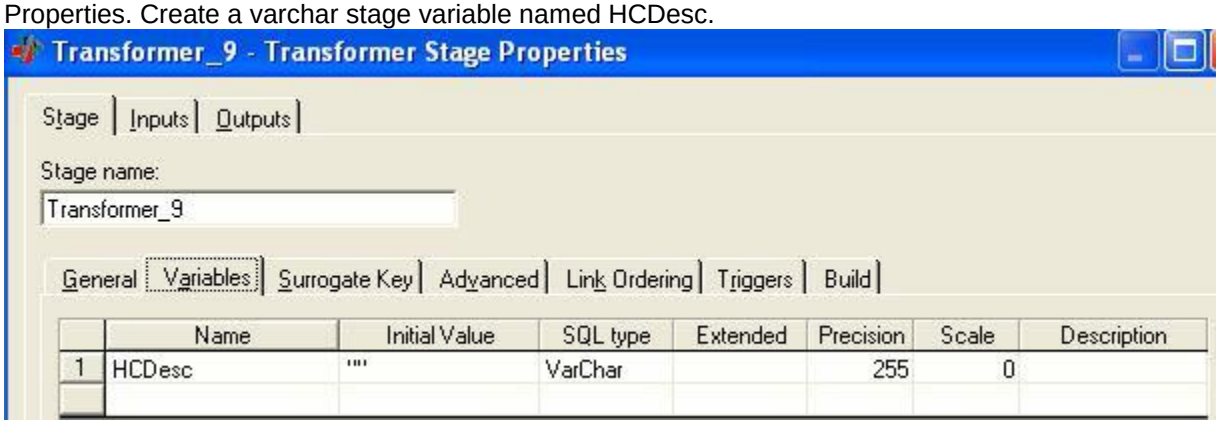
Task: Define derivations

In this task, you define two derivations. The first derivation constructs addresses from several input columns. The second replaces one special handling code by another.

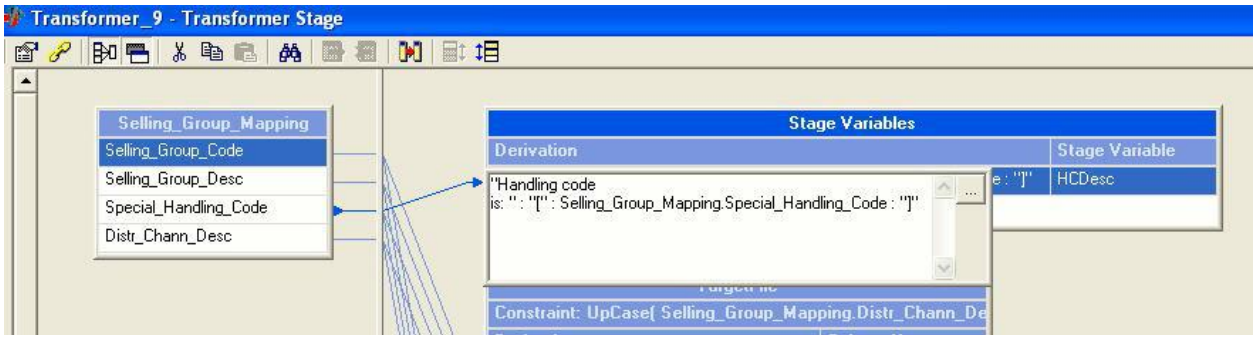
1. Save your job as TransSellingGroupDerivations.
2. Open the Transformer. If you do not see the Stage Variables window at the top right corner, in the toolbar at the top of the Transformer, click the Show/Hide Stage Variables icon.



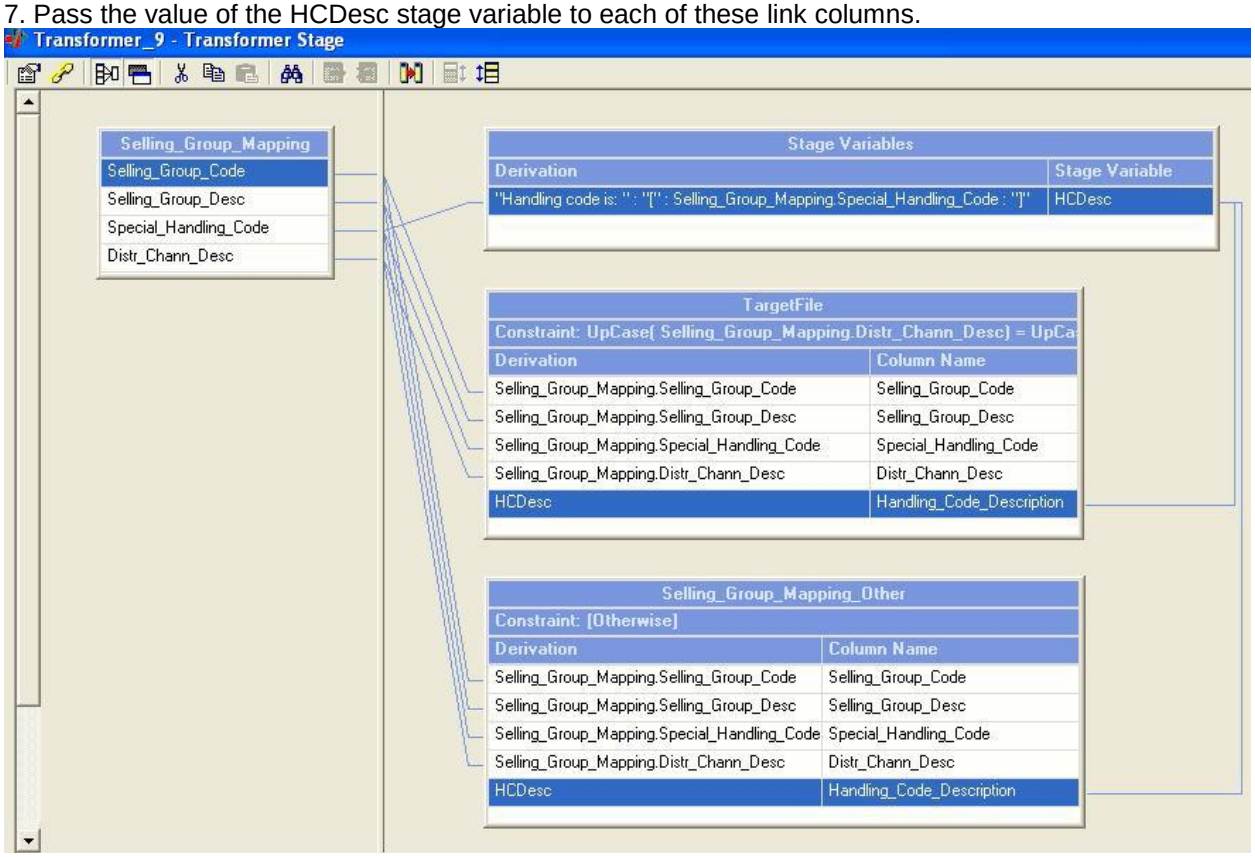
3. Click the right mouse button over the Stage Variables window and click Stage Variable Properties. Create a varchar stage variable named HCDesc.



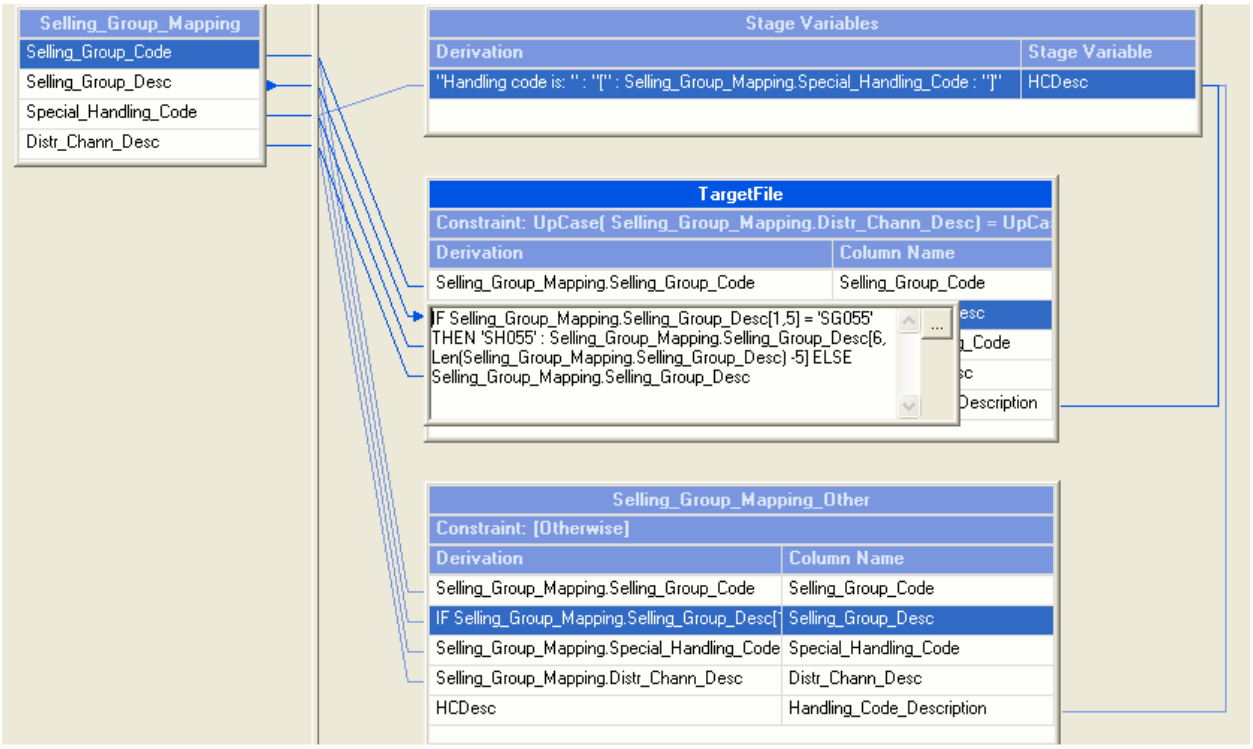
4. Close the Stage Variable Properties window.
5. Double-click in the cell to the left of the HCDesc stage variable. Define a derivation that for each row's Special_Handling_Code produces a string of the following form: "Handling code is: [xxx]". Here "xxx" is the value in the Special_Handling_Code column.



6. Create a new column named Handling_Code_Description for each of the two output links.
7. Pass the value of the HCDesc stage variable to each of these link columns.



8. Write a derivation for the target Selling_Group_Desc columns that replaces “SG055” by “SH055”, leaving the rest of the description as it is. In other words “SG055 Live Swine”, for example, becomes “SH005 Live Swine”. Hint: Use the IF THEN ELSE operator. Also, you may need to use the substring operator and Len functions.



9. Compile, run, and test your job. Here some the output. Notice specifically, the third row (550000), which shows the replacement of SG055 with SH055 in the second column. Also notice the format of the data in the last column.

TransSellingGroupDerivations..TargetFile.TargetFile - Data Browser

Selling_Group_Code	Selling_Group_Desc	Special_Handling_Code	Distr_Chann_Desc	Handling_Code_Description
530000	SG053 TRUCKING	6	Other	Handling code is: [6]
860000	SG086 TRUCKING	6	Other	Handling code is: [6]
1120000	SG112 N.W.A. SWINE	6	Other	Handling code is: [6]
1370000	SG137 EQUITY	6	Other	Handling code is: [6]
3100000	SG310 DOMESTIC SEAFOOD	6	Other	Handling code is: [6]
4030000	SG403 DOMESTIC SEAFOOD	6	Other	Handling code is: [6]
4090000	SG409 SURIMI SALES	6	Other	Handling code is: [6]
9160000	SG916 NATIONAL ACCTS XFER PR	6	Other	Handling code is: [6]
9250000	SG925 CONTRACTS XFER PRICING	6	Other	Handling code is: [6]
150000	SG015 FREEZERS-MILLENNIUM	6	Other	Handling code is: [6]
550000	SH055 LIVE SWINE	6	Other	Handling code is: [6]
870000	SG087 FREEZERS	6	Other	Handling code is: [6]
3110000	SG311 INTERNATIONAL SEAFOOD	6	Other	Handling code is: [6]
4060000	SG406 HOSPITALITY XFER PRICI	6	Other	Handling code is: [6]
4230000	SG423 MEXICAN ORIGINAL SALES	6	Other	Handling code is: [6]
4430000	SG443 MALLARD'S SALES	6	Other	Handling code is: [6]
5550000	SG555 COBB VANTRESS	6	Other	Handling code is: [6]
9130000	SG913 DISTRIBUTION XFER PRIC	6	Other	Handling code is: [6]
9230000	SG923 NEW COMMODITY XFER PRI	6	Other	Handling code is: [6]
9990000	SG999 MISCELLANEOUS	6	Other	Handling code is: [6]

Lab 07: Work with Relational Data

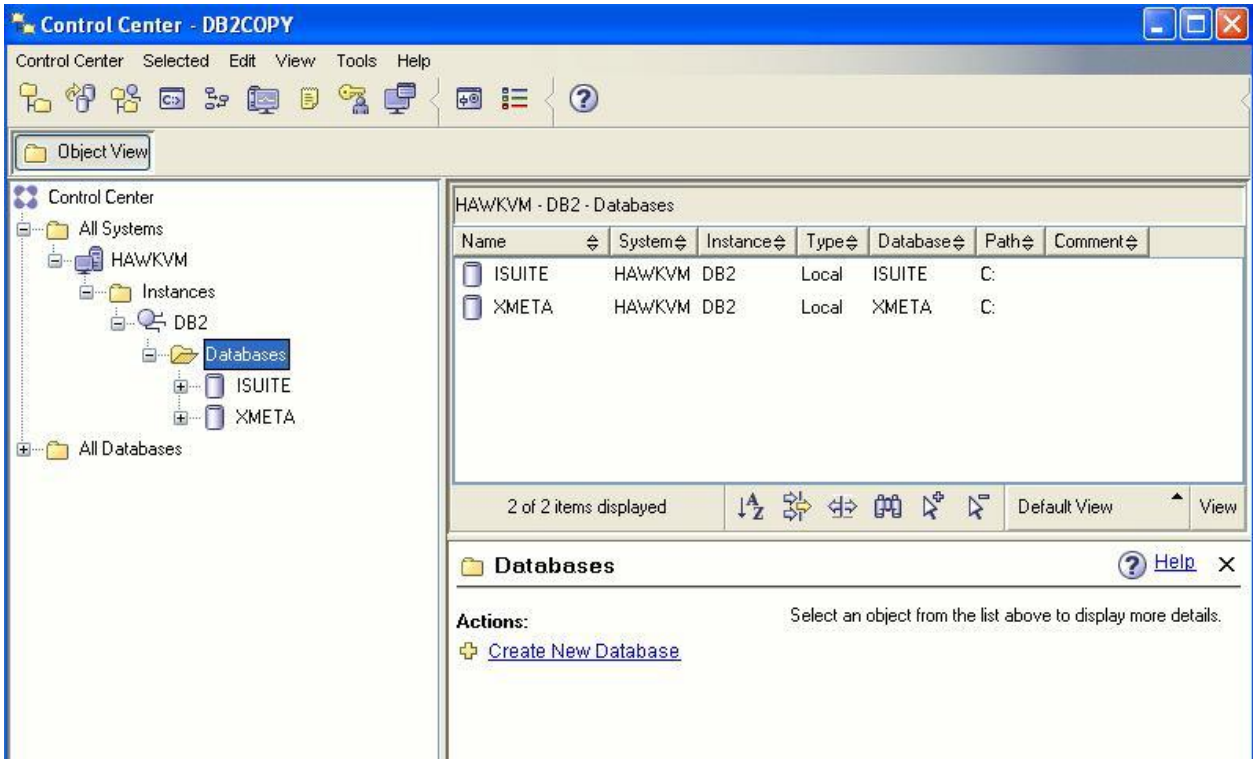
Assumptions:

- You have access to DB2
- You have a working LookupWarehouseItem job
- Completed PPT training session for Lession 4

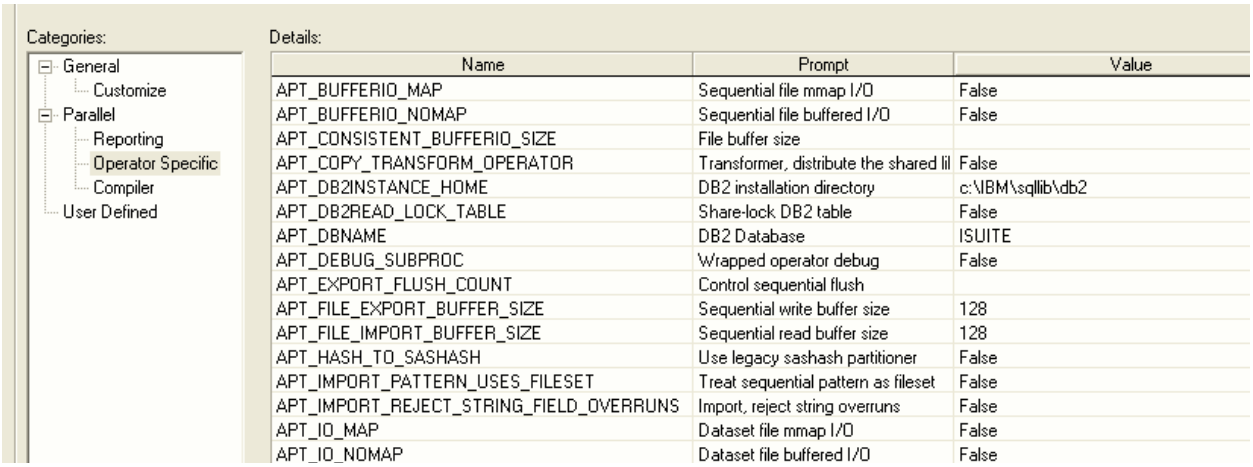
Task: DB2 Setup

*****NOTE:** The steps in this task only need to be performed if DataStage and DB2 have not been set up to work with each other. You instructor will give you this information.

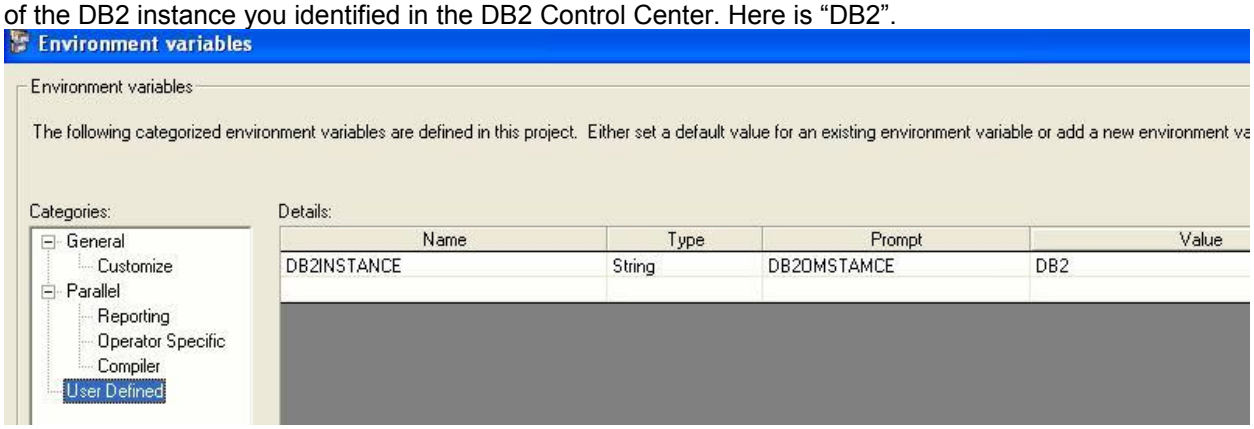
1. Click on your DB2 icon and select Control Center.
2. Click on the All Databases folder. If this is the DB2 instance with the Information Server Repository you will see the Repository database, XMETA.
3. Click Create New Database named ISUITE. Click the Finish button to create the database.



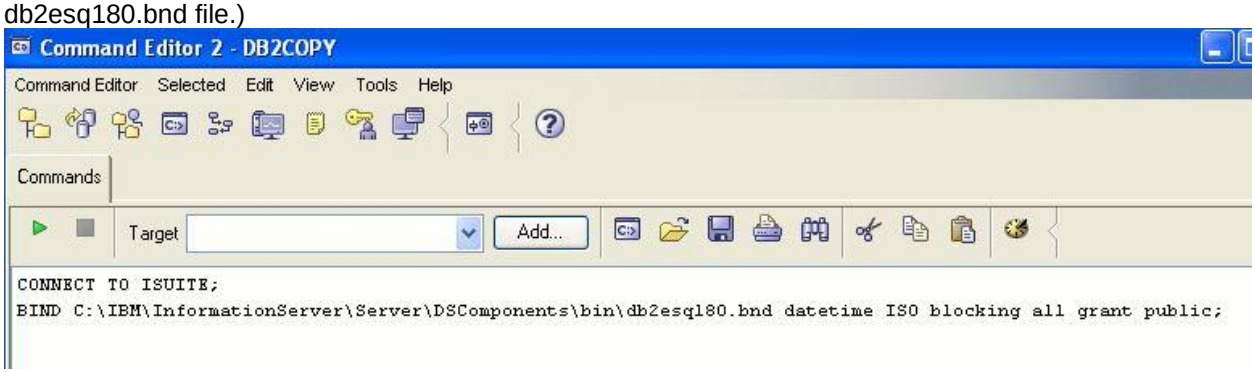
4. Make a note of your DB2 instance name under the Instances folder. Here it is “DB2”. Keep the Control Center open.
5. Open up Administrator, select your project, click the Properties button, and then click the Environment button. Click the Operator Specific folder. Set the APT_DB2INSTANCE_HOME variable to the DB2 installation directory. The default is shown below. Set the APT_DBNAME variable to a database (ISUITE) that you want to be the default.



6. Click the User-Defined folder. Create a variable named DB2INSTANCE. Set it to the name of the DB2 instance you identified in the DB2 Control Center. Here is “DB2”.



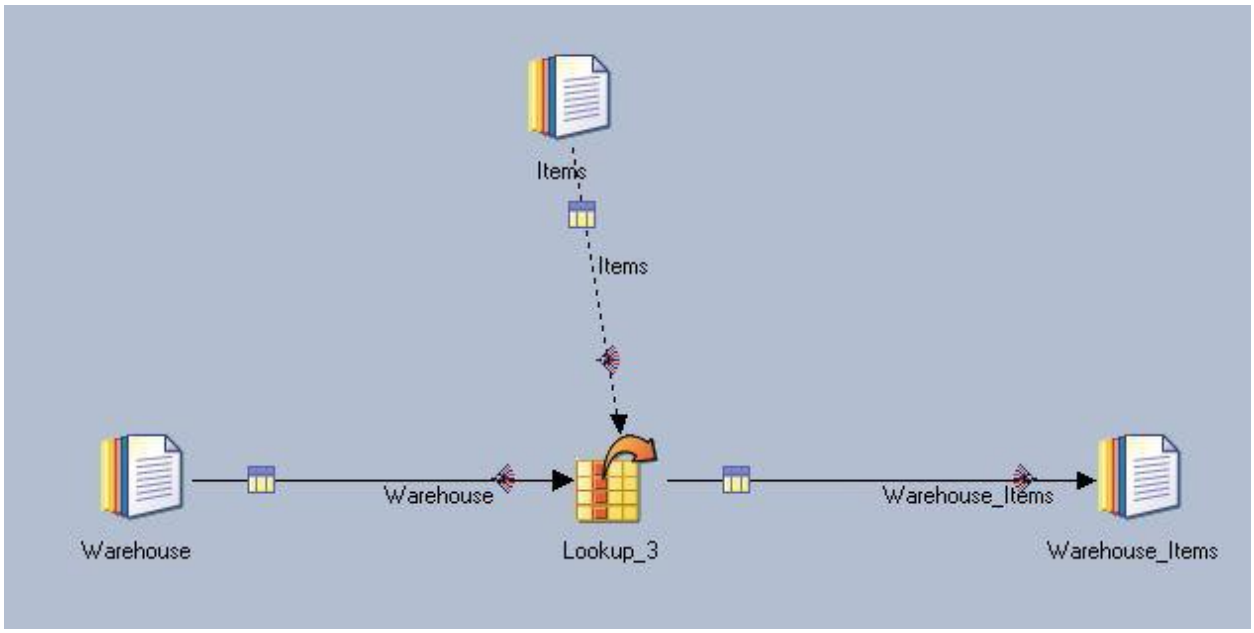
7. Open up the DB2 Control Center. Click the Command Editor button in the toolbar. Type the commands shown (or copy them from the DB2 Bind.txt file in your ISFiles folder). The first command connects to the database you created. The second binds DataStage components to this database. (Note: You may need to alter the directory path to the db2esq180.bnd file.)



