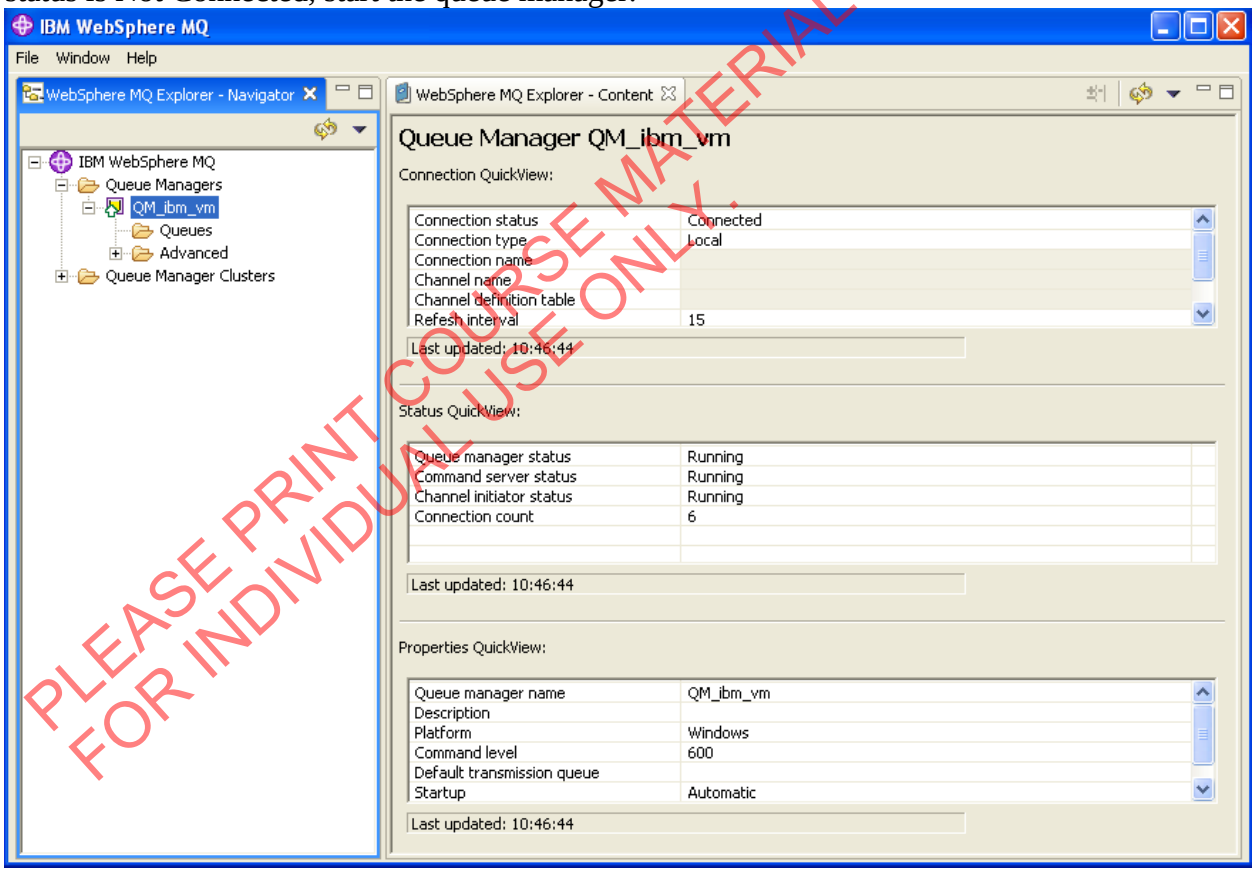
Special Topic 1: MQ Stage

Assumptions:

- IBM WebSphere MQ 6.0 is installed and configured.
- The MQ stage is installed on the DataStage Server.
  - Queue Manager name: QM_ibm_vm
  - Queue name: postcard

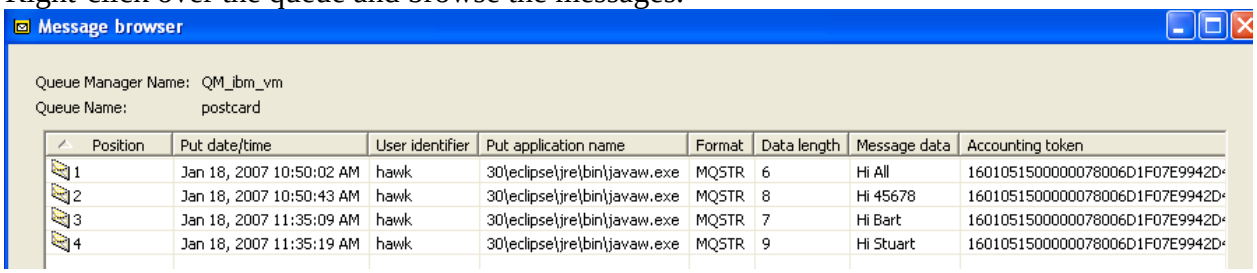
Task: Create MQ messages

1. Open up the WebSphere MQ Explorer.
2. Expand the Queue Managers folder. Select the queue manager. If its connection status is Not Connected, start the queue manager.