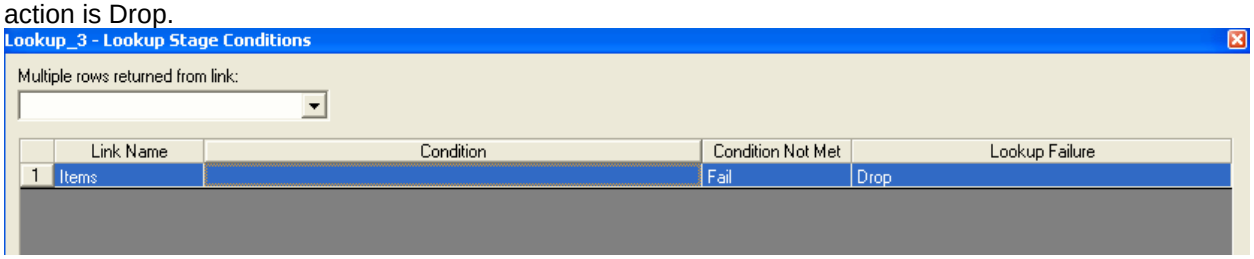
8. Click the Execute button

Task: Create and load a DB2 table using the DB2 Enterprise stage

1. Open up your LookupWarehouseItem job. You’re your job as LookupWarehouseItemDB2.

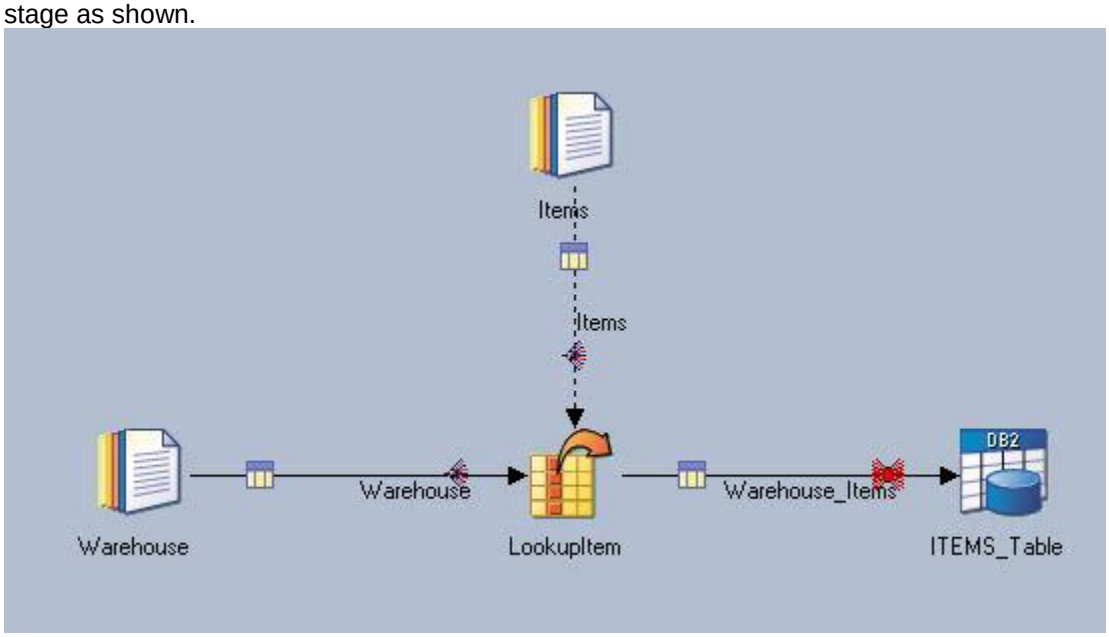


2. Open up the Lookup stage and click the Constraints icon. Specify that the Lookup Failure action is Drop.

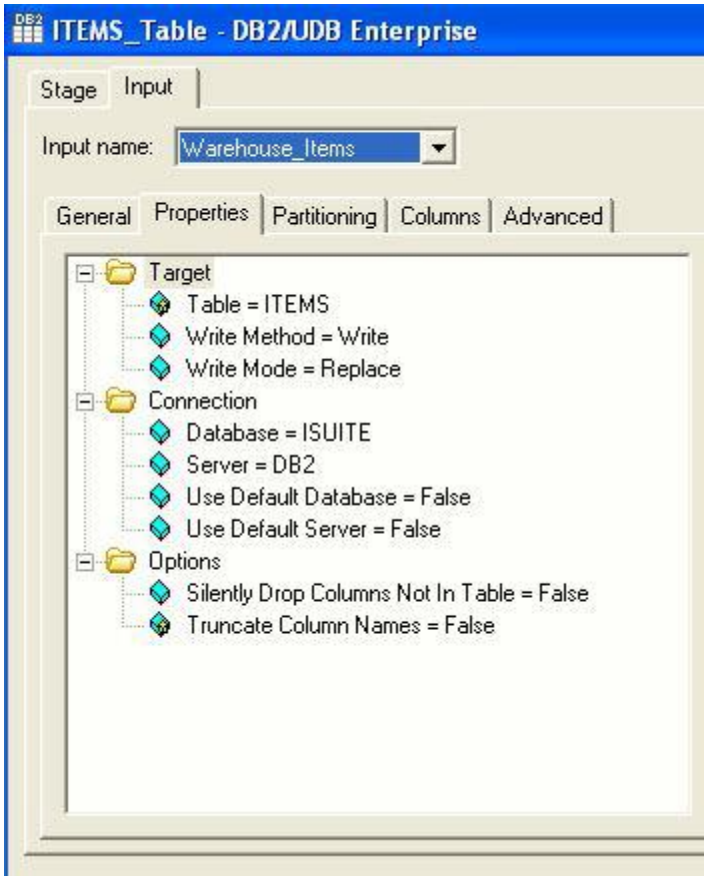


3. ***Important:** Compile and run your job and verify that it is working before you proceed.

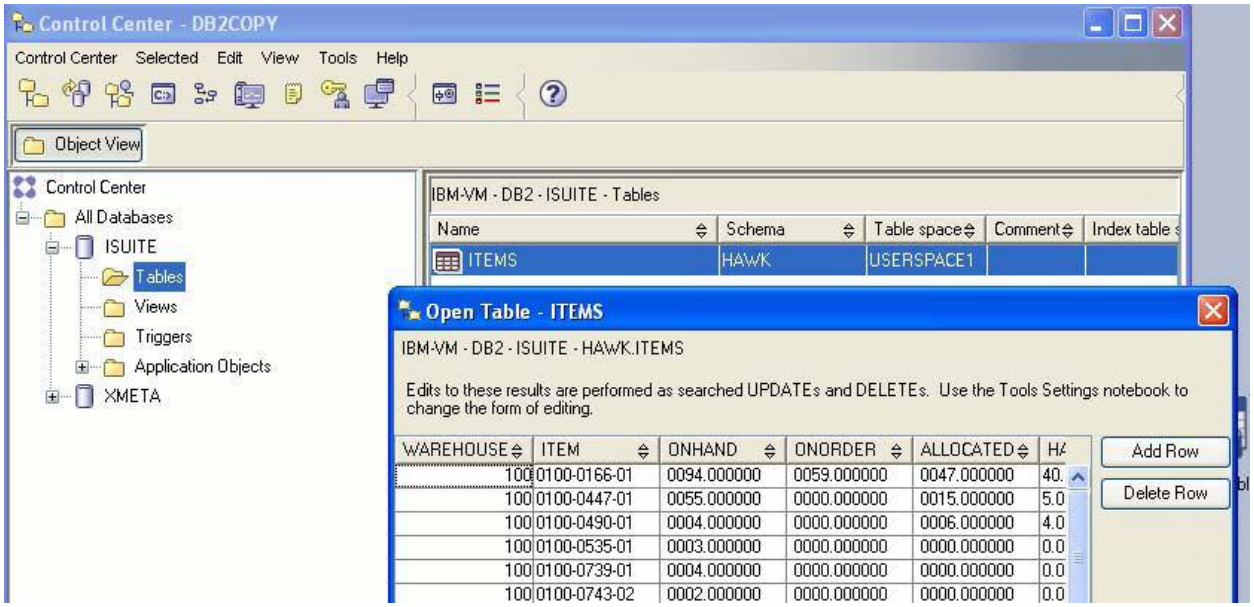
stage as shown.



5. Edit the DB2 stage as shown. Set the Write Method property to Write. Set the Write Mode property to Replace. Set the Use Default Database and Use Default Sever properties to False. This displays the Database and Server properties. Set their values as shown.



6. Compile and run. Check the job log for errors.
7. Open up the DB2 Control Center. Double-click on the name of your ITEMS table in the ISUITE database and view the data.

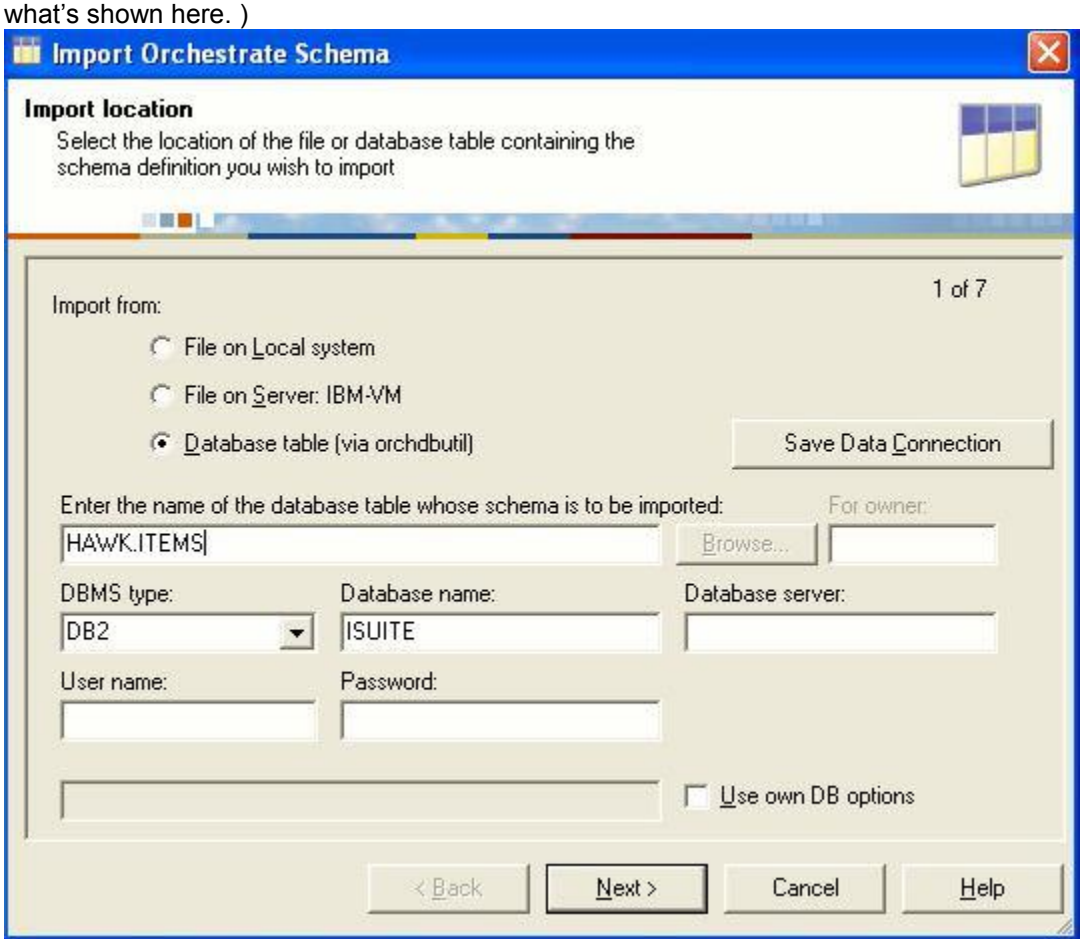


8. Make a note of the Schema name of the ITEMS table you created. (Here, the schema name is HAWK.)

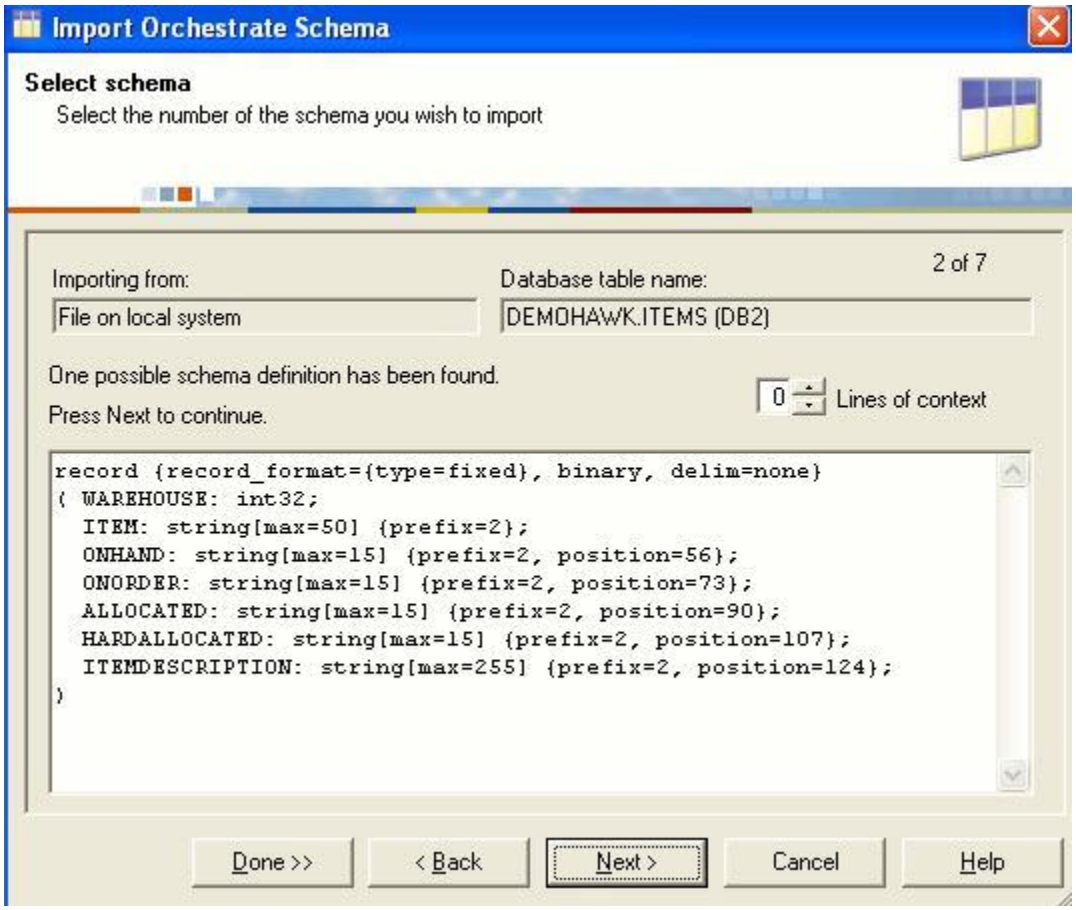
Task: Import a table definition using orchdbutil

1. In Designer, click the Import menu and then select Table Definitions. Then select

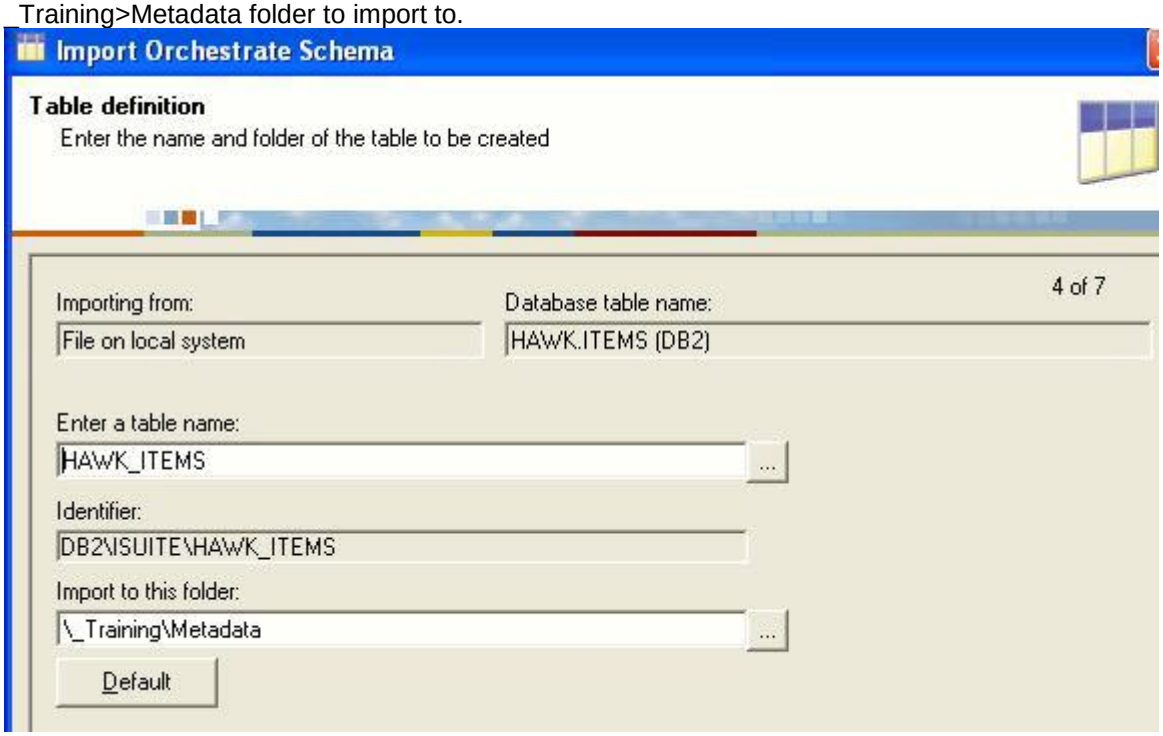
Orchestrate schema definitions.
2. In the Import Orchestrate Schema window, select the “Database table (via orchdbutil)” button.
3. Fill in the information needed to import a table definition for the ITEMS table in your DB2 database named ISUITE. (Note: Your schema name may be different than what’s shown here.)



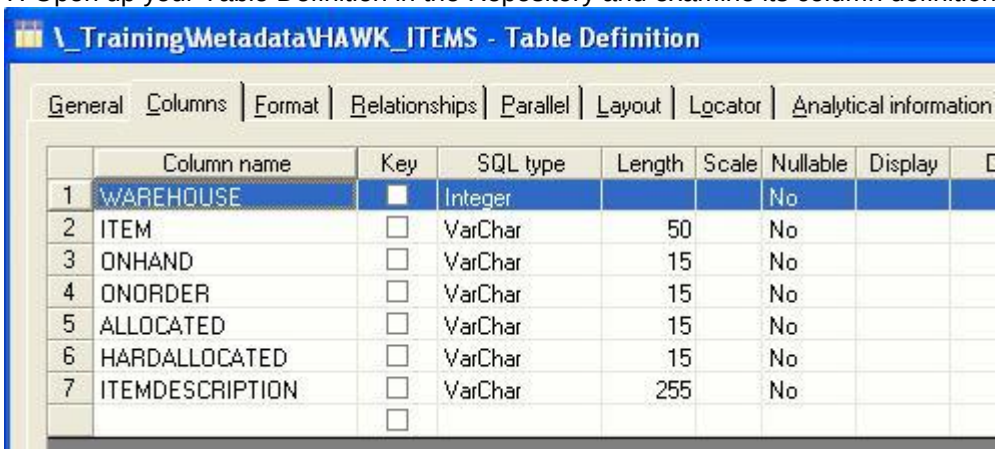
4. Click Next. You should see the schema definition of ITEMS displayed in the window.



5. Click Next until you move to the Table definition window. Select your _Training>Metadata folder to import to.



6. Complete the import process.
7. Open up your Table Definition in the Repository and examine its column definitions.



	Column name	Key	SQL type	Length	Scale	Nullable	Display
1	WAREHOUSE	☐	Integer			No	
2	ITEM	☐	VarChar	50		No	
3	ONHAND	☐	VarChar	15		No	
4	ONORDER	☐	VarChar	15		No	
5	ALLOCATED	☐	VarChar	15		No	
6	HARDALLOCATED	☐	VarChar	15		No	
7	ITEMDESCRIPTION	☐	VarChar	255		No	
		☐					

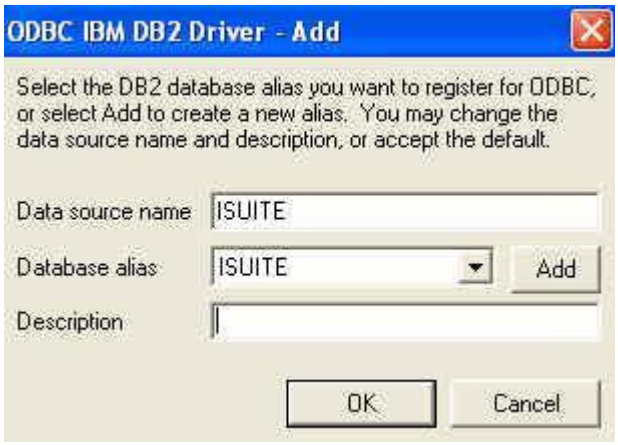
8. Click on the Locator tab and examine its contents. Be sure all the information is correct, especially DATABASE, DataSchema, and TABLE. This information will be used in the SQL Builder.

9. Close the Table Definition.

Task: Create an ODBC Data Source

This step is necessary only if a DB2 ODBC datasource named ISUITE hasn't been set up for you already. Check with your instructor.

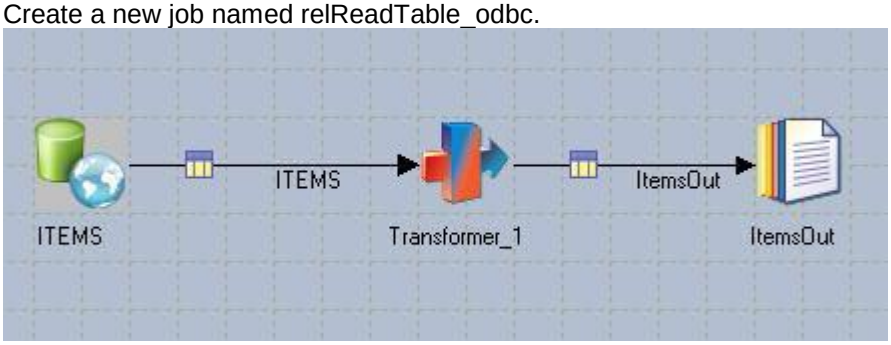
1. Open up your operating sytem ODBC Data Source Administrator.
2. Click the System DSN tab.
3. Click Add.
4. Select the IBM DB2 ODBC driver.
5. The data source name is ISUITE. The Database alias is ISUITE.



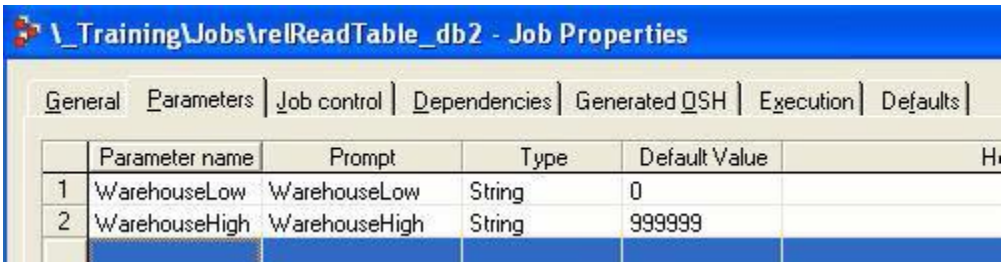
6. Click OK.

Task: Create a job that reads from a DB2 table using the ODBC Connector stage

1. Create a new job named relReadTable_odbc.

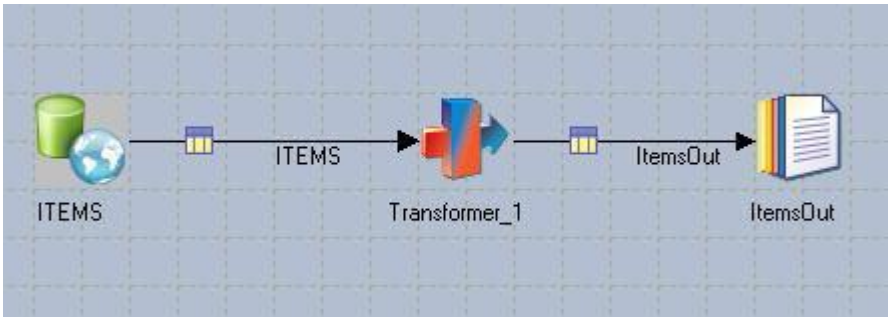


2. Create two job parameters. The first is named WarehouseLow with a default value of 0. The second is named WarehouseHigh with a default value of 999999.



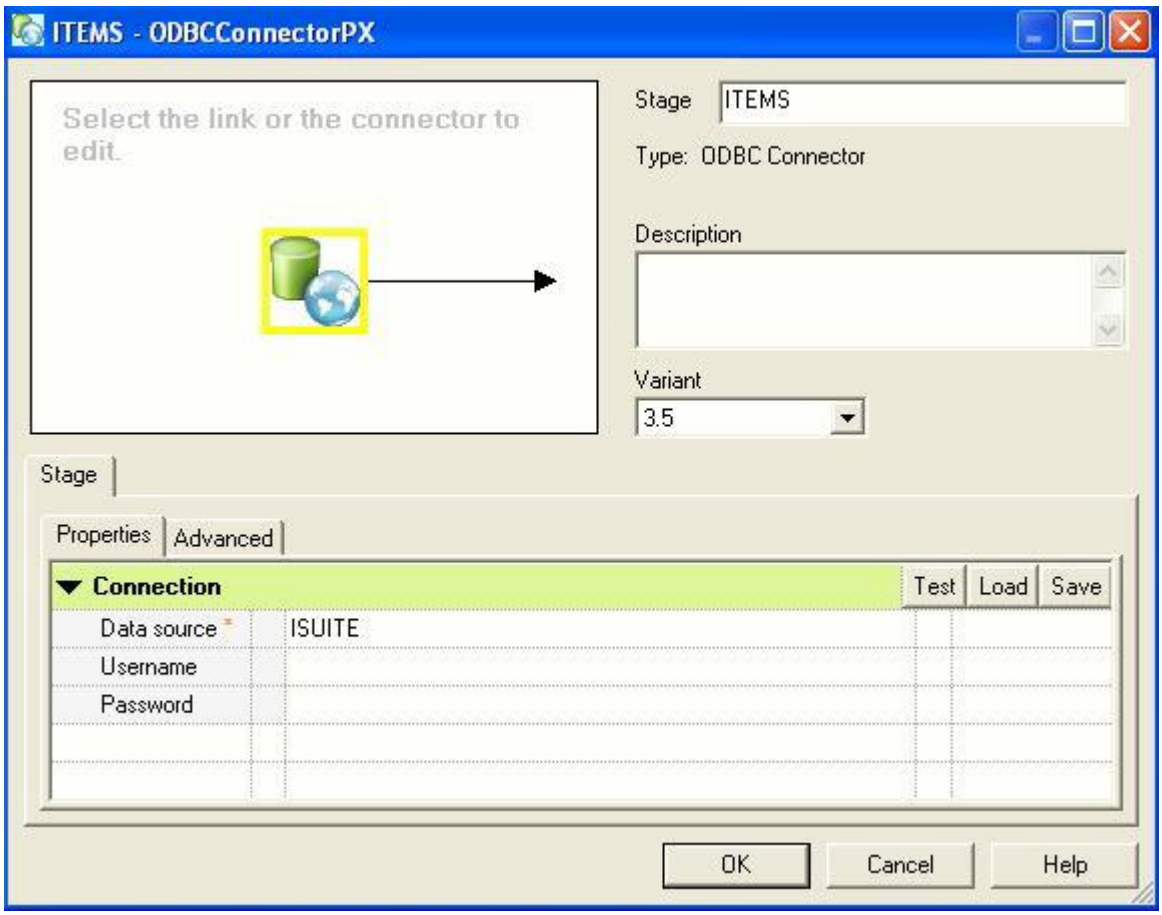
Task: Create a job that reads from a DB2 table using the ODBC Connector stage

1. Create a new job named relReadTable_odbc



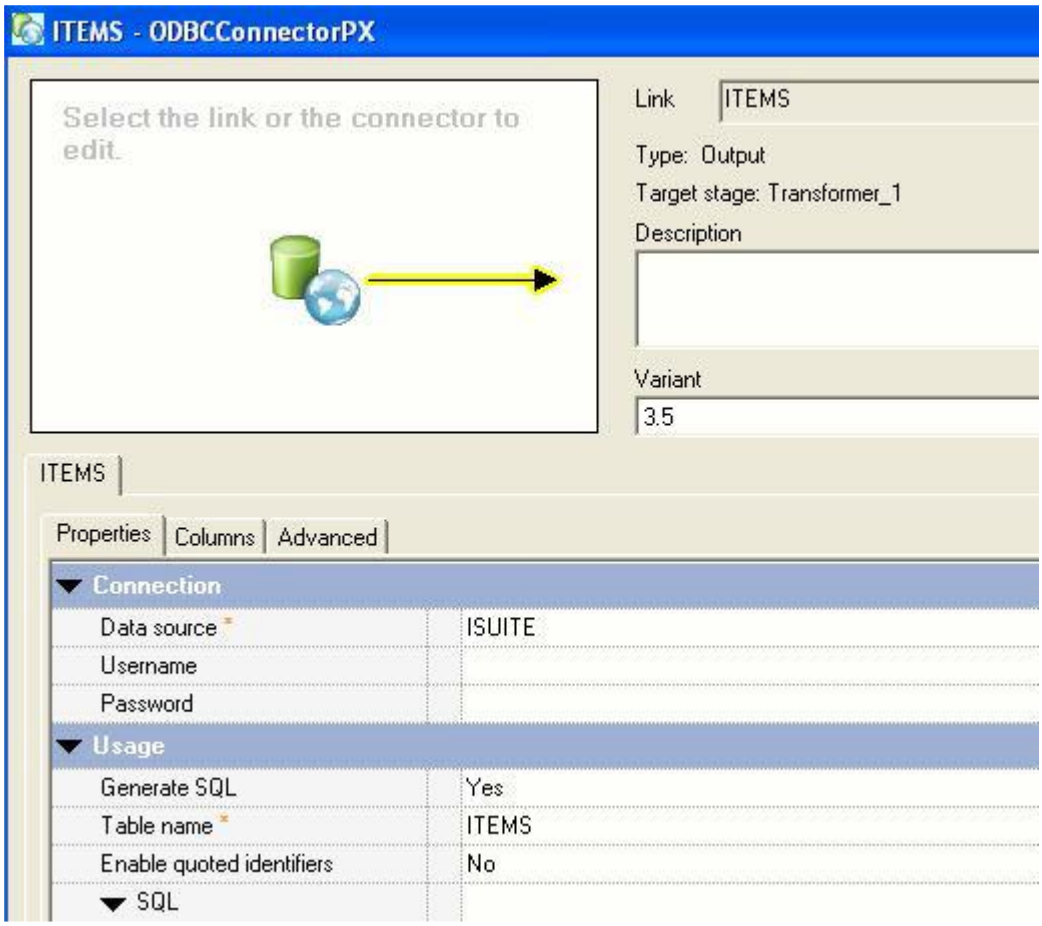
2. Create two job parameters. The first is named WarehouseLow with a default value of 0. The second is named WarehouseHigh with a default value of 999999.

3. Open up the ITEMS stage to the Properties tab. Click the Data Source cell and then click the Data Source button. Select ISUITE. Click Test to test the connection.



4. Click the output link in the Navigator panel. Click the Columns tab. Load your ITEMS table definition.

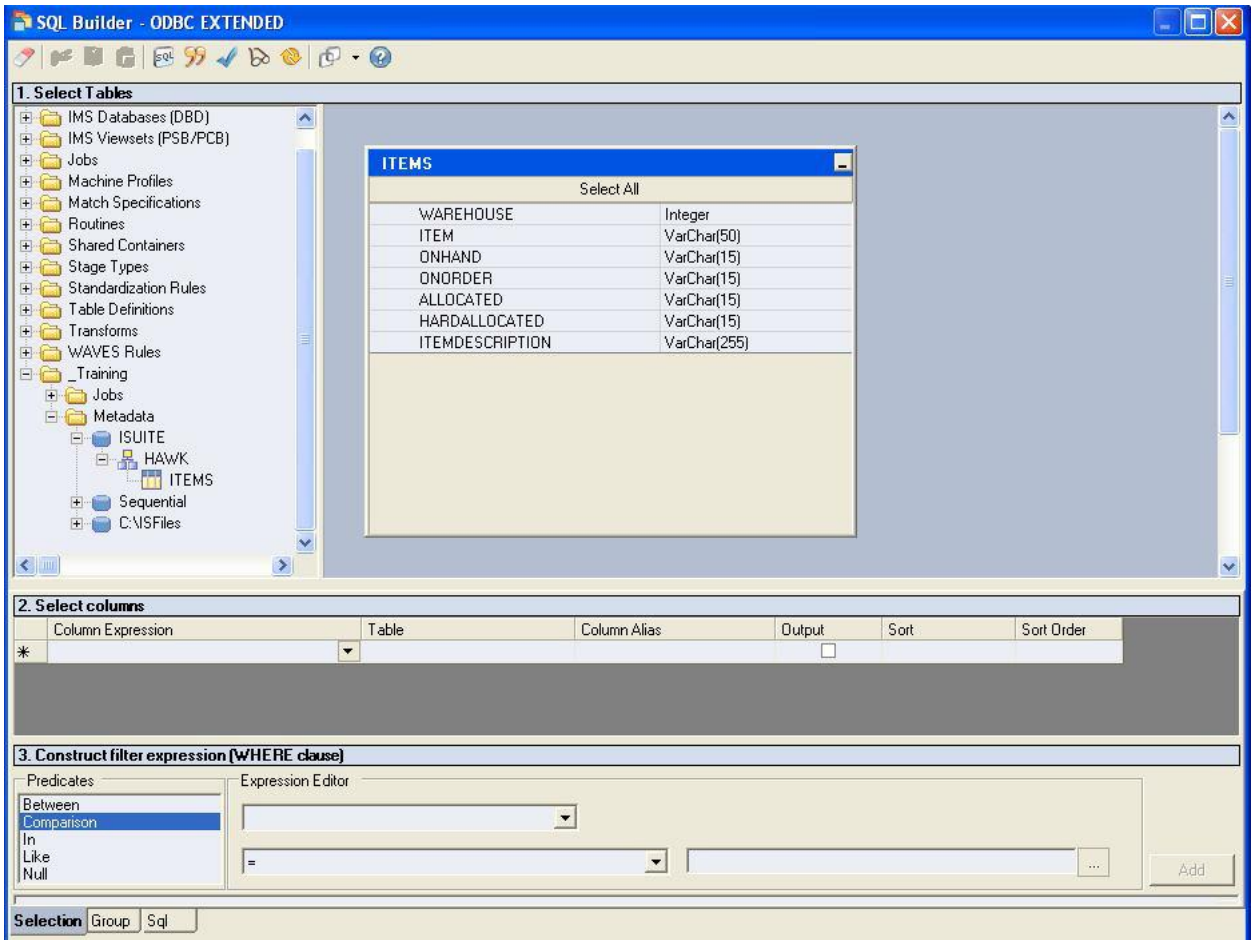
5. Click the Properties tab. In the Usage folder, set the Generate SQL property to Yes. Type in the name of the table, i.e., ITEMS.



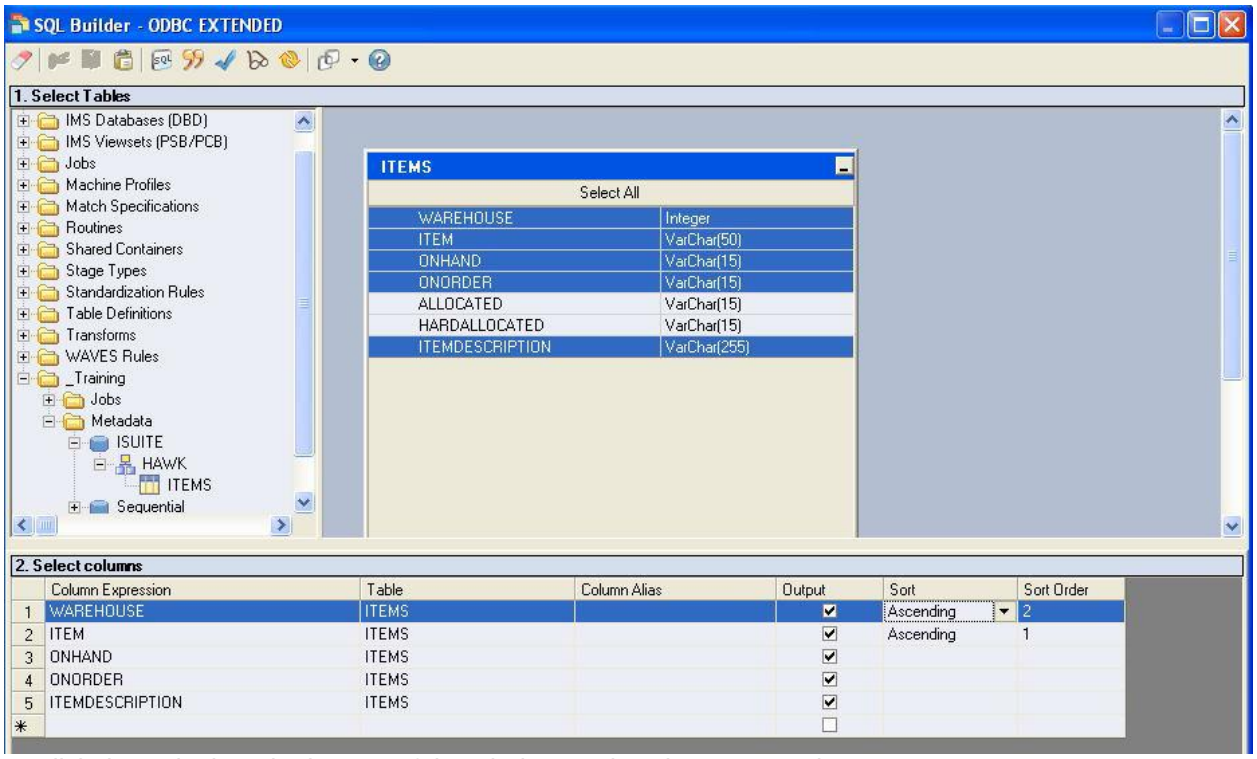
6. In the Transformer stage map all columns across.
7. In the target Sequential stage, write to a file named ITEMS.txt.
8. Compile and run your job.

Task: Build an SQL SELECT statement using SQL Builder

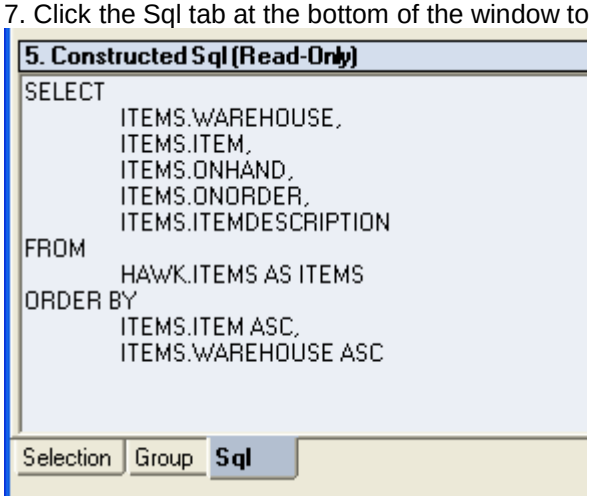
1. Save your job as relReadTable_odbc_sqlBuild.
2. Open up the Connector source stage. In the Usage folder, set the Generate SQL property to No.
3. Click the Select statement cell and then click the Build. Select the extended syntax. This opens the SQL Builder window.
4. Drag your ITEMS Table Definition onto the canvas.



5. Select all the columns except ALLOCATED and HARDALLOCATED and drag them to the Select columns window.
6. Sort by ITEM and WAREHOUSE in that order, ascending. To accomplish this select Ascending in the Sort column. Specify the sort order in the last column.



7. Click the Sql tab at the bottom of the window to view the generated SQL.

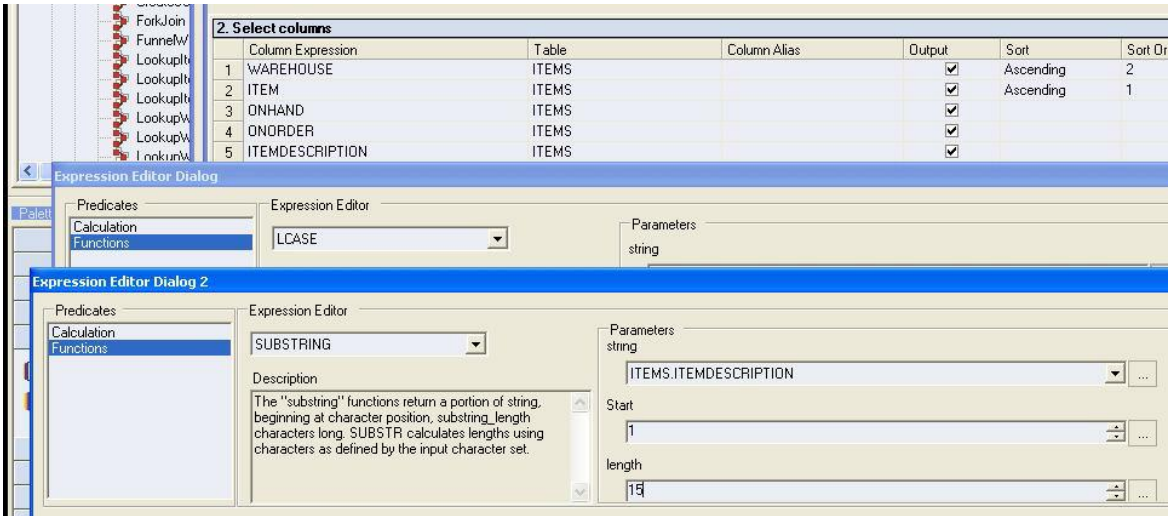


8. Click OK to save and close your SQL statement. You may get some warning messages. Accept the SQL as generated and allow DataStage to merge the SQL Builder selected columns with the columns on the Columns tab.
9. In the Transformer stage remove the ALLOCATED and HARDALLOCATED columns from the output or verify that they have been removed.
10. Compile and run. View the job log.
11. View the data in the target stage.

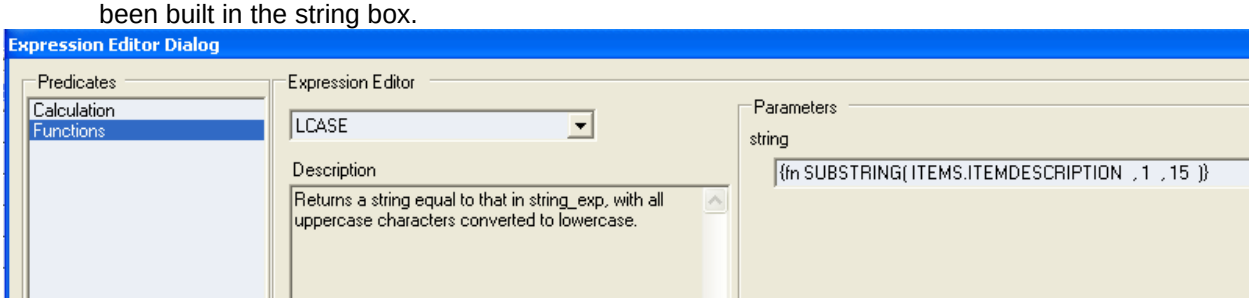
Task: Use the SQL Builder expression editor

1. Save your job as relReadTable_ODBC_expr.
2. Open up your source ODBC Connector stage. Then open SQL Builder for the SELECT statement.

3. Click in the empty Column Expression cell after the last listed column, ITEM DESCRIPTION. Select Expression editor from the drop-down list. This opens the Expression Editor Dialog window.
4. In the Expression Editor box select the Functions predicate and then select the LCASE function. In the Parameters string box, select Expression Editor again to open a second Expression Editor Dialog.
5. In this use the SUBSTRING function to select the first 15 characters of the ITEMDESCRIPTION column.

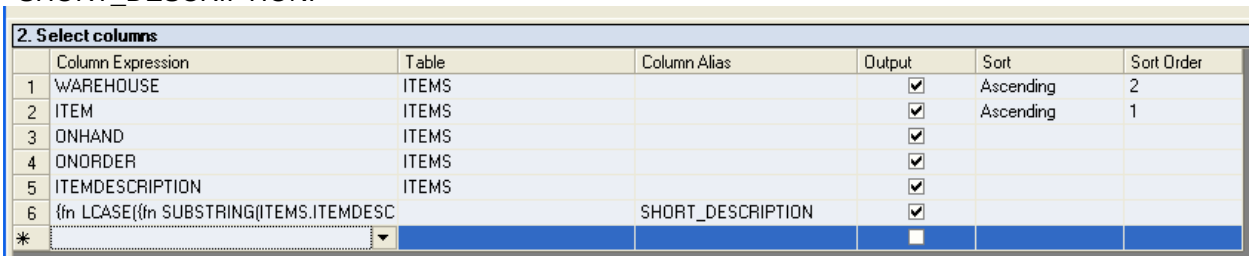


1. Click OK to return to the first Expression Editor window. Notice the expression that has been built in the string box.

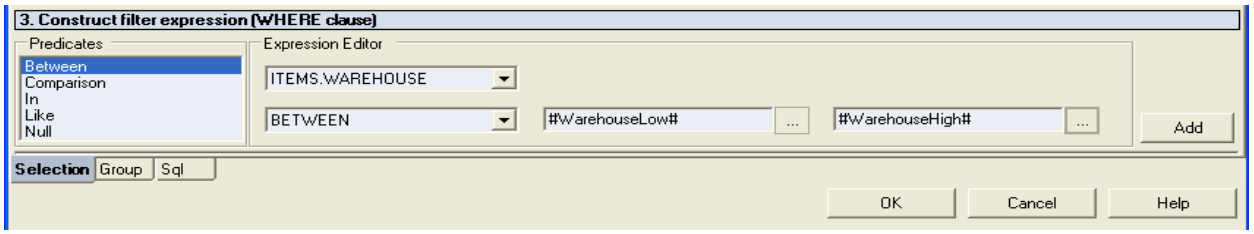


2. Click OK.

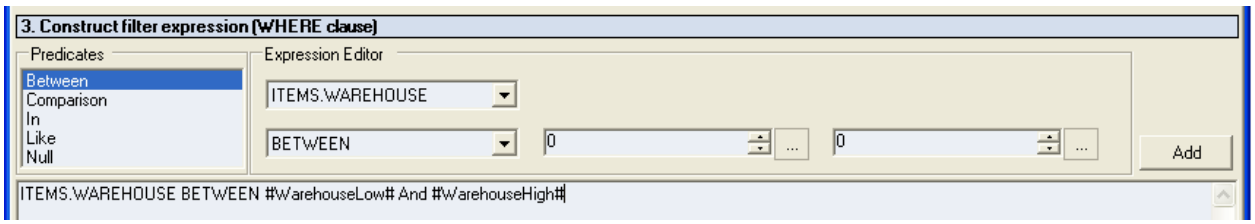
3. Close down the Expression Editor windows. Specify a Column Alias named SHORT_DESCRIPTION.



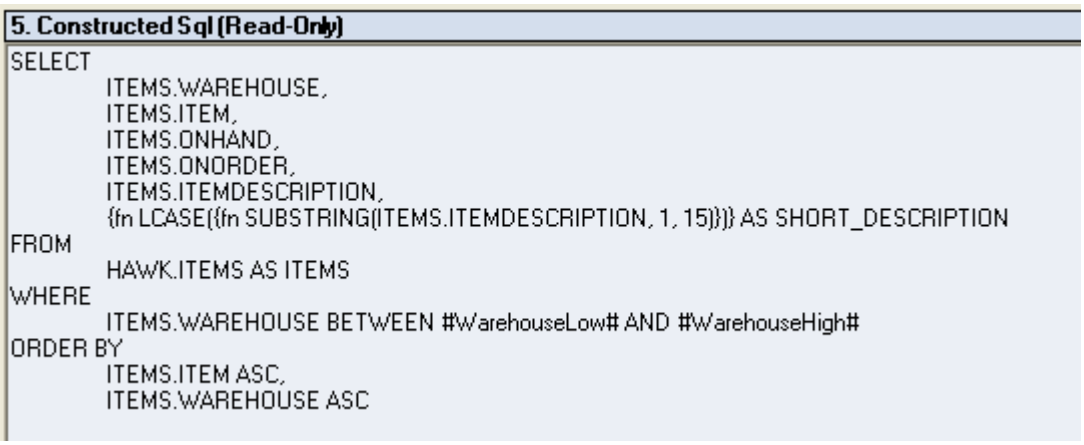
4. In the Construct filter expression window, specify the following WHERE clause: Warehouses with numbers between WarehouseLow and WarehouseHigh, where these are job parameters.