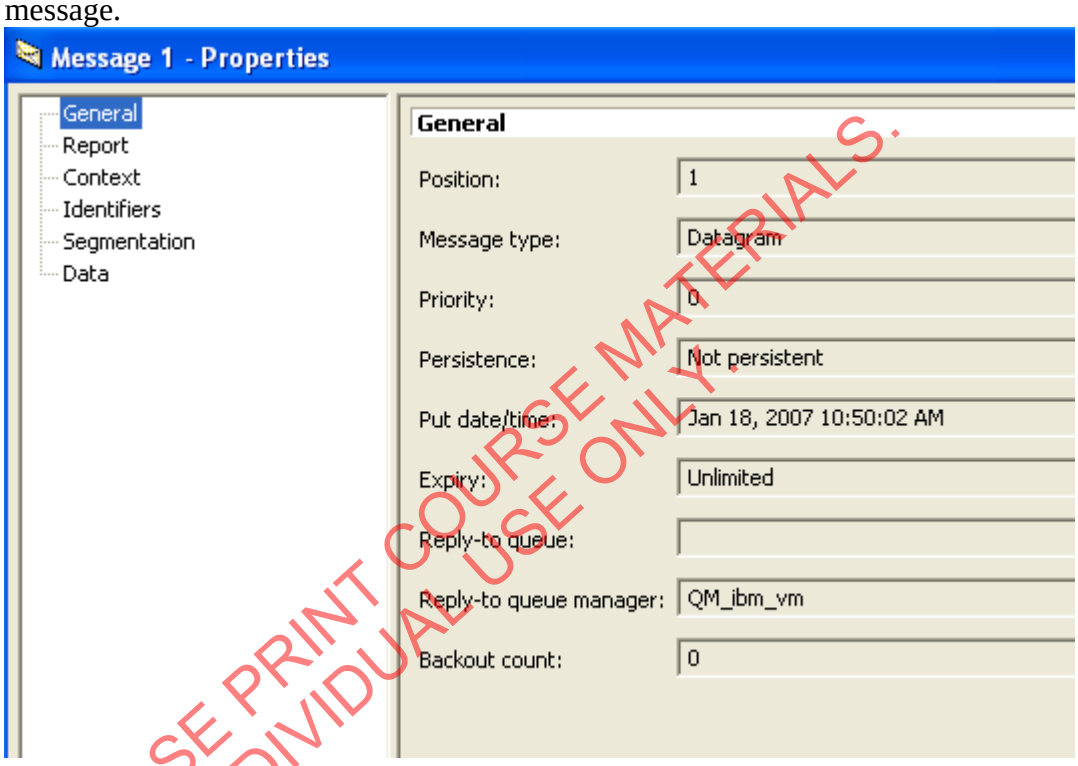


3. Click on the Queues folder. Select queue named postcard and then click your right mouse button. Click Put test message. Add four messages into the queue. Choose your own message data.