5. Click the Add button to add it to the Selection window.



6. Click the Sql tab at the bottom of the SQL Builder to view the constructed SQL. Verify that it is correct.

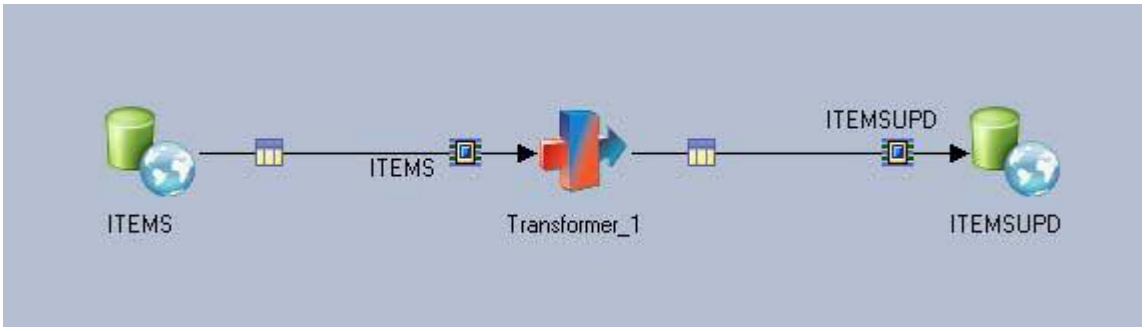


7. Click OK to return to the Properties tab. A message is displayed informing you that your columns in the stage don't match columns in the SQL statement. Click Yes so that DataStage will add the SHORT_DECRIPTION column to your metadata.
8. On the Columns tab, specify the type correctly for the SHORT_DESCRIPTION column, Varchar(15).

14. Open the Transformer and map the new SHORT_DESCRIPTION column across.
15. Compile and run. Try out different parameter values. View the log.
16. View the data.

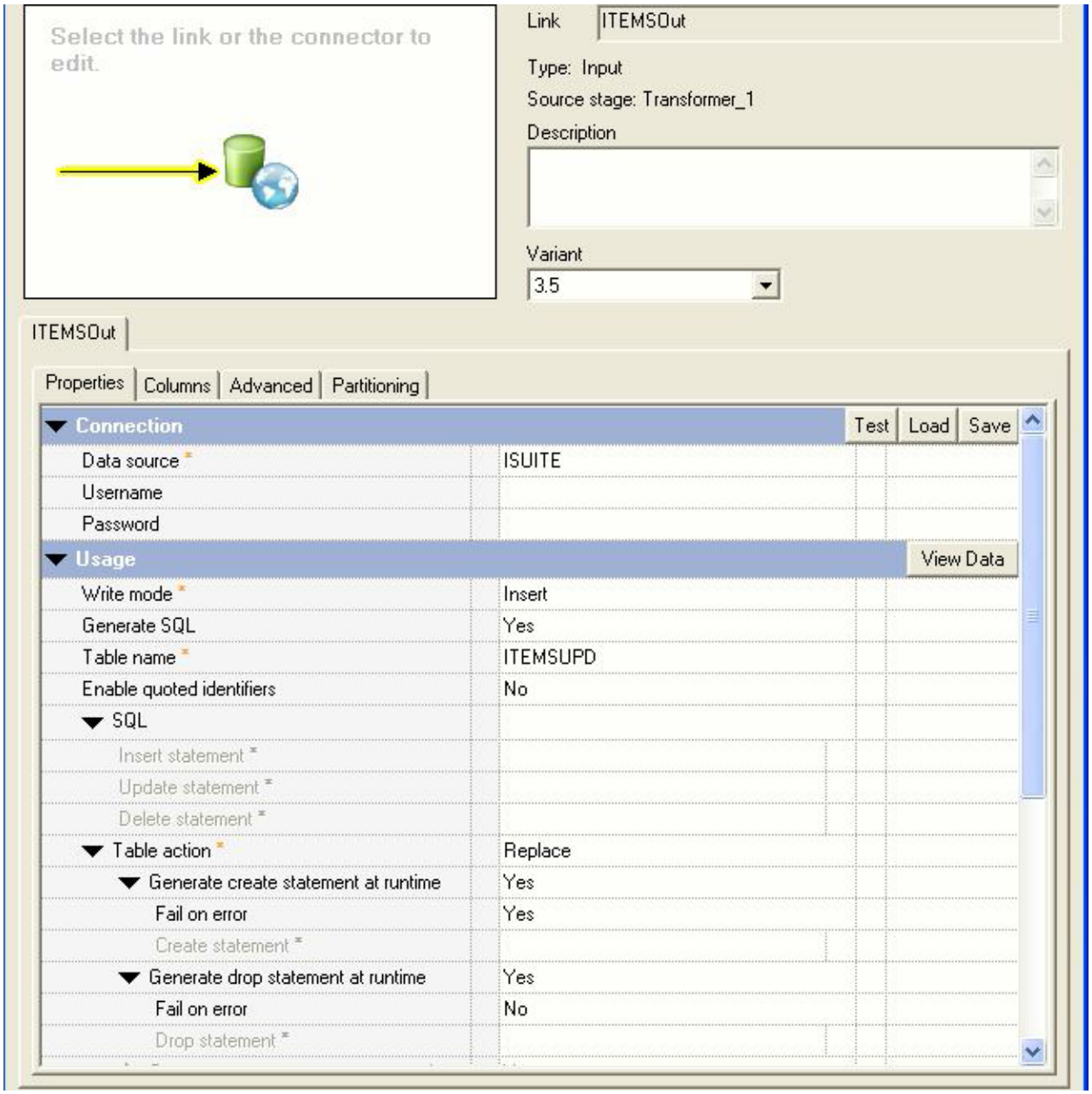
Task: Create and write to a DB2 table

1. Save your job as relWriteTable_ODBC. Replace your target Sequential File stage with an ODBC Connector stage. Name the links and stages as shown



2. Open the target Connector stage to the Properties tab. Click the input link in the Navigator panel. Set the properties as shown. The Data source is ISUITE. The Write Mode is Insert. The Generate SQL property is set to Yes. The Table name is ITEMSUPD, and the Table Action is Replace. Generate drop statement is Yes.

Fail on error (for the drop statement) is No.

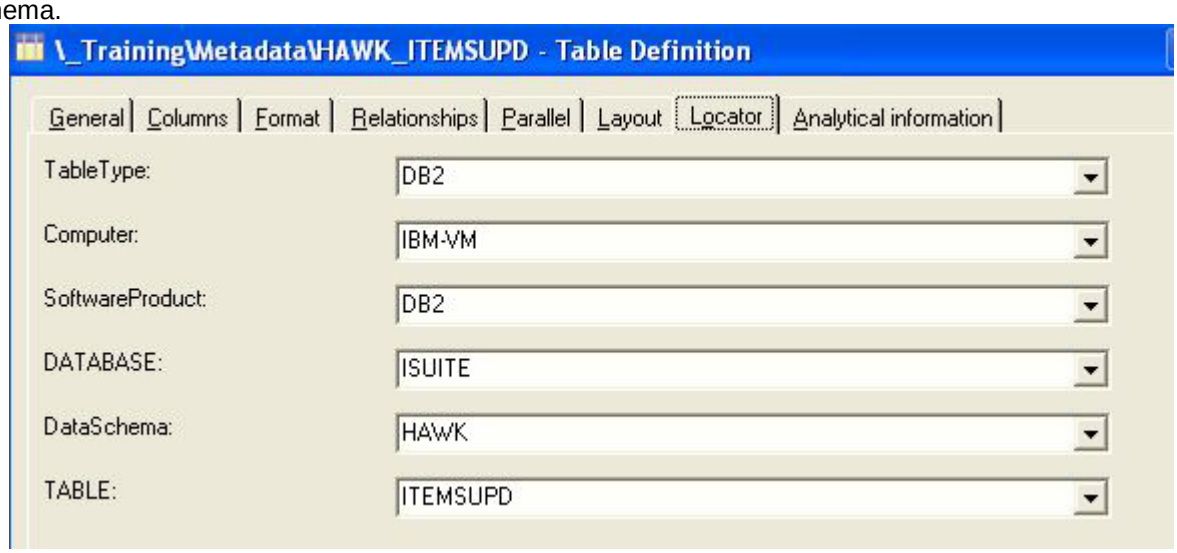


3. Click OK.
4. Compile and run. Check the job log for errors.
5. View the data.

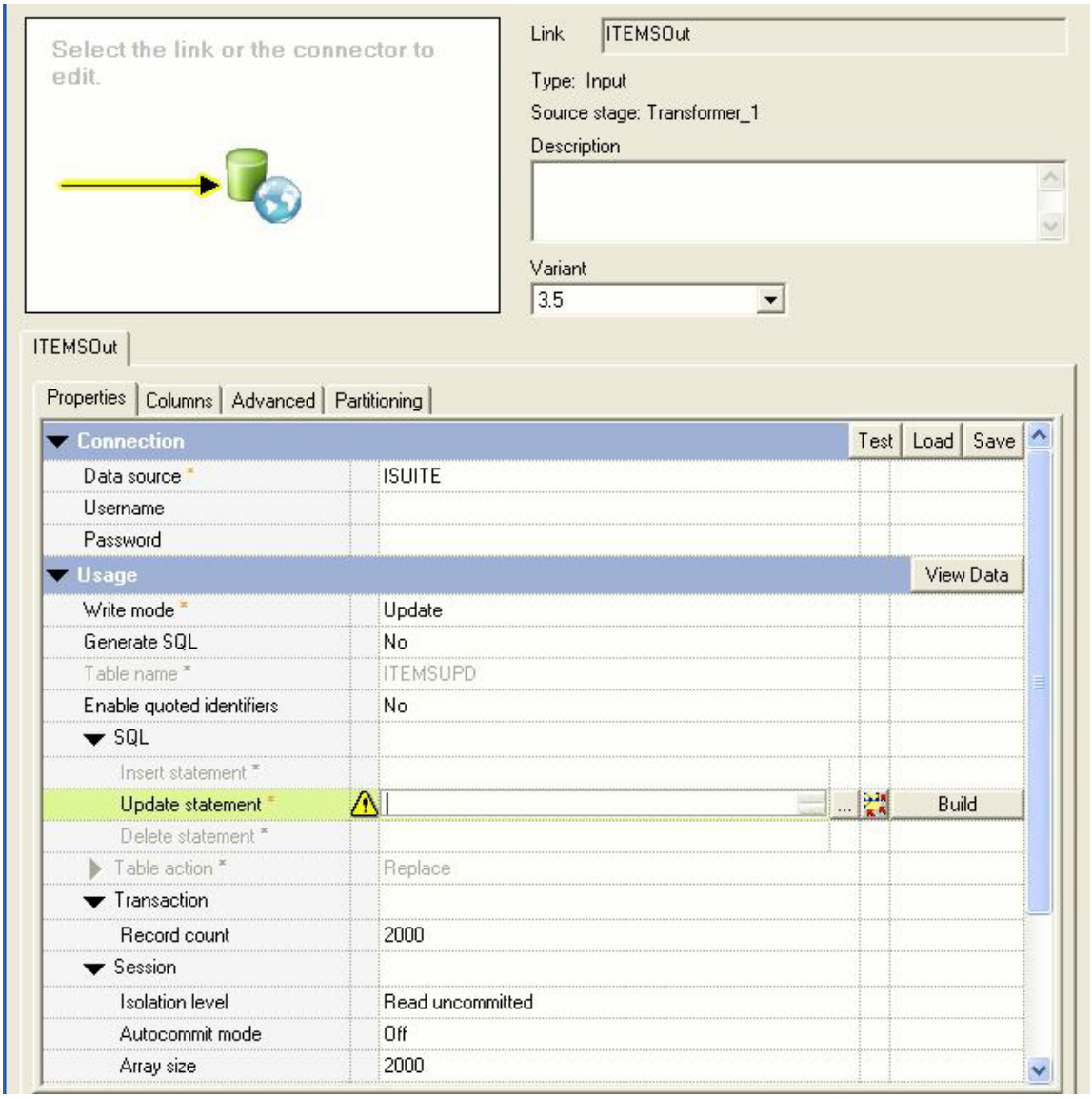
Task: Update a DB2 table

1. Save your job as relWriteTable_ODBC_Update.
2. Import the table definition for your ITEMSUPD table into your _Training>Metadata folder.

3. Open your table definition to the Locator tab, and specify the correct table name and schema.



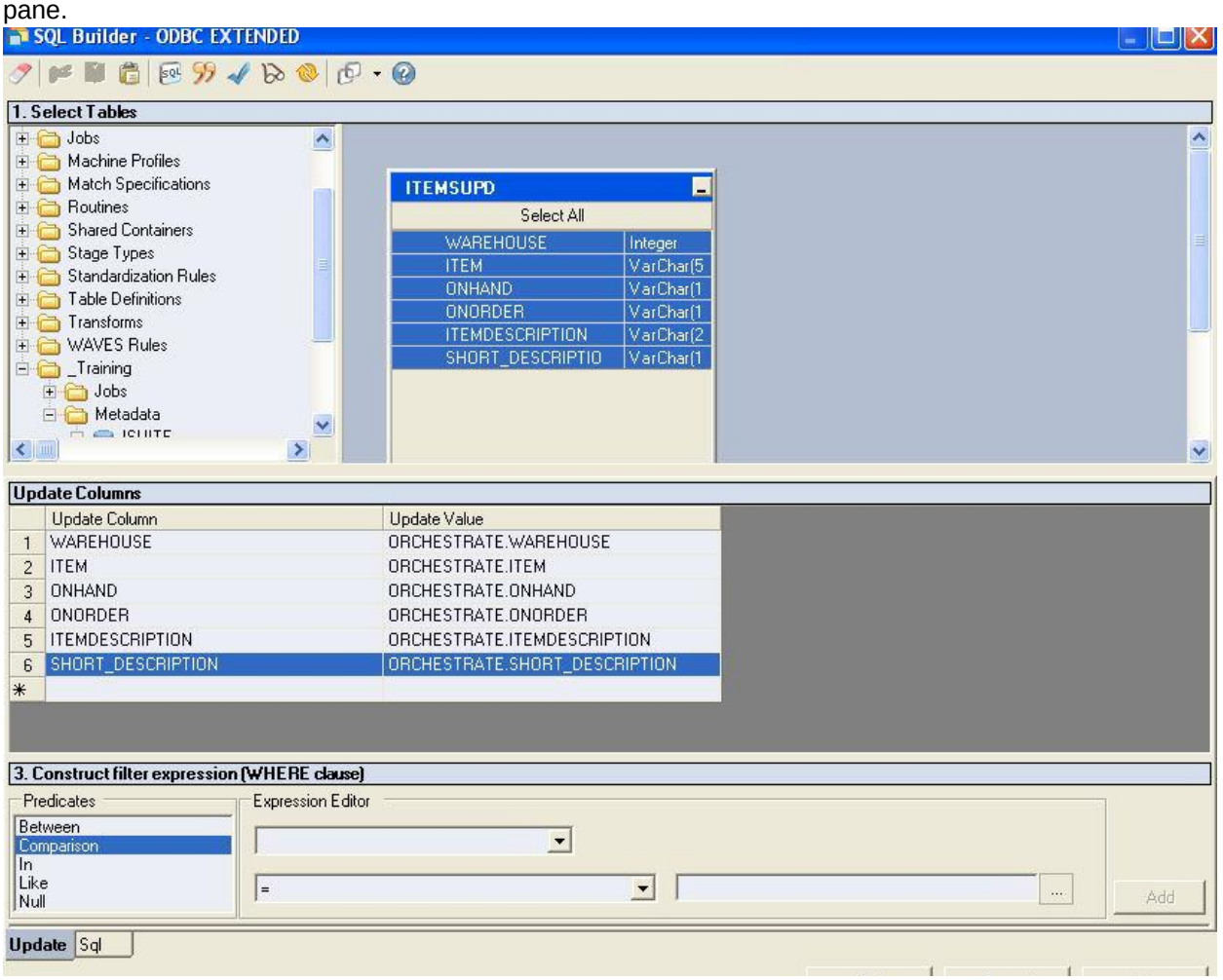
4. Open the target Connector to the Properties tab. Change the Write mode to Update.
   Set
   Generate SQL to No.



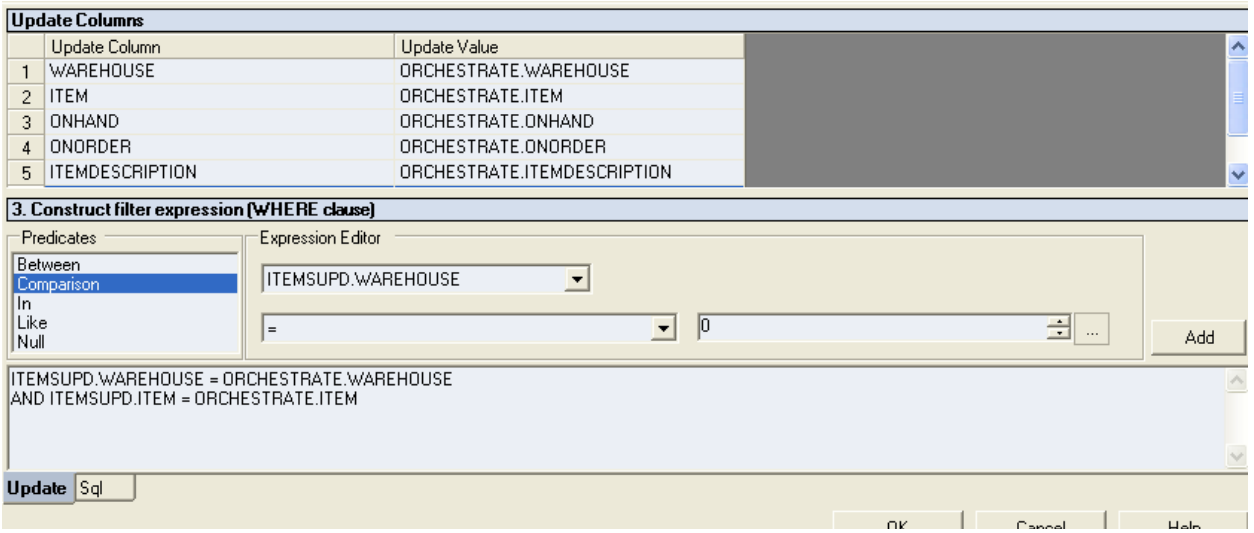
5. Update statement cell Select the and then click Build to open SQL Builder.

6. Drag your ITEMSUPD table definition to the canvas. Select and drag all the columns

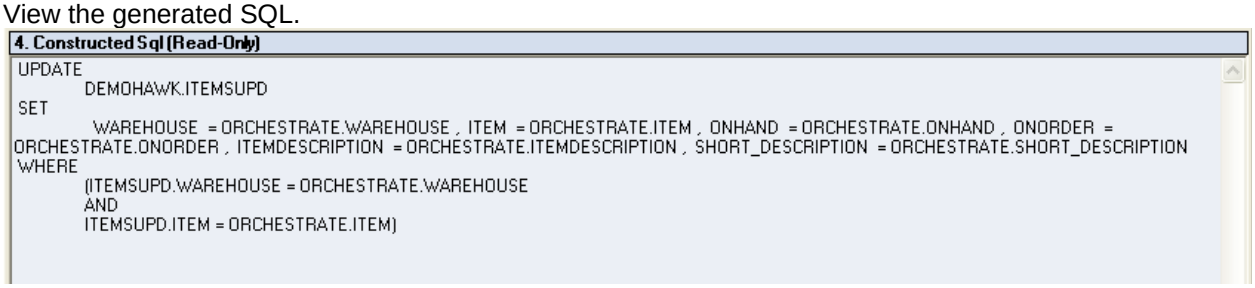
down to the Update columns pane.



7. Define a WHERE clause that updates rows with the same WAREHOUSE and ITEM values. Here, you do equality comparisons between columns in the table and input columns, which you select from the drop-down lists. Click the Add button after each expression is defined.



8. View the generated SQL.



9. Close the SQL Builder and the stage.
10. Compile and run.
11. Open up Director and view the job log.

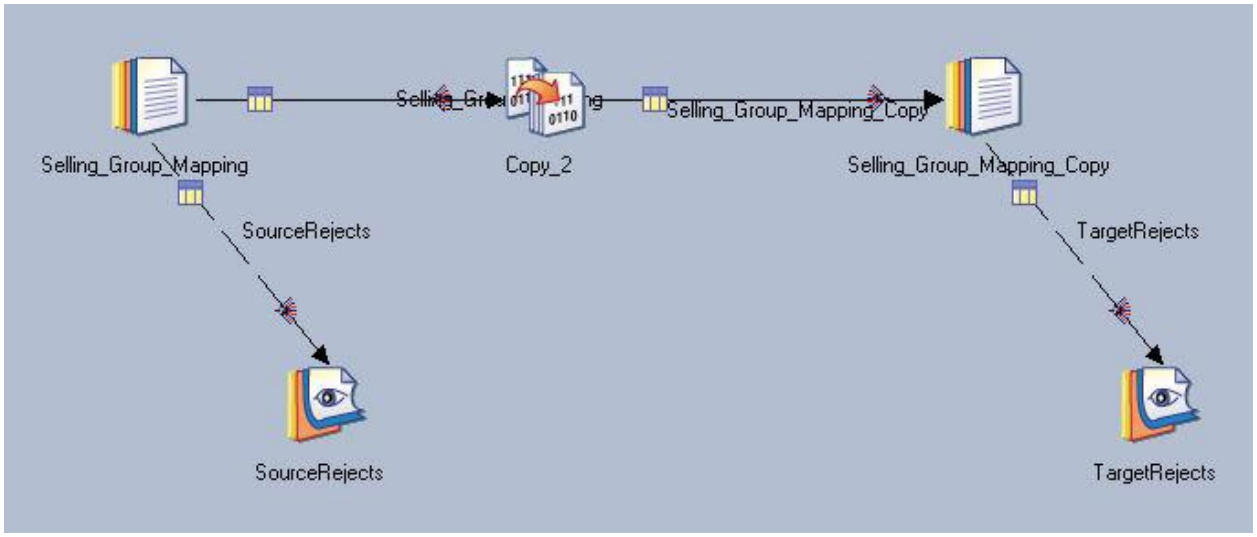
Lab 08: Platform Architecture

Assumptions:

- Completed PPT training session for Lession 6

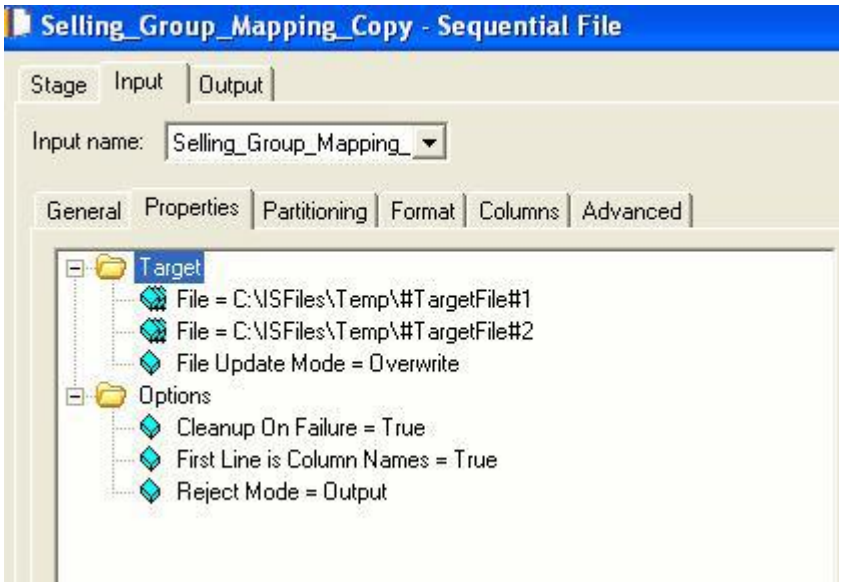
Task: Partitioning and collecting

1. Save your CreateSeqJobParam job as CreateSeqJobPartition.

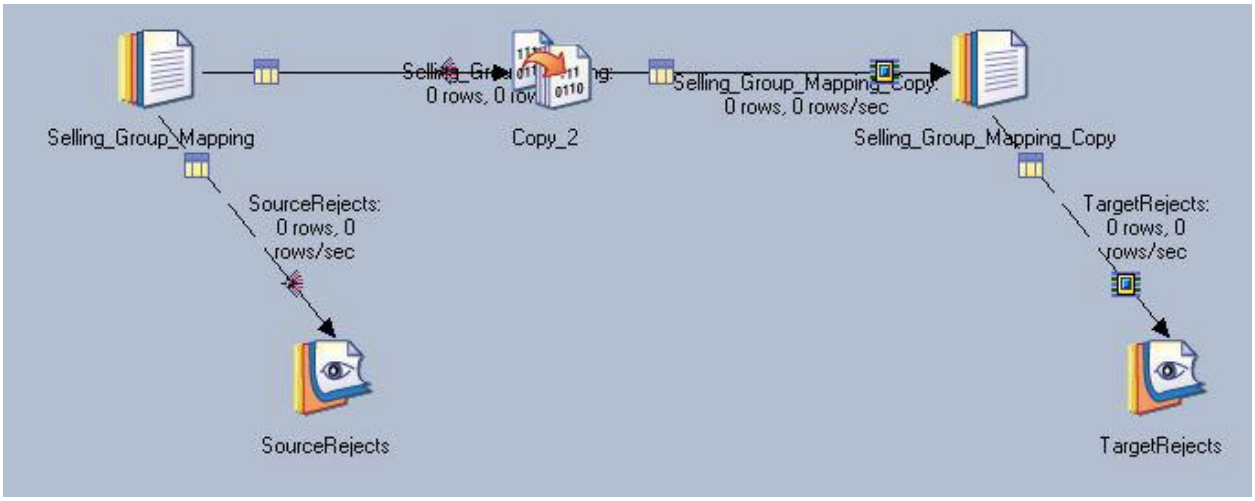


2. Note the icon on the input link to the target stage (fan-in). It indicates that the stage is collecting the data.
3. Open up the target Sequential stage to the Input>Partitioning tab.
4. Note the collecting algorithm (Auto) that's selected.
5. Compile and run your job.
6. View the data.
7. Open up the target Sequential stage to the Properties tab. Instead of writing to one file, write to two files. Make sure they have different names. Create the files in your

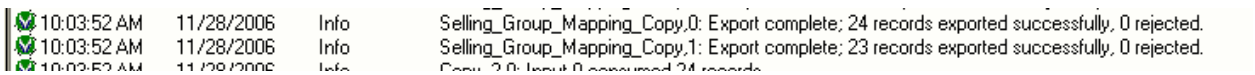
ISFiles>Temp directory.



8. Go to the Partitioning tab. Notice that it is no longer collecting, but now is partitioning. You can see this by noting the words on top of the “Partitioning / Collecting” box. If it says “Partition type:”, then the stage is partitioning. If it says “Collector type:”, it is collecting.
9. Close the stage. Notice that the partitioning icon has changed. It no longer indicates collecting. (Note: If you don’t see this, refresh the canvas by turning “Show link markings” off and on using the toolbar button.) The icon you see now means Auto partitioning.



10. Compile and run your job.
11. View the job log. Notice how the data is exported to the two different partitions (0 and



12. Open each of the two target files in a text file viewer outside of DataStage.


```
150000,SG015 FREEZERS-MILLENNIUM,6,Other
530000,SG053 TRUCKING,6,Other
550000,SG055 LIVE SWINE,6,Other
860000,SG086 TRUCKING,6,Other
870000,SG087 FREEZERS,6,Other
1120000,SG112 N.W.A. SWINE,6,Other
1150000,SG115 BURGER KING,2,Food Service
1190000,SG119 DISTRIBUTION CVP,2,Food Service
1200000,SG120 RETAIL F/P,1,Retail
1210000,SG121 RETAIL FRESH,1,Retail
1230000,SG123 STORE DOOR,2,Food Service
1250000,SG125 FOOD SERVICE,2,Food Service
1360000,SG136 MCDONALDS,2,Food Service
1370000,SG137 EQUITY,6,Other
1380000,SG138 SPECIALTY FOODS,4,Specialty Products
1390000,SG139 MILITARY,1,Retail
1400000,SG140 CLUB STORES,1,Retail
1410000,SG141 INTERNATIONAL,3,International
```

13. Source file:

```
150000,SG015 FREEZERS-MILLENNIUM,6,Other
550000,SG055 LIVE SWINE,6,Other
870000,SG087 FREEZERS,6,Other
1150000,SG115 BURGER KING,2,Food Service
1200000,SG120 RETAIL F/P,1,Retail
1230000,SG123 STORE DOOR,2,Food Service
1360000,SG136 MCDONALDS,2,Food Service
1380000,SG138 SPECIALTY FOODS,4,Specialty Products
1400000,SG140 CLUB STORES,1,Retail
1420000,SG142 GREATER CHINA,3,International
1440000,SG144 JAPAN,3,International
1460000,SG146 R.E.M.A.,3,International
3110000,SG311 INTERNATIONAL SEAFOOD,6,Other
4010000,SG401 KOCH IND/HEFLIN,2,Food Service
4060000,SG406 HOSPITALITY XFER PRICING,6,Other
```

14. TargetFile.txt1:


```
530000,SG053 TRUCKING,6,Other
860000,SG086 TRUCKING,6,Other
1120000,SG112 N.W.A. SWINE,6,Other
1190000,SG119 DISTRIBUTION CVP,2,Food Service
1210000,SG121 RETAIL FRESH,1,Retail
1250000,SG125 FOOD SERVICE,2,Food Service
1370000,SG137 EQUITY,6,Other
1390000,SG139 MILITARY,1,Retail
1410000,SG141 INTERNATIONAL,3,International
1430000,SG143 AMERICAS,3,International
1450000,SG145 KOREA,3,International
3100000,SG310 DOMESTIC SEAFOOD,6,Other
3120000,SG312 COMMODITY PORK,2,Food Service
4030000,SG403 DOMESTIC SEAFOOD,6,Other
4090000,SG409 SURIMI SALES,6,Other
4240000,SG424 COMMODITY POULTRY,2,Food Service
4800000,SG480 DELI SALES,2,Food Service
```

15. TargetFile.txt2:
16. Notice how the data is partitioned. Here, we see that the 1 , 3 , 5 , etc. st go into one file rd th
and the 2n d , 4 , 6 , etc. go in the other file. This is because the default partitioning th th algorithm is Round Robin.
17. Change the partitioning algorithm to, e.g., Entire. Notice how the data gets distributed. Experiment with different partitioning algorithms!

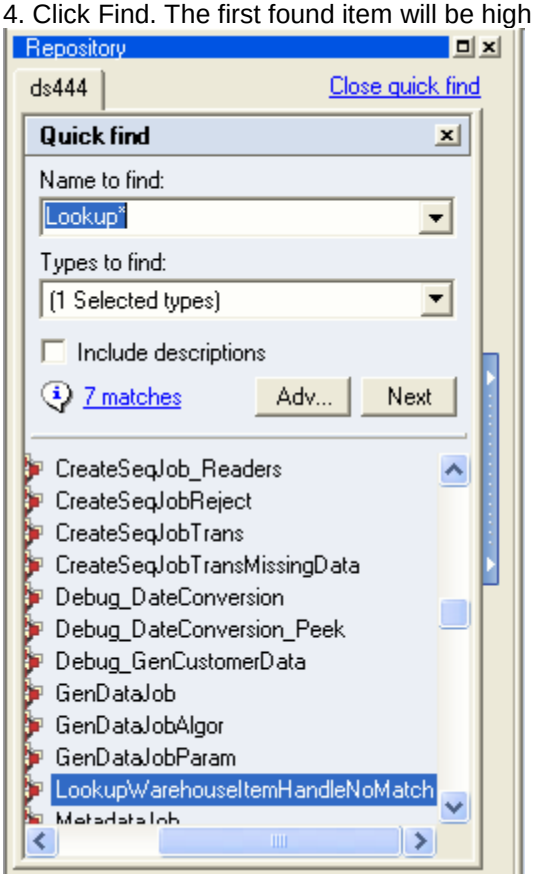
Lab 09: Repository Functions

Assumptions:

- Completed PPT training session for Lession 7.1 to 7.5

Task: Execute a Quick Find

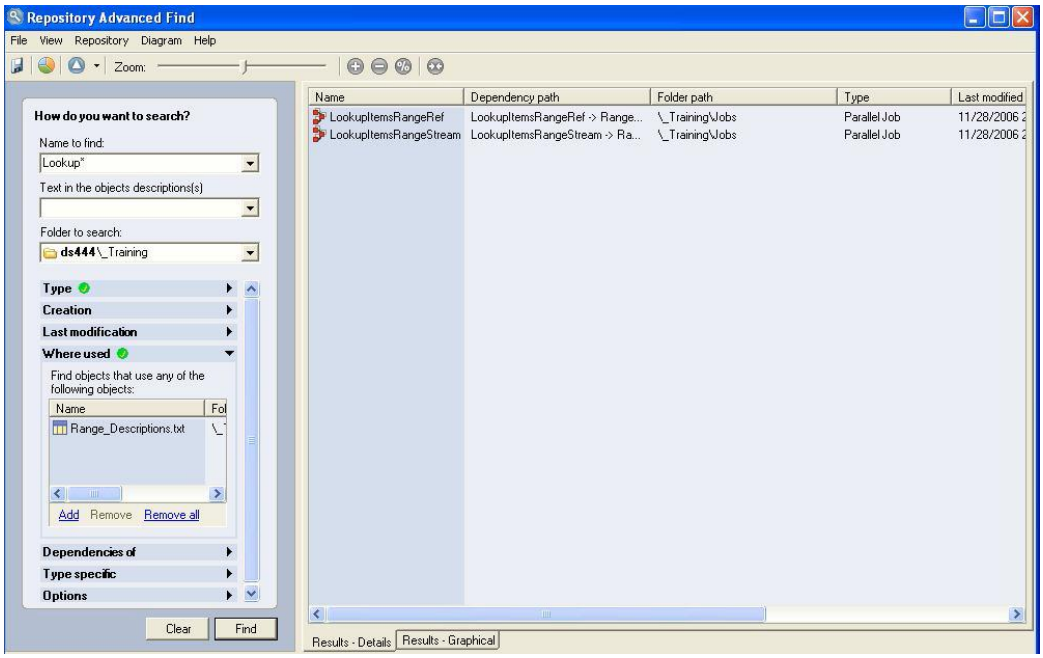
1. Open Quick Find by clicking the link at the top of the Repository window.
2. In the Name to find box type Lookup* and then click the Find button.
3. In the Types to find list select Parallel Jobs.
4. Click Find. The first found item will be highlighted.



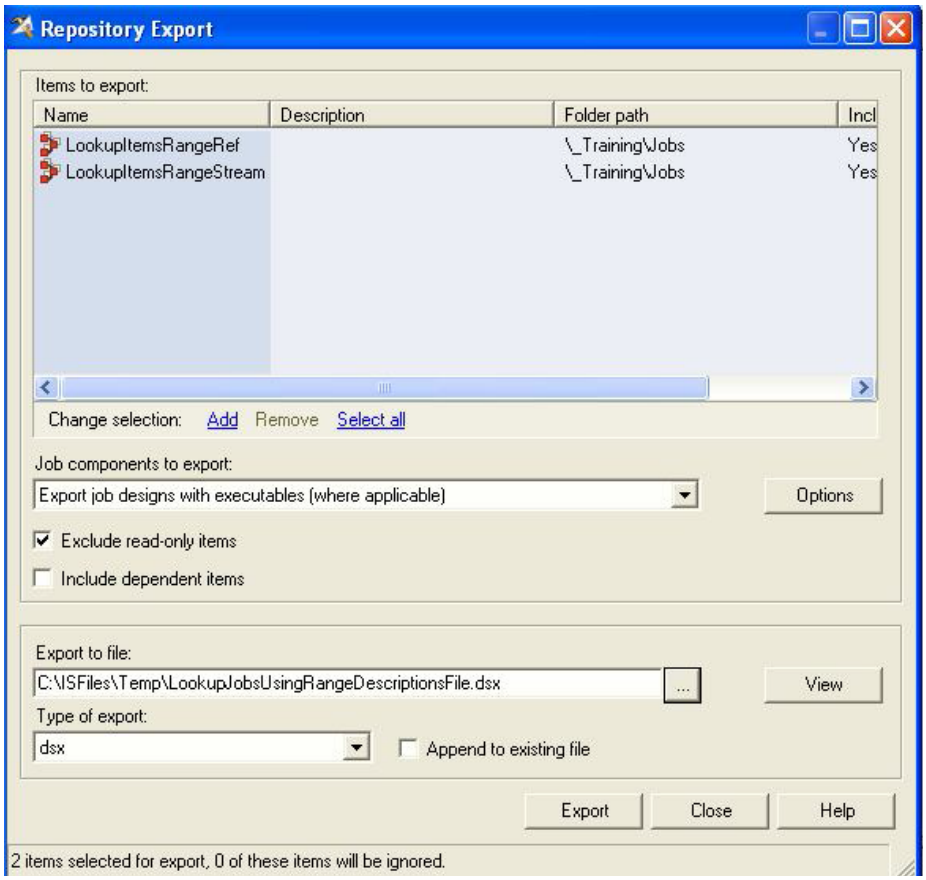
5. Click Next to highlight the next item.

Task: Execute an Advanced Find

1. Click on the link that displays the number of matches. This opens the Advanced Find window and displays the items found so far in the right pane.
2. Change the folder to search to _Training.
3. Open the Last modification folder. Specify objects modified within the last week.
4. Open up the Where Used folder. Add the Range_Descriptions.txt Table Definition.



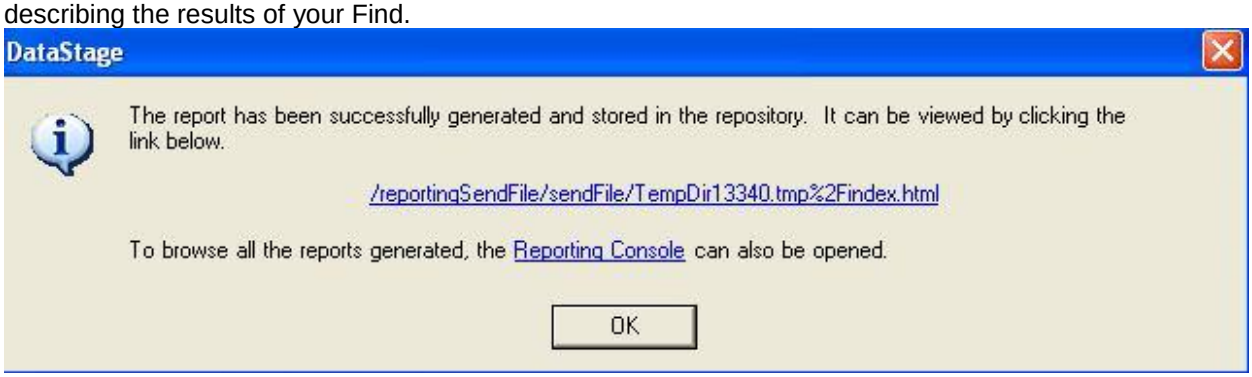
Click Find. This reduces the list of found items to those that use this Table Definition.
5. Open up the Options folder. Specify that the search is to be case sensitive. Click Find.
6. Select the found items and then click the right mouse button over them. Export these jobs to a file named LookupJobsUsingRangeDescriptionsFile.dsx in your Training>Temp folder.



7. Close the Export window.

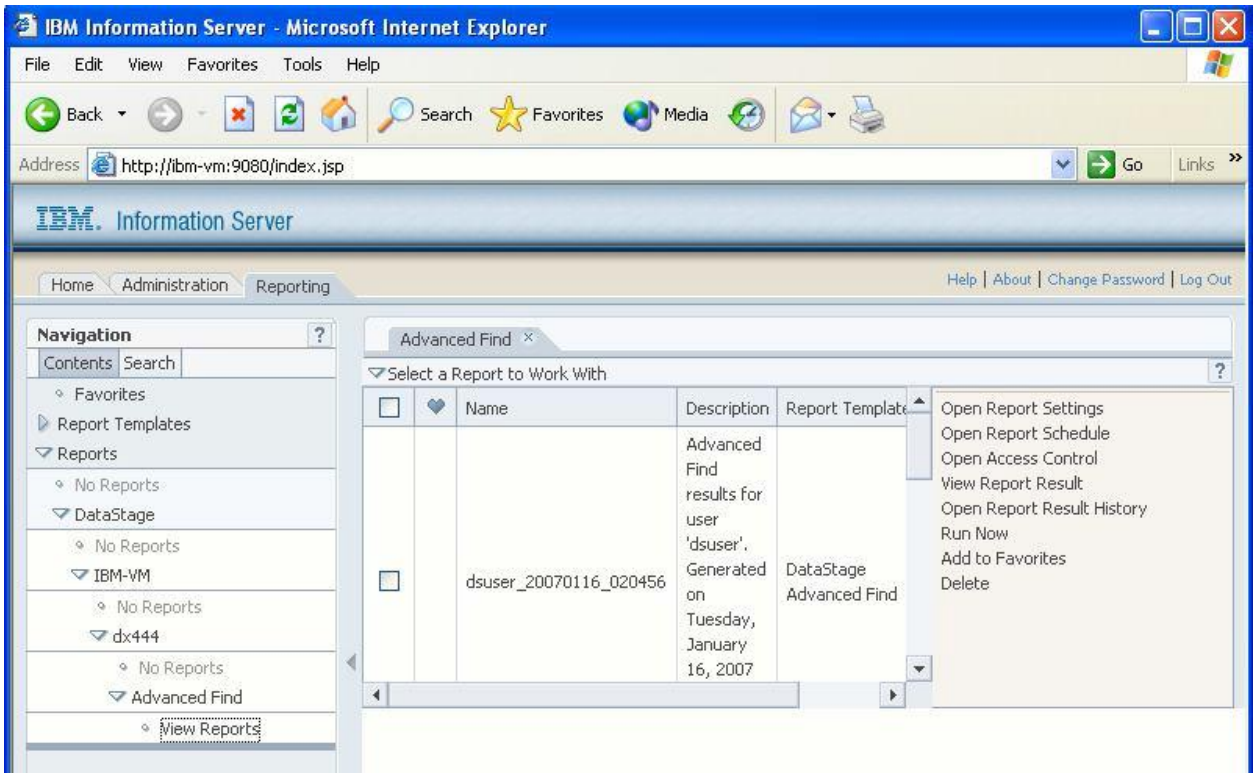
Task: Generate a report

1. Click File>Generate Report to open a window from which you can generate a report describing the results of your Find.



2. Click on the top link to view the report. This report is saved in the Repository where it can be viewed by logging onto the Reporting Console.

3. After closing this window, click on the Reporting Console link. On the Reporting tab, expand the Reports folder as shown.

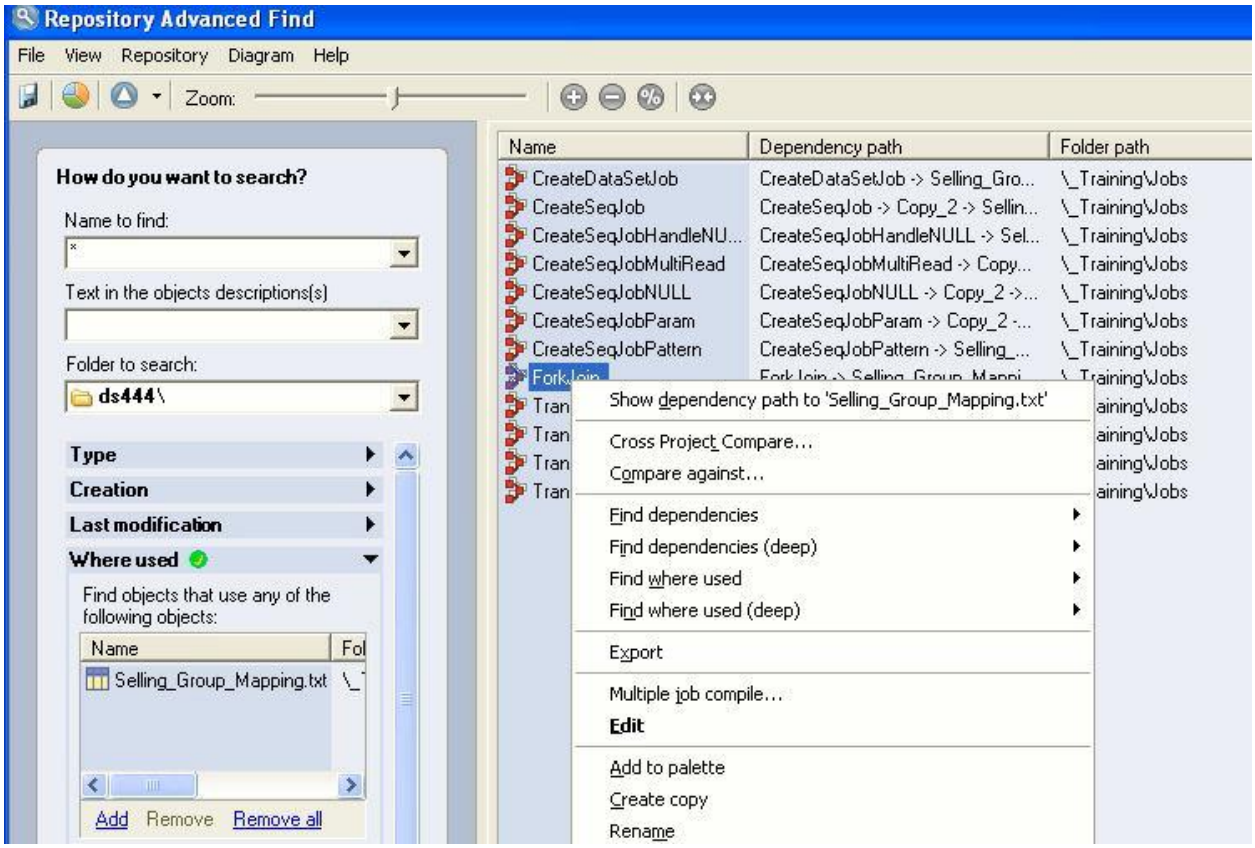


4. Select your report and then click View Report Result. This displays the report you viewed earlier from Designer. By default, a Suite user only has permission to view the report. A Suite administrator can give additional administrative functions to a Suite user, including the ability to alter report properties, such as format.

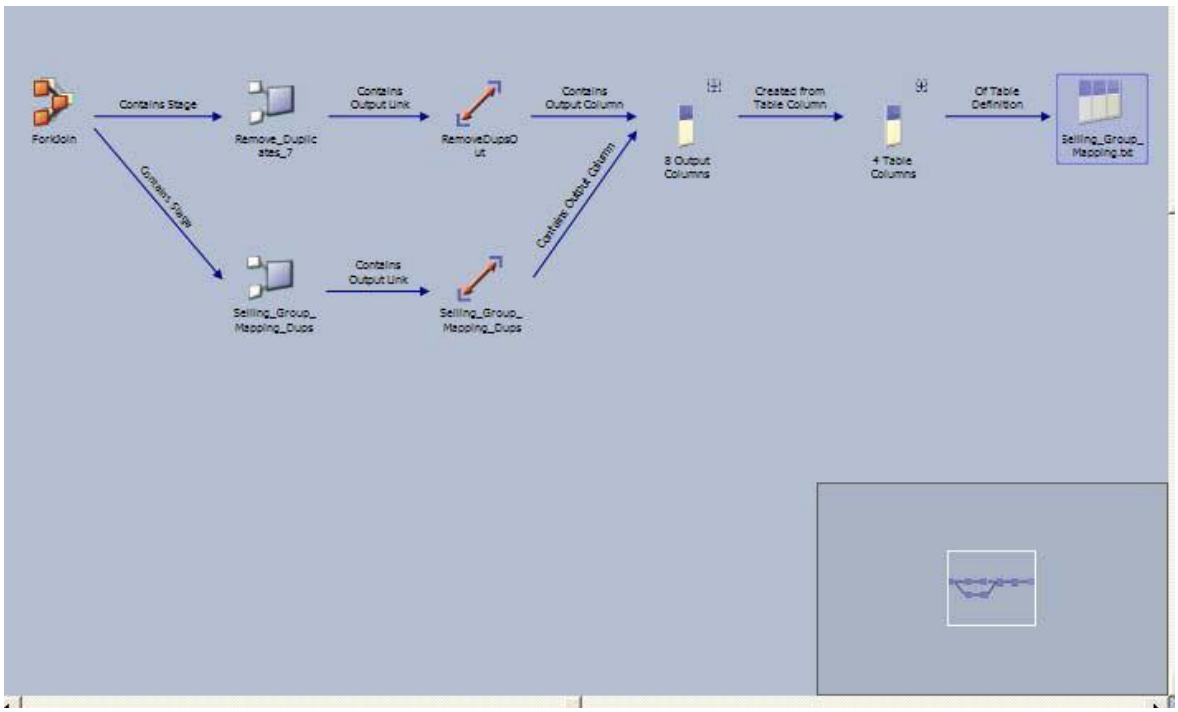
Impact AnalysiS

Task: Perform an impact analysis

1. In the Repository window, select your Selling_Group_Mapping.txt Table Definition. Click the right mouse button and then select Find Where Used>All Types.
2. Click the right mouse button over the ForkJoin job listed and then click “Show dependency path to…”



3. Use the Zoom button to adjust the size of the dependency path so that it fits into the window.



4. Hold right mouse button over a graphical object and move the path around.
5. Notice the “birds-eye” view box in the lower right hand corner. This shows how the path is situated on the canvas. You can move the path around by clicking to one side of the image in

the birds-eye view window and by holding the right mouse button down over the image and moving the image around.

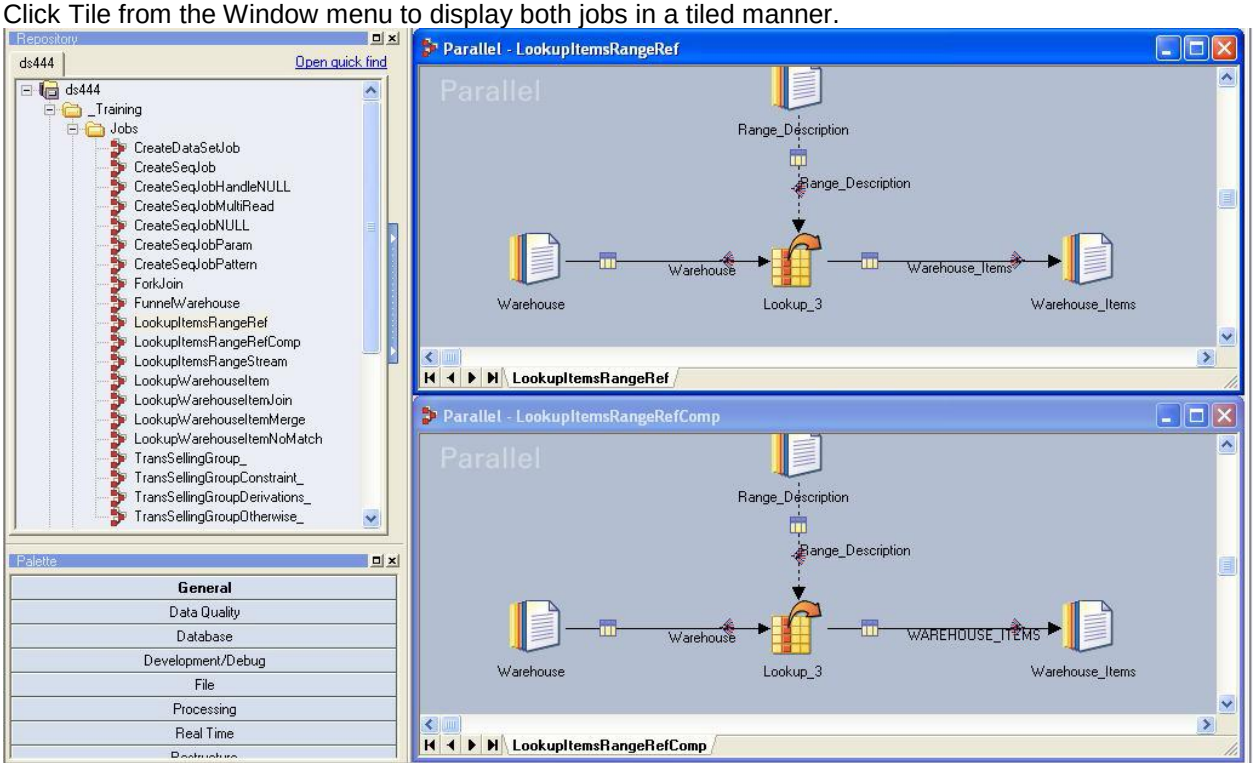
Job and Table Difference Reports

Assumptions:

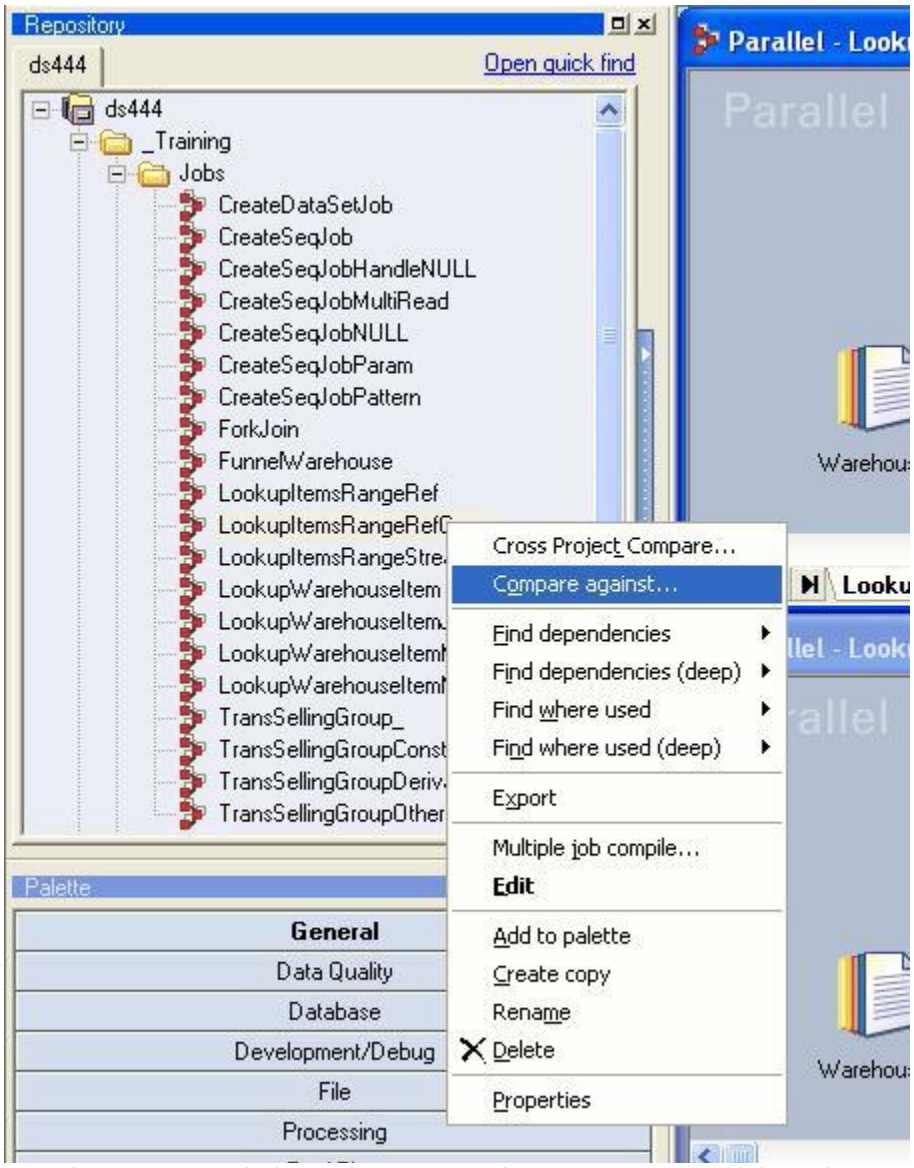
• You have created the LookupItemsRangeRef job in a previous exercise
• You have created the Warehouse.txt Table Definition in a previous exercise

Task: Find the differences between two jobs

1. Open your LookupItemsRangeRef job. Save it as LookupItemsRangeRefComp into your _Training>Jobs folder.
2. Make the following changes to the LookupItemsRangeRefComp job.
3. Open up the RangeDescription Sequential File stage on the reference link. On the Columns tab, change the length of the first column (StartItem) to 111. On the Properties tab, change the First Line is Column Names to False.
4. Change the name of the link going to the Warehouse_Items target Sequential File stage to WAREHOUSE_ITEMS.
5. Open the Lookup stage. In the constraints window, change the Lookup Failure condition to “Drop”.
6. Save the changes to your job.
7. Open up both the LookupItemsRangeRef and the LookupItemsRangeRefComp jobs. Click Tile from the Window menu to display both jobs in a tiled manner.



8. Right-click over your LookupItemsRangeRefComp job name in the Repository window and then select Compare Against.



9. In the Compare window select your LookupItemsRangeRef job on the Item Selection tab.

10. Click OK to display the Comparison Results window.

11. Click on a stage or link in the report, e.g., Range_Description. Notice that the stage is highlighted in both of the jobs.
12. Click on one of the underlined words. Notice that the editor is opened for the referenced item.
13. With the Comparison Results window the active window, click File>Save as. This saves your report as an html file.
14. Open up the html file in a Browser to see what it looks like.

Task: Find the differences between two Table Definitions

1. Create a copy of your Warehouse.txt Table Definition.
2. Make the following changes to the *copy* .
3. On the Columns tab change the name of the Item column to ITEM_ZZZ. And change its type and length to Char(33).
4. On the General tab, change the short description to your name.
5. Click OK.
6. Right-click over your Table Definition copy and then select Compare Against.
7. In the Comparison window select your Warehouse.txt Table.
8. Click OK to display the Comparison Results window.

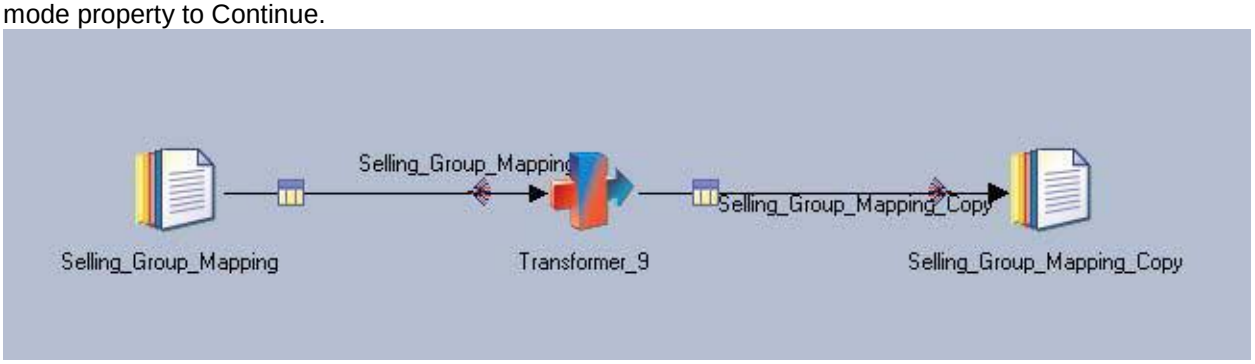
Lab 10: Metadata in the Parallel Framework

Assumptions:

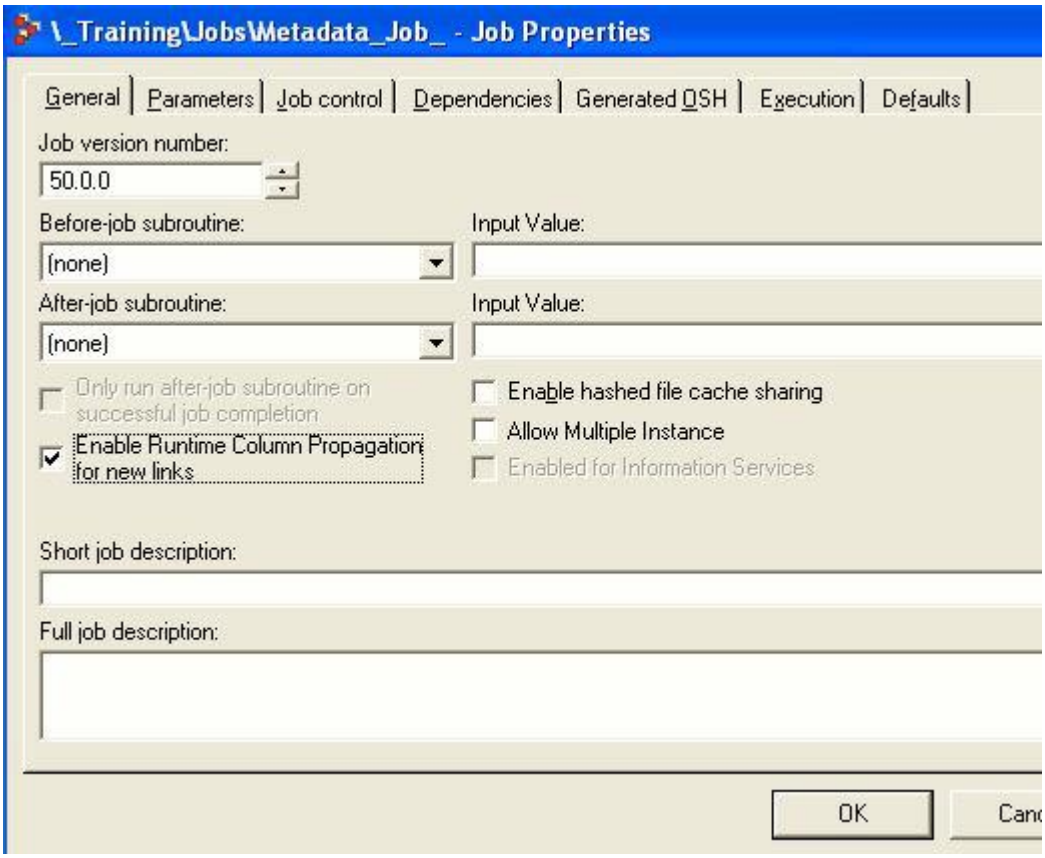
- You have a working TransSellingGroup job.
- Completed PPT training session for Lession 7.1 to 7.5

Task: Use a schema in a Sequential stage

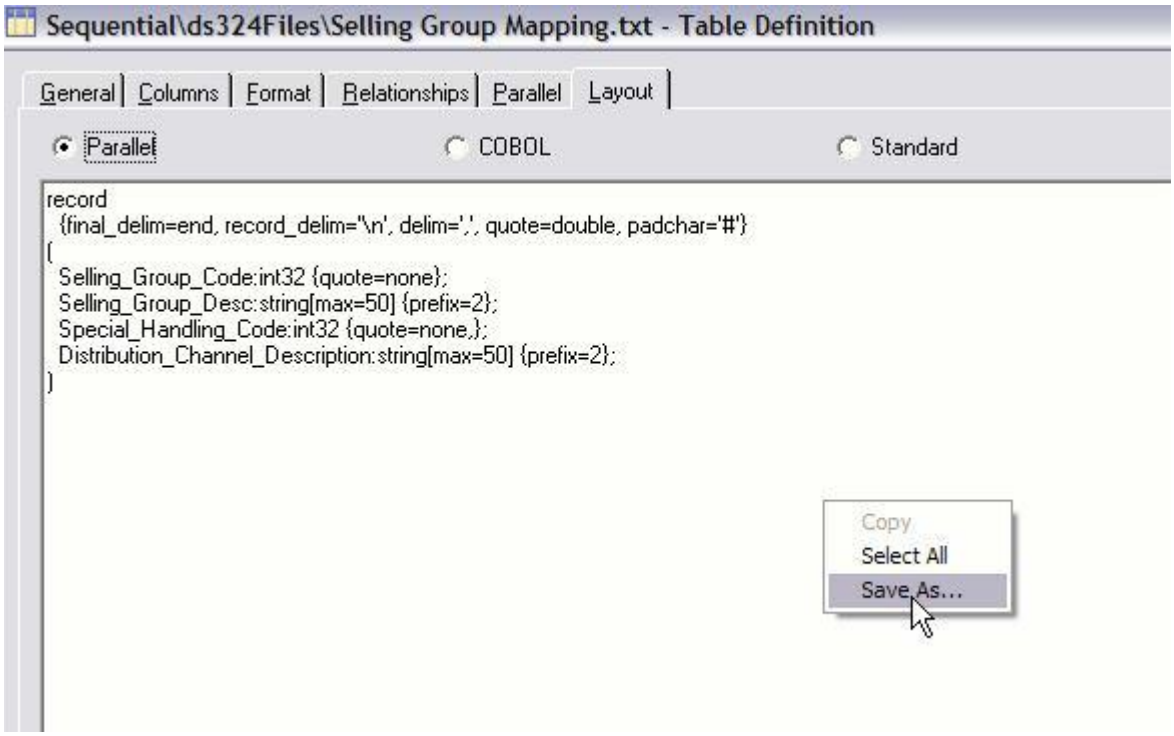
1. Log on to Administrator. On the Projects tab, select your project and then click Properties. Enable RCP (Runtime Column Propogation) for your project or verify that it is enabled.
2. Open your TransSellingGroup job and save it as Metadata_job. Remove the two reject links. Open the two Sequential Files stages and change the value of the Reject mode property to Continue.



3. Open up the Job Properties window and enable RCP for all links of your job.



4. In the Repository window, locate the Selling_Group_Mapping.txt Table Definition that was loaded into the source. Double-Click to open the Table definition.
5. On the Layout tab, select the Parallel button to display the OSH schema. Click the right mouse button to save this as a file in your ISFiles directory. Name the file Selling_Group_Mapping.schema.
6.



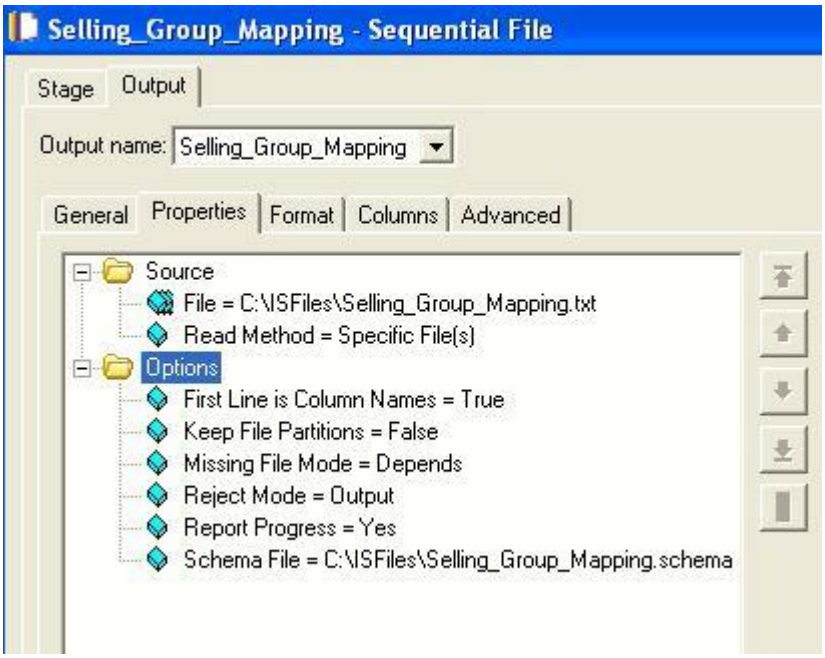
7. Open up the schema file in WordPad to view its contents.

```
record
  {final_delim=end, record_delim='\n', delim=',', quote=double, padchar='#'}
(
  Selling_Group_Code:int32 {quote=none};
  Selling_Group_Desc:string[max=50] {prefix=2};
  Special_Handling_Code:int32 {quote=none,};
  Distribution_Channel_Description:string[max=50] {prefix=2};
)
```

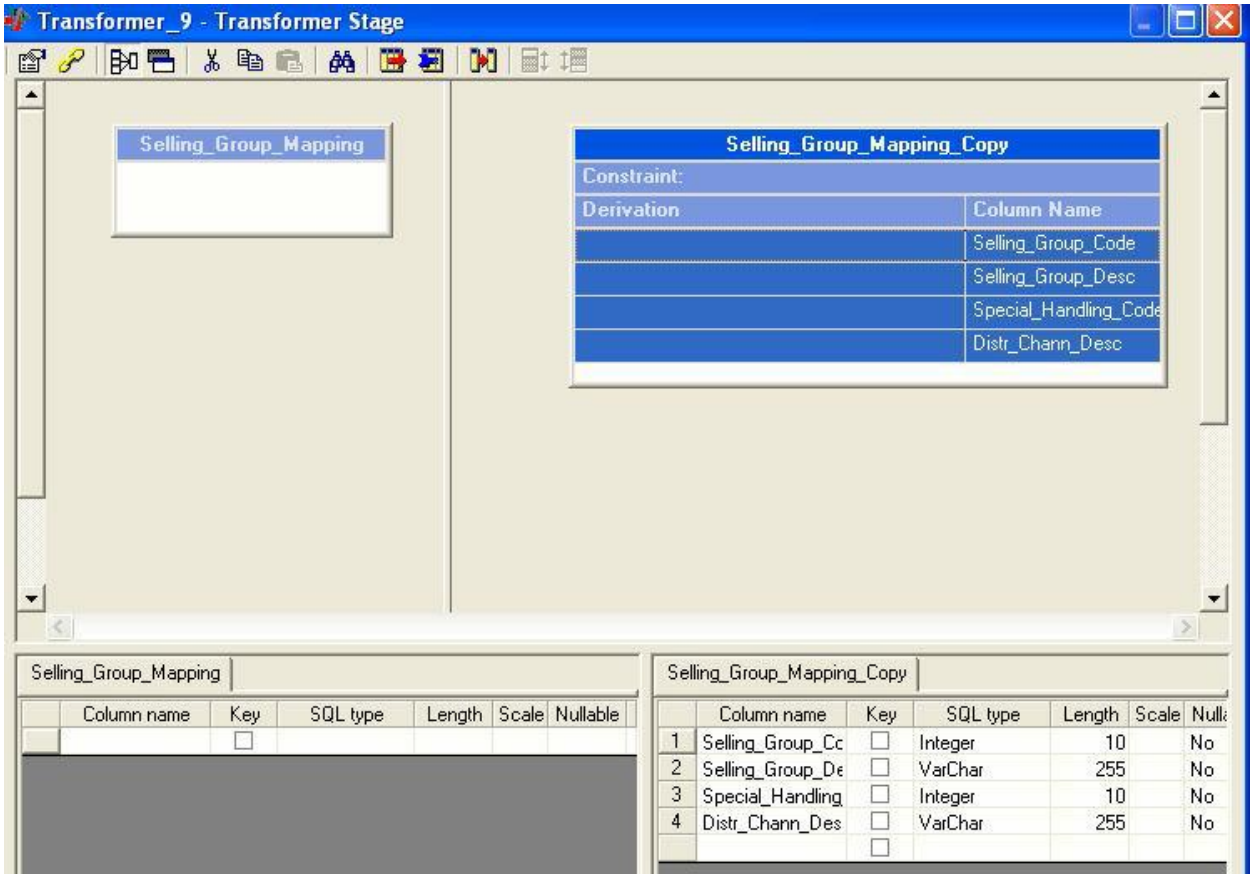
8. Important: The “{prefix=2}” extended properties you see in the schema file are definitely something we don’t want. Remove them.

```
record
  {final_delim=end, record_delim='\n', delim=',', quote=double, padchar='#'}
(
  Selling_Group_Code:int32 {quote=none};
  Selling_Group_Desc:string[max=50];
  Special_Handling_Code:int32 {quote=none,};
  Distribution_Channel_Description:string[max=50];
)
```

9. Open up your Source Sequential stage to the Properties tab. Add the Schema file option. Then select the schema file you copied to the ISFiles directory in the previous step.



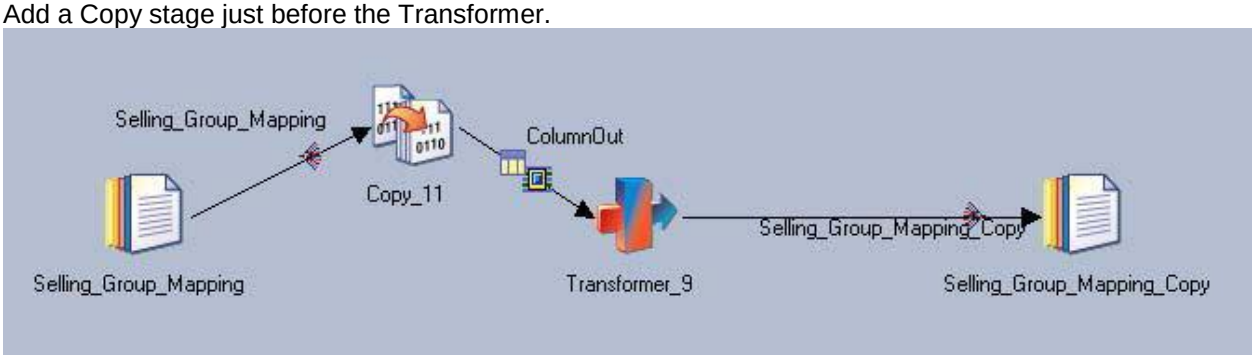
10. On the Columns tab, remove all the columns.
11. In the Transformer, clear all column derivations (don't delete the output columns!) going into the target columns and verify that RCP is enabled. Also remove any constraints, if any are defined. If you don't remove the constraints, the job won't compile, because the constraint references an unknown input column.



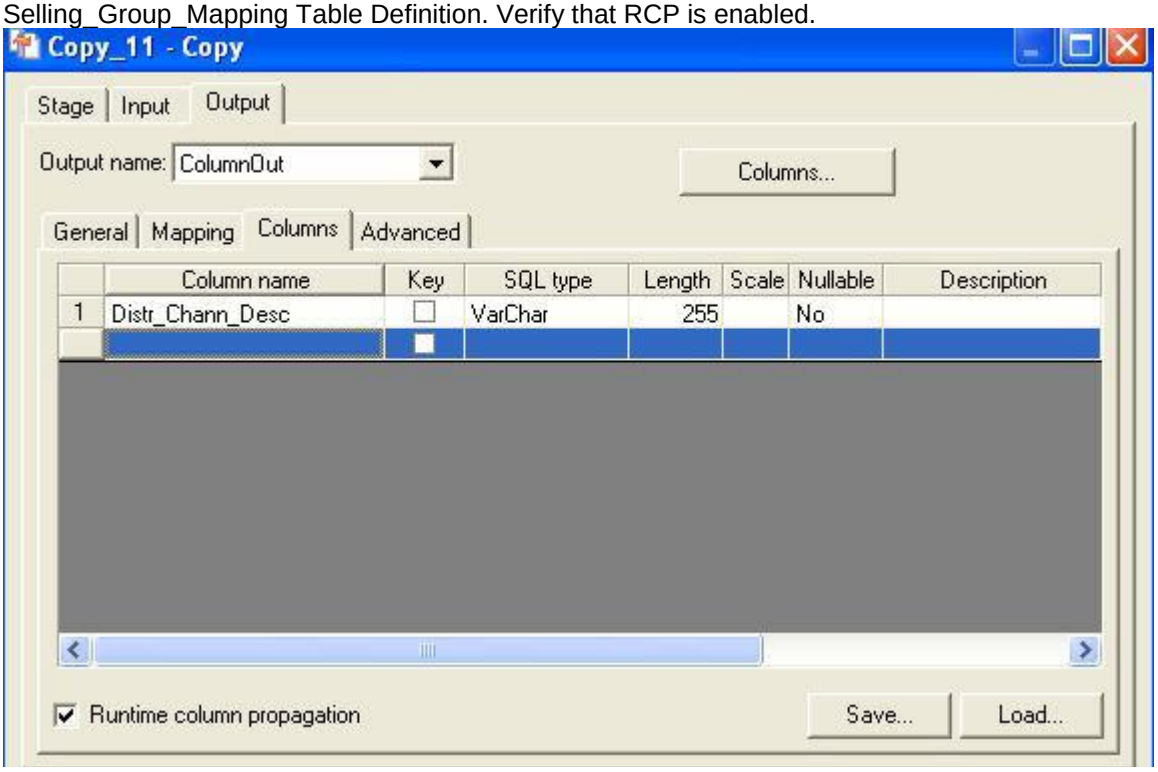
12. Compile and run your job. Verify that the data is written correctly.

Task: Define a derivation in the Transformer

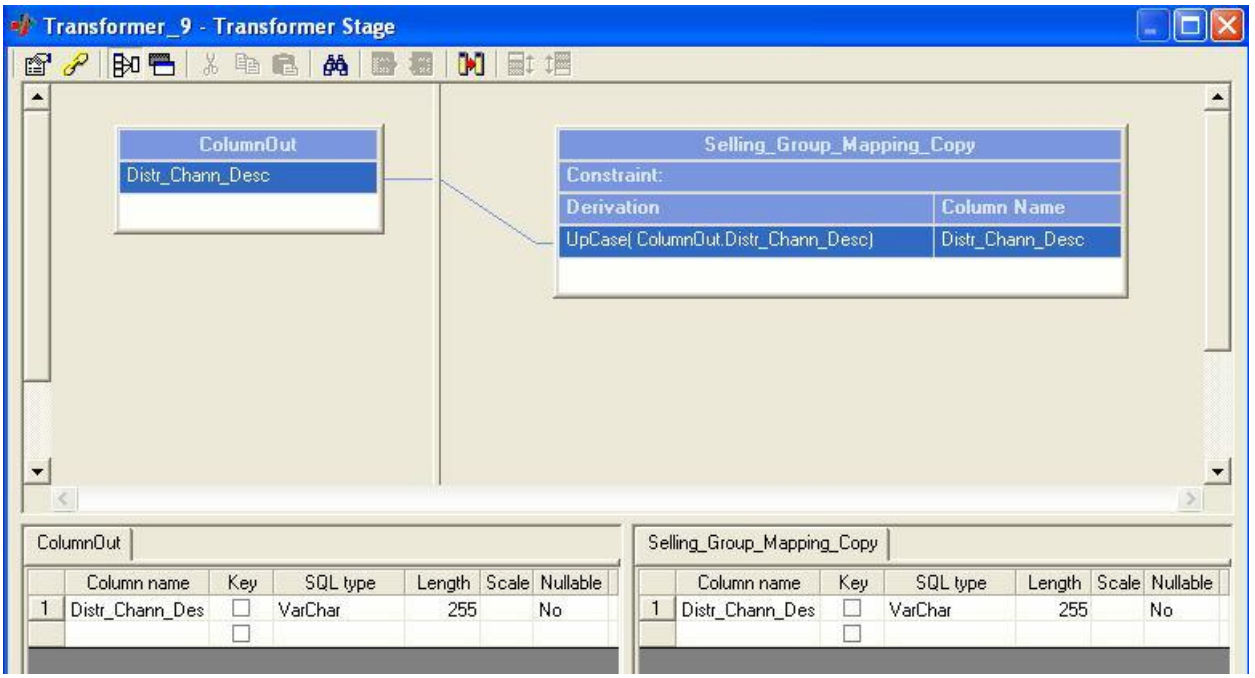
1. Save your job as Metadata_job_02.
2. Open the target Sequential File stage. Remove all the columns. Add the optional Schema File property and select your schema file for it.
3. Add a Copy stage just before the Transformer.



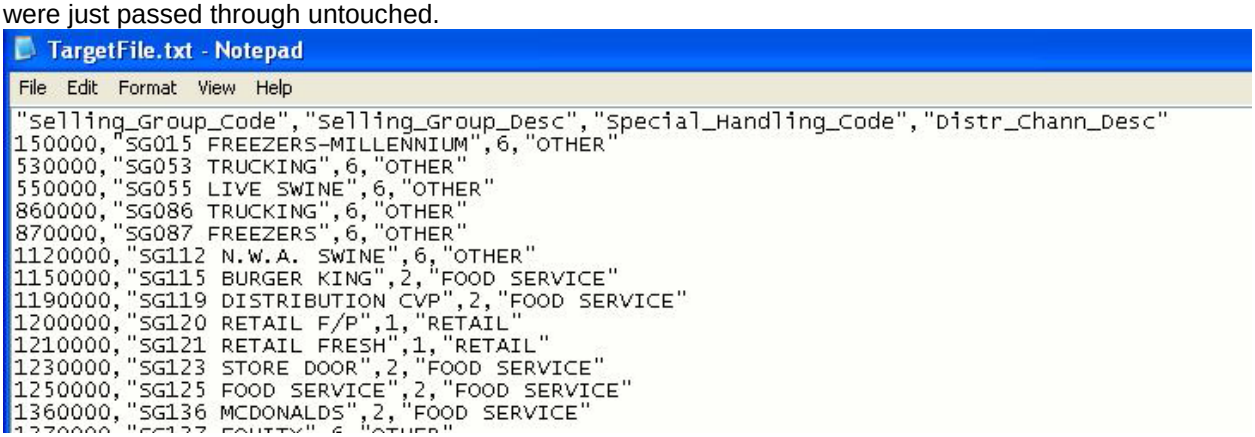
4. On the Columns tab of the Copy stage, load just the Distr_Chann_Descfield from the Selling_Group_Mapping Table Definition. Verify that RCP is enabled.



5. Open the Transformer. Map the Distr_Chann_Desc column across the Transformer. Define a derivation for the output column that turns the input column to uppercase.

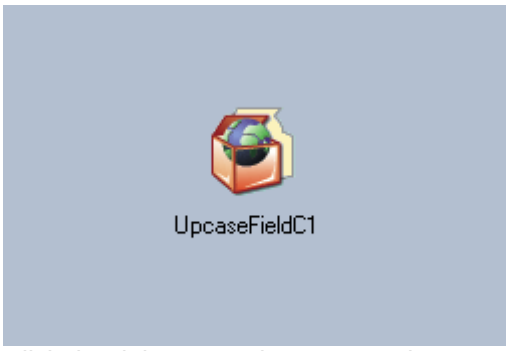


6. Compile and run your job.
7. View the data in the file (not using DataStage View Data). Notice that the Distr_Chann_Desc column data has been turned to uppercase. All other columns were just passed through untouched.

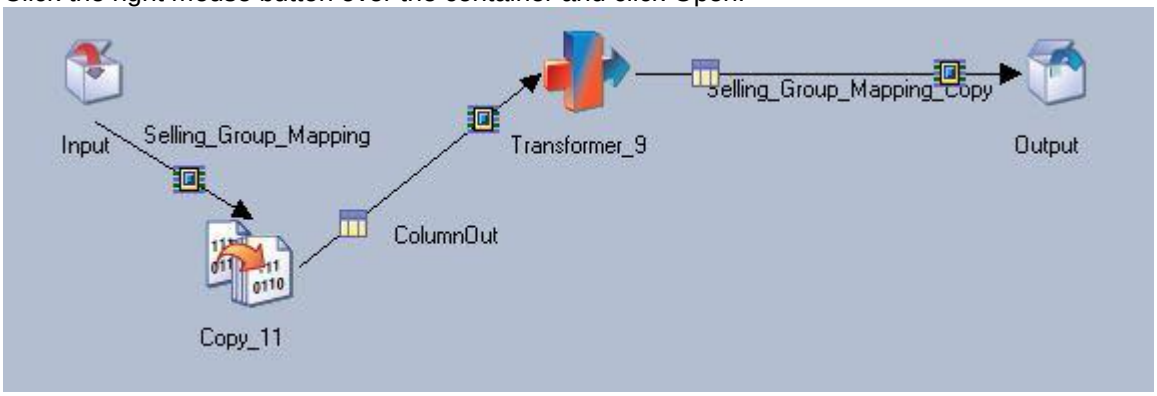


Task: Create a Shared Container

1. Highlight the Copy and Transformer stages of your job. Click Edit>Construct Container>Shared. Save your container, named UpcaseField, into your _Training>Jobs folder.
2. Close your job without saving it. ***NOTE: Don't save your job! It was just used to create the container.***
3. Create a new parallel job named Metadata_Shared_Container.
4. Drag your shared container to the canvas. This creates a *reference* to the shared container, meaning that changes to the shared container will automatically apply to any job that uses it.

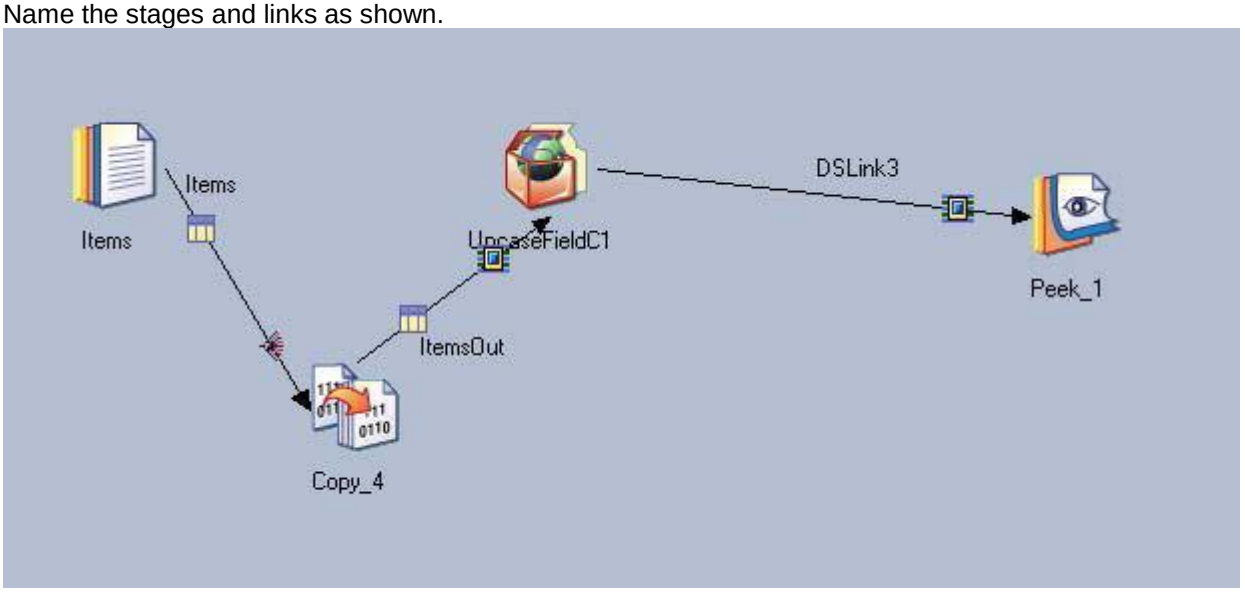


5. Click the right mouse button over the container and click Open.

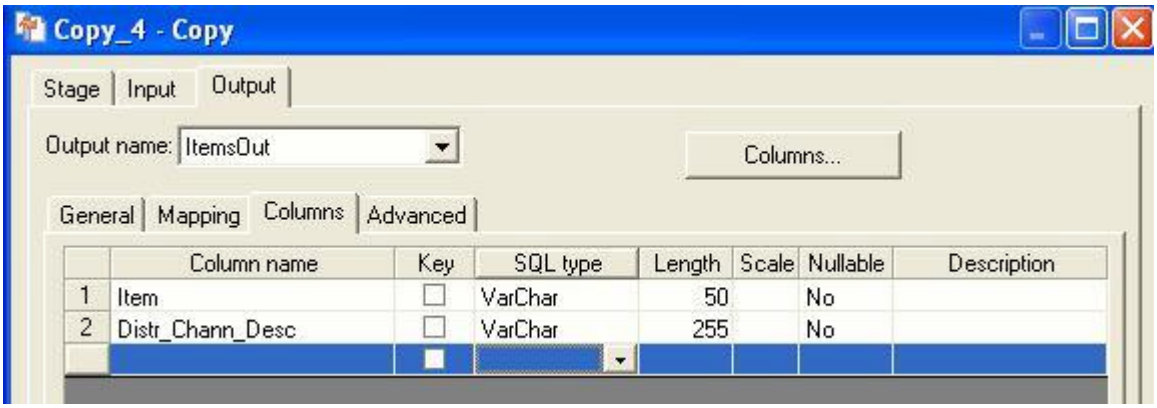


6. Open up the Transformer and note that it applies the Upcase function to a column named Distr_Chann_Desc. Close the Transformer and Container job without saving it.

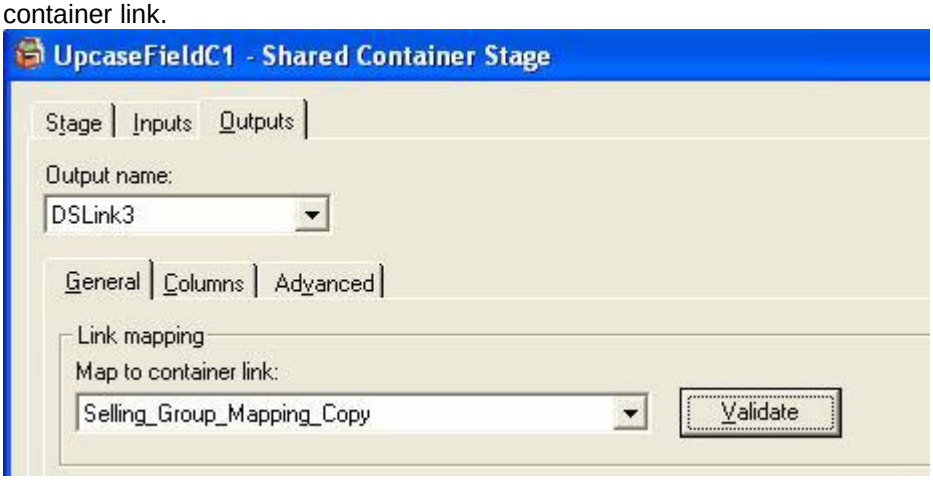
7. Add a source Sequential File stage, Copy stage, and a target Peek stage as shown. Name the stages and links as shown.



8. Edit the Items Sequential stage to read from the Items.txt sequential file. You should already have a Table Definition, but if you don't you can always create one.
9. Verify that you can view the data.
10. In the Copy stage, move all columns through. On the Columns tab, change the name of the second column to Distr_Chann_Desc so that it matches the column in the Shared Container Transformer that the Upcase function is applied to.



11. On the Outputs tab, map the output link to the Selling_Group_Mapping_Copy container link.



12. Compile and run your job.
13. Open up the Director log and find the Peek messages. Verify that the second column of data has been changed to uppercase.

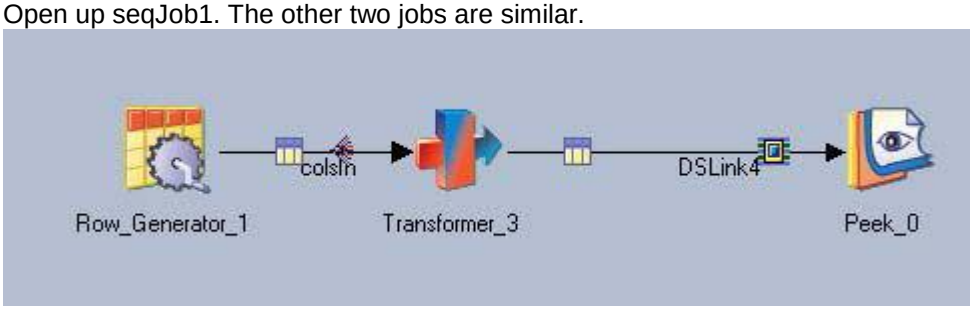
Lab 11: Job Control

In this exercise, you create a single job sequence that executes three jobs.

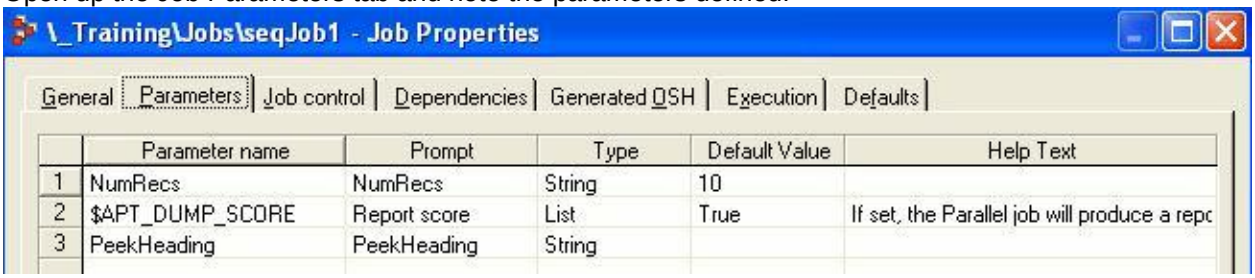
Assumptions:

- 3 working job. Assume names seqJob1, seqJob2, seqJob3
- Completed PPT training session for Lession 7

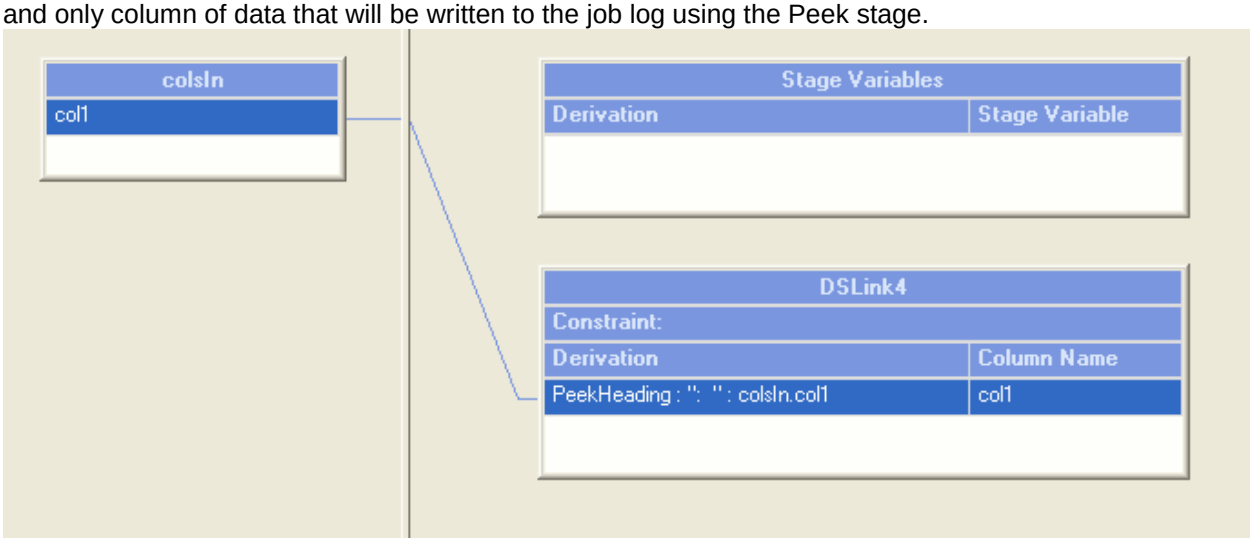
1. Open up seqJob1. The other two jobs are similar.



2. Open up the Job Parameters tab and note the parameters defined.

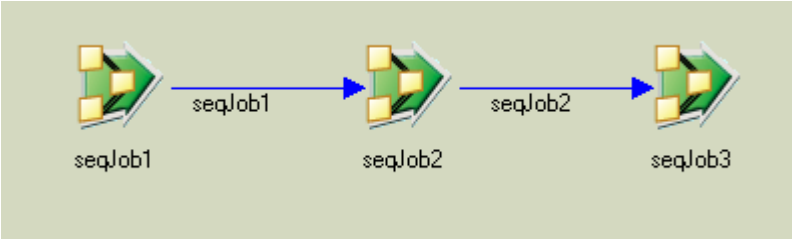


3. Open the Transformer. Notice that the job parameter PeekHeading prefixes the one and only column of data that will be written to the job log using the Peek stage.

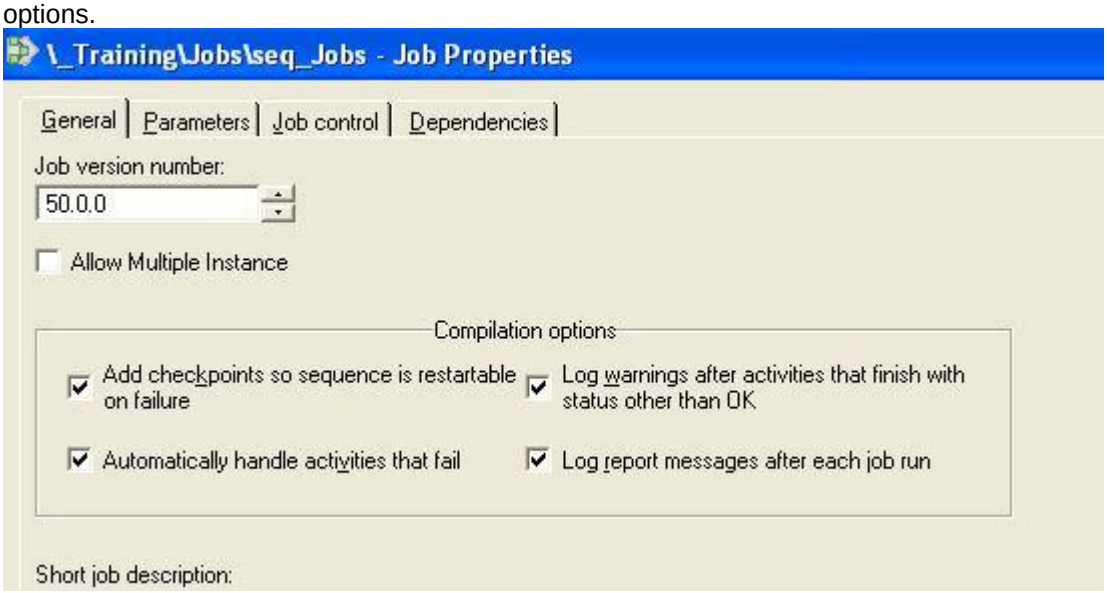


4. Open a new Job Sequence and save it as seq_Jobs.

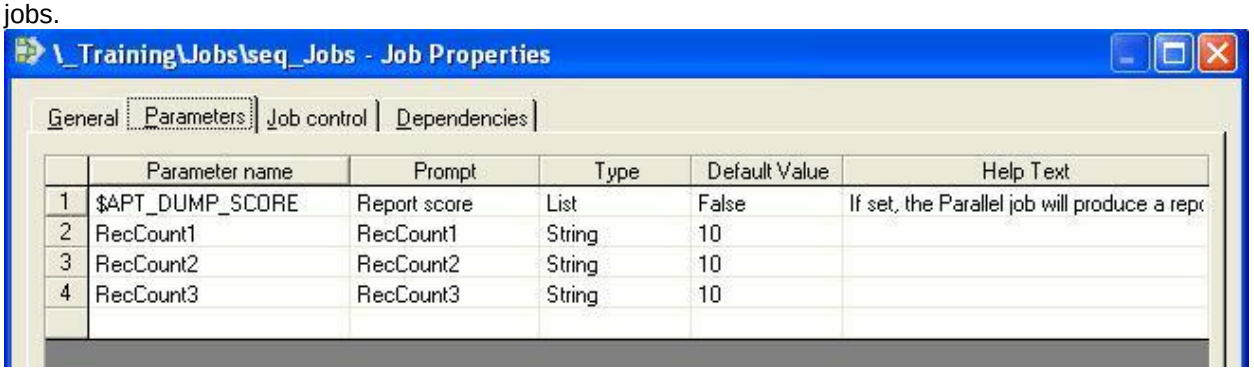
5. Drag your 3 jobs to the canvas, link them, and name the stages and links as shown



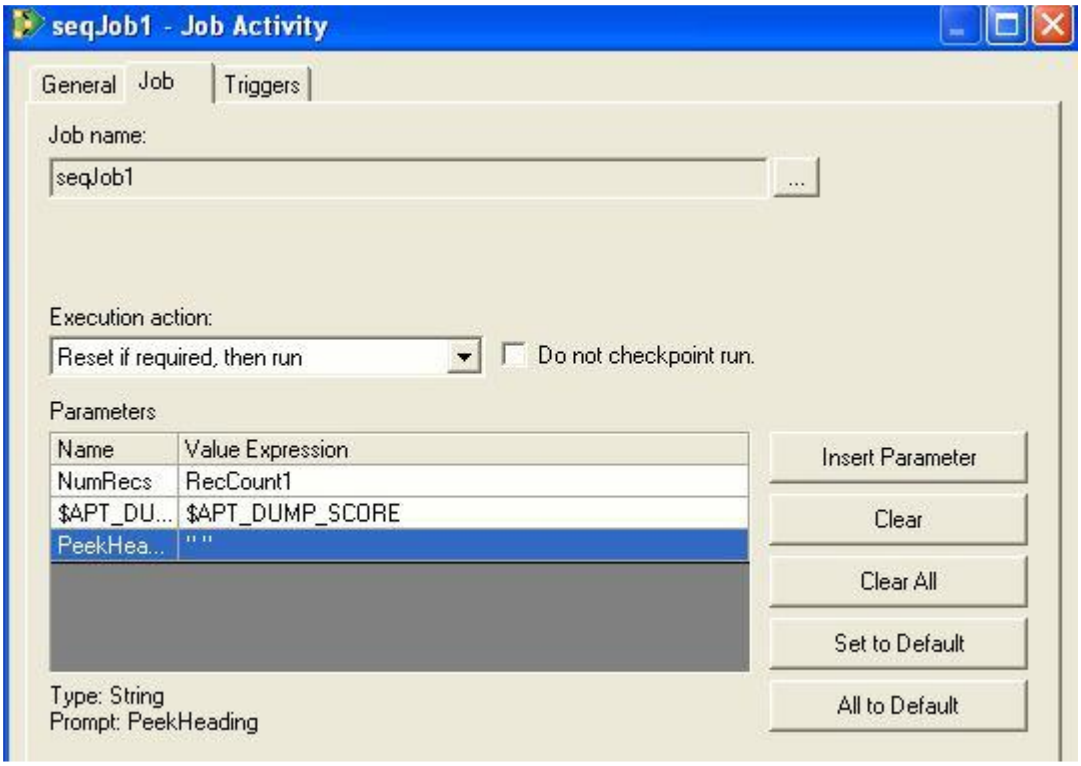
6. Open the Job Properties to the General tab. Read and check all the compilation options.



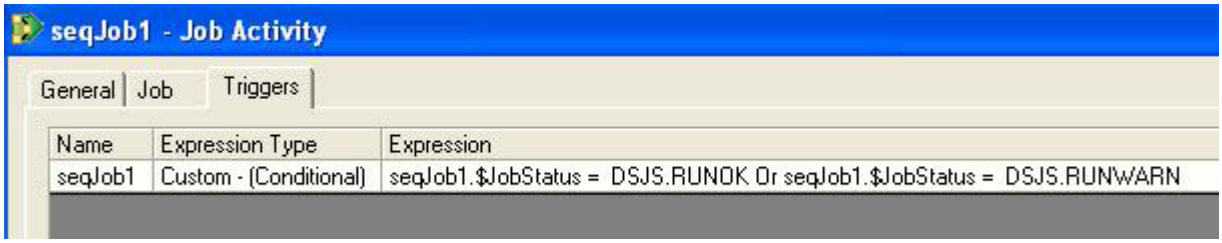
7. Add job parameters to the job sequence to supply values to the job parameters in the jobs.



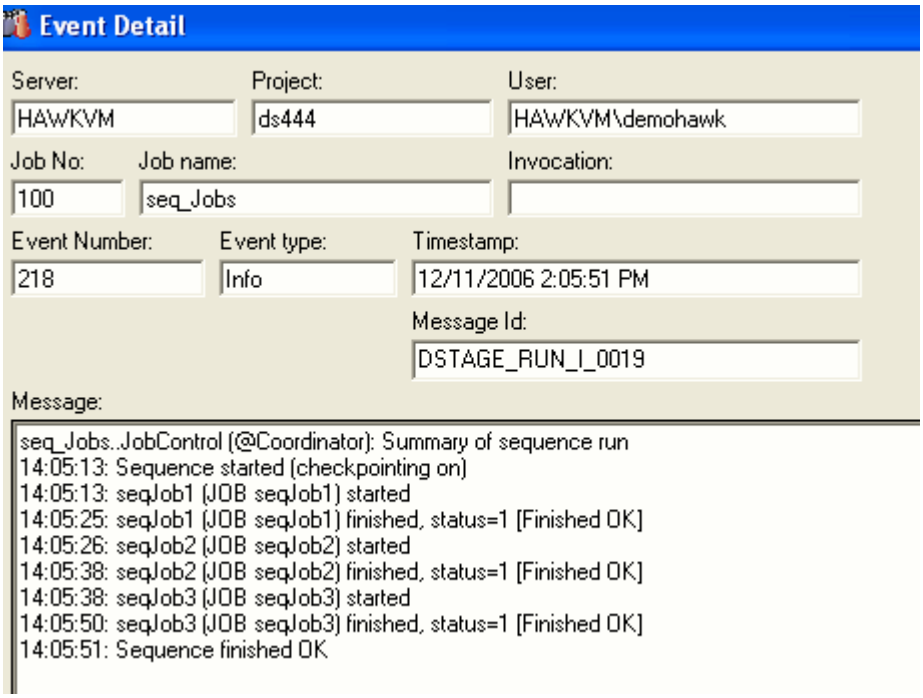
8. Open up each of the Job Activity stages and set the job parameters in the Activity stages to the corresponding job parameters of the Job Sequence. For PeekHeading value use a string with a single space.



9. In each of the first two Job Activity stages, set the job triggers so that later jobs only run if earlier jobs run without errors, although possibly with warnings.
    a. Note: This means that the **DSJ.JOBSTATUS** is either **DSJS.RUNOK**or **DSJS.RUNWARN** .
    b. To create this using one trigger link, create a custom trigger such that the previous job's job status is equal to one of the above two values. Click the right mouse button in the expression window to insert the Activity variable.



10. Compile and run your job sequence.
11. Open the job log for the sequence. Verify that each job ran successfully and examine the job sequence summary.

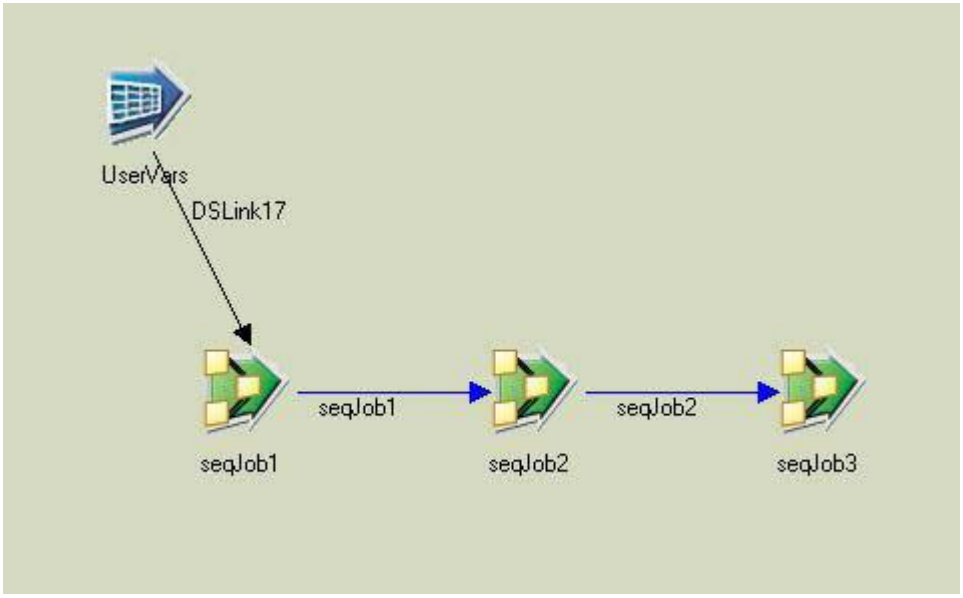


12. Examine what happens if the second job aborts. To see this pass -10 as the record count (RecCount2). Notice from the log and job summary that the first job runs OK but the second job aborts. This aborts the job sequence

13. Re-run the job using valid parameter values. Notice that the job restarts with seqJob2 because the Restart option was selected on the Job Properties window and seqJob1 ran successfully in the previous run.

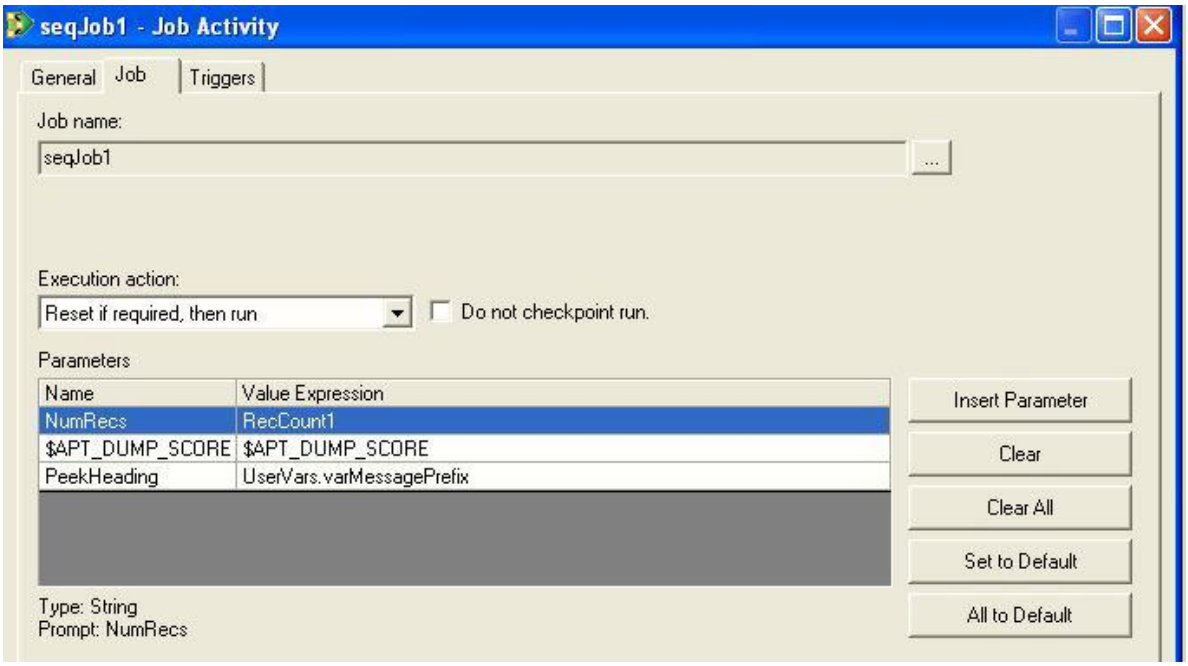
Task: Add a user variable

1. Save your job as seq_Jobs_2. Add a User Variables stage as shown.

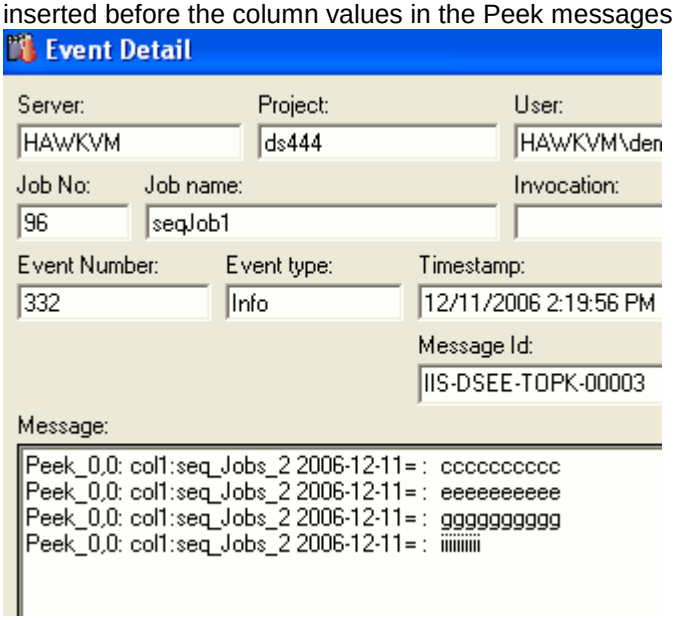


2. Open the User Variables stage to the User Variables tab. Create a user variable named varMessagePrefix. Open the expression editor. Concatenate the DSJobName DSMacro, a single space, the DSJobStartDate DSMacro, and an equal sign.

3. Open each Job Activity stage. For each PeekHeading parameter insert the varMessagePrefix for its value expression.



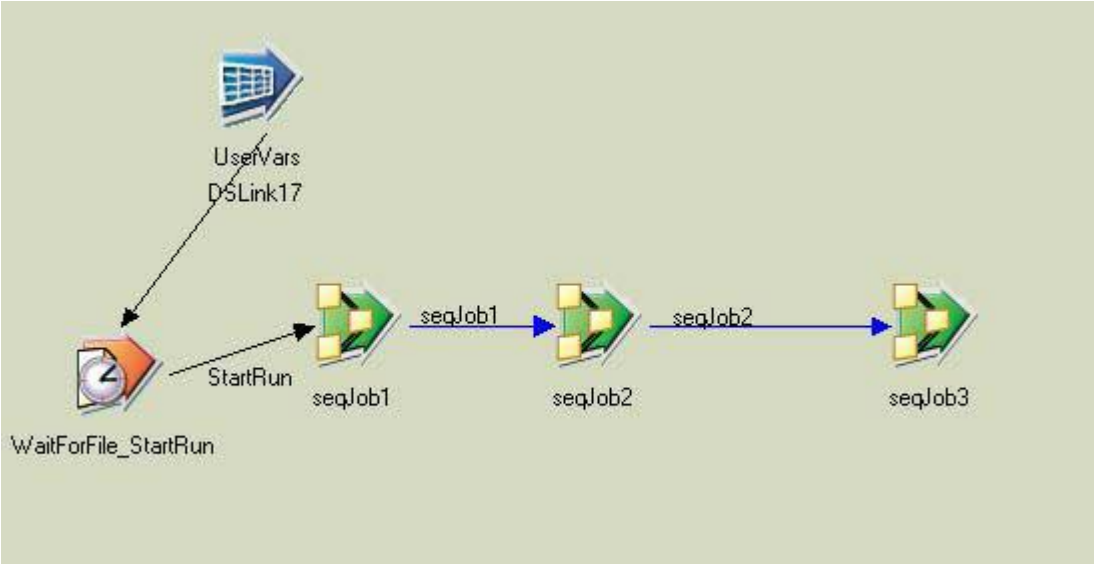
4. Compile and run. View the job log for the seqJob1 job. Verify that the PeekHeading is inserted before the column values in the Peek messages in the log.



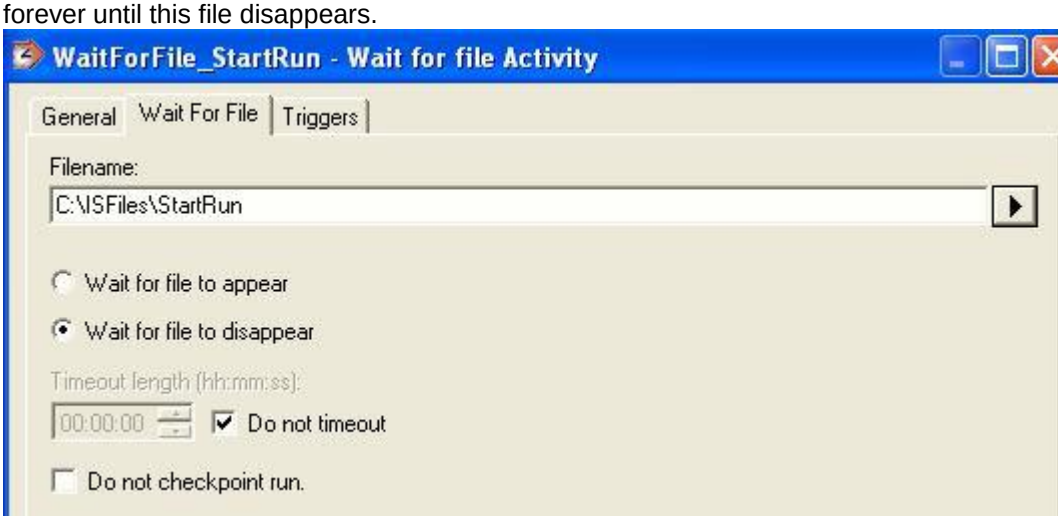
Task: Add a Wait for File stage

In this task, you modify your design so that the your job isn't executed until the Start Run file disappears from your ISFiles directory.

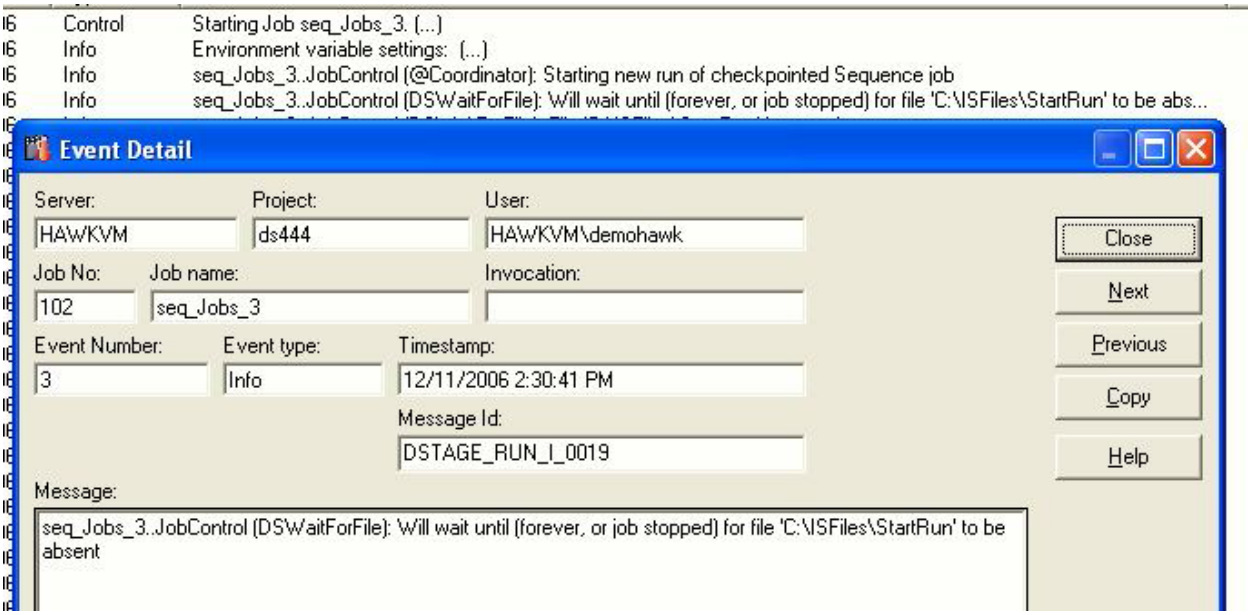
1. Save your job as seq_Jobs_3.
2. Add Wait for File stage as shown.



3. Edit the Wait for File stage. Select the StartRun file. Specify that the job is to wait forever until this file disappears.

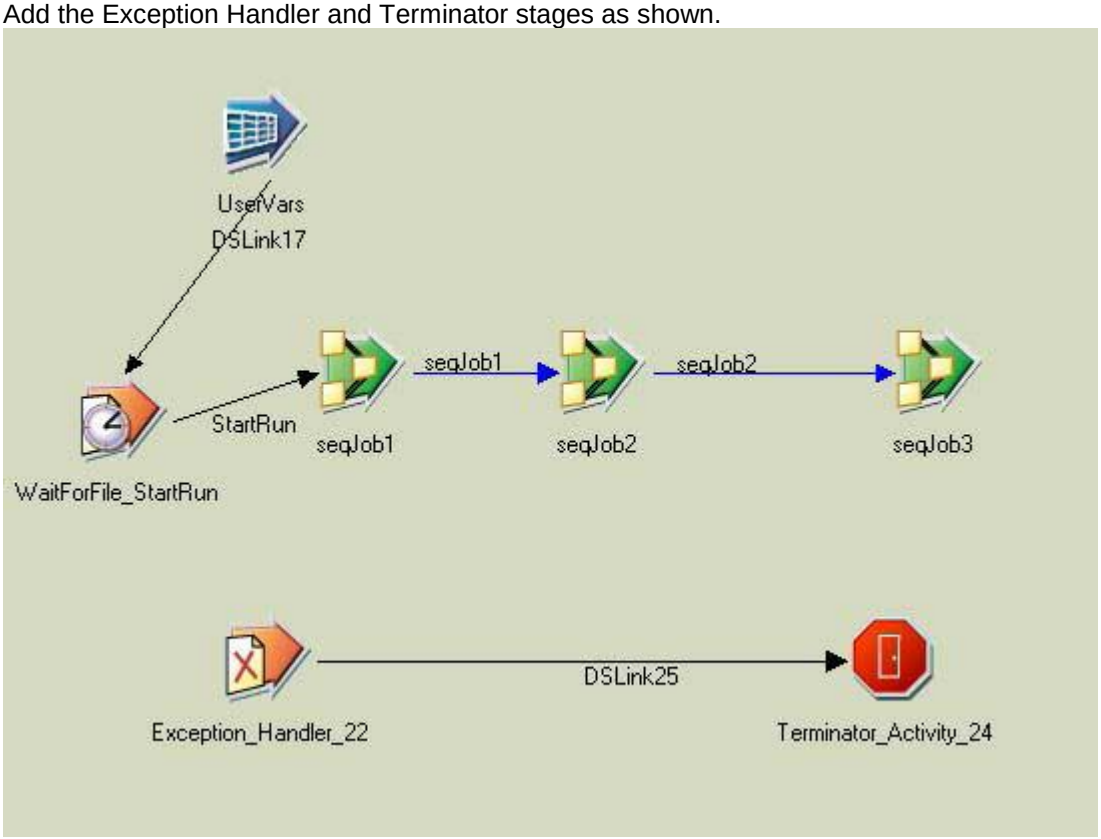


4. Define an unconditional trigger.
5. Compile and run your job. Test the Wait for File stage by first starting your job. After you view the log, rename the StartRun file (so that the StartRun file disappears).

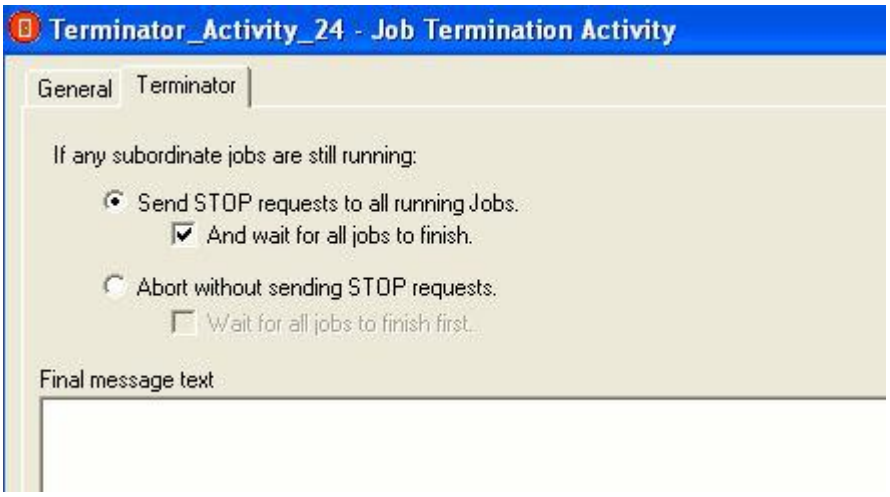


Task: Add exception handling

1. Save your job as seq_Jobs_4.
2. Add the Exception Handler and Terminator stages as shown.



3. Edit the TerminateJobs stage so that any running jobs are stopped when an exception occurs.



4. Compile and run your job. To test that it handles exceptions make an Activity fail. For example, set one of the run count parameters to -10.