4. Right-click over the queue and browse the messages.

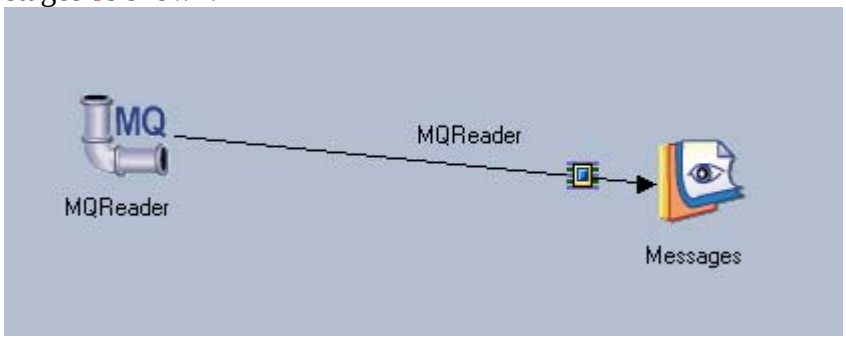


5. Select your first message and then click Properties. Explore the properties of your message.

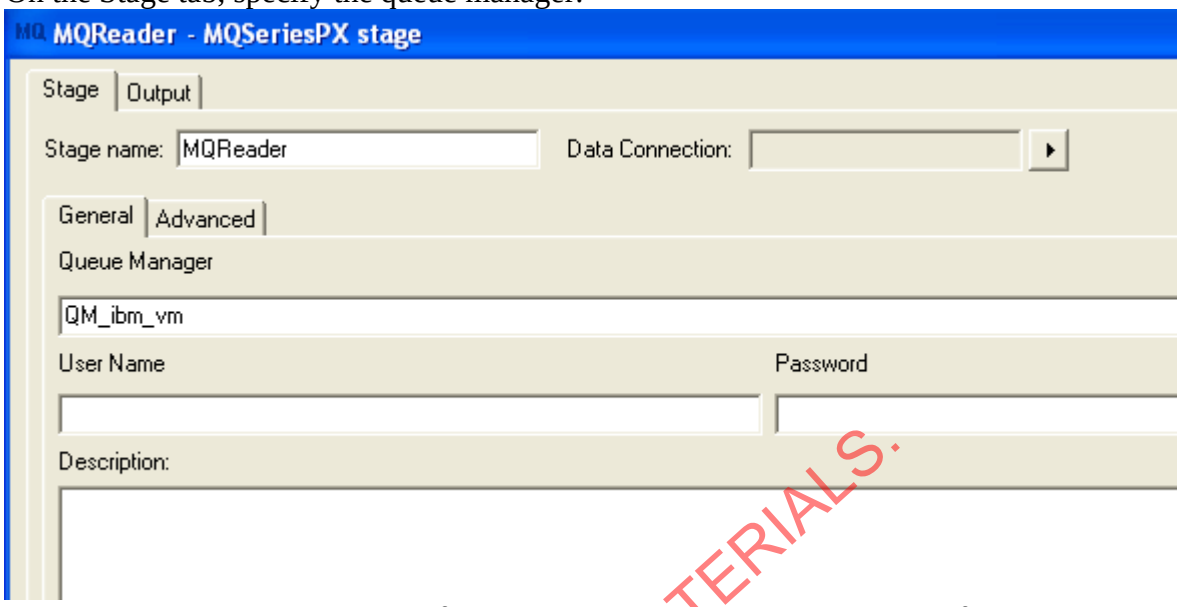


Task: Read MQ messages

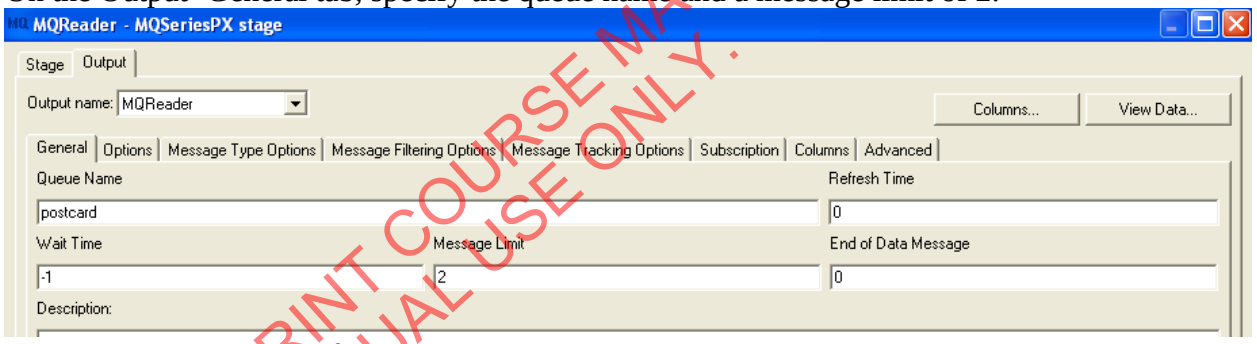
1. Create a new parallel job named mqMessages. Add the links and stages as shown. The leftmost is an MQ stage found in the Real Time folder. Name the links and stages as shown.



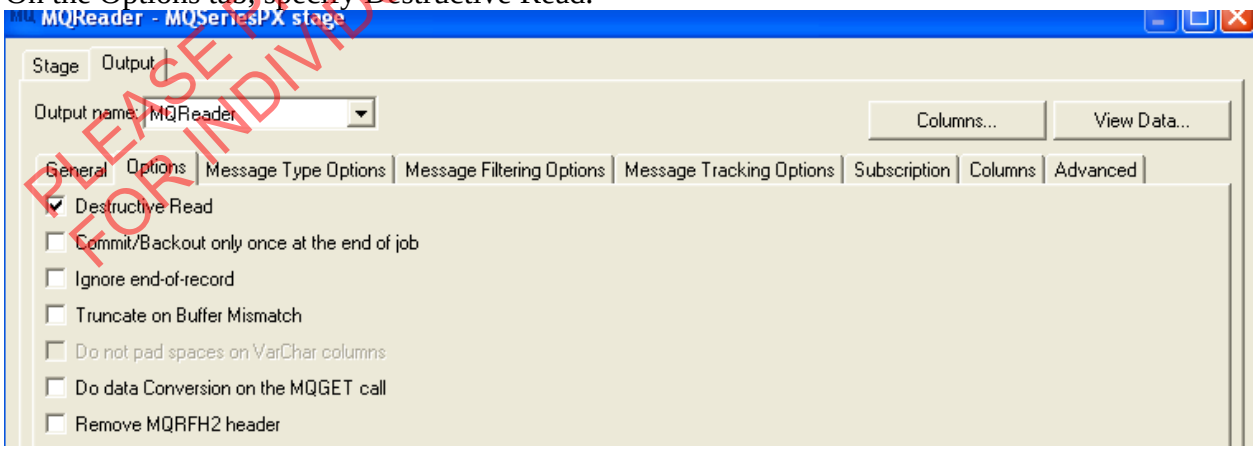
2. On the Stage tab, specify the queue manager.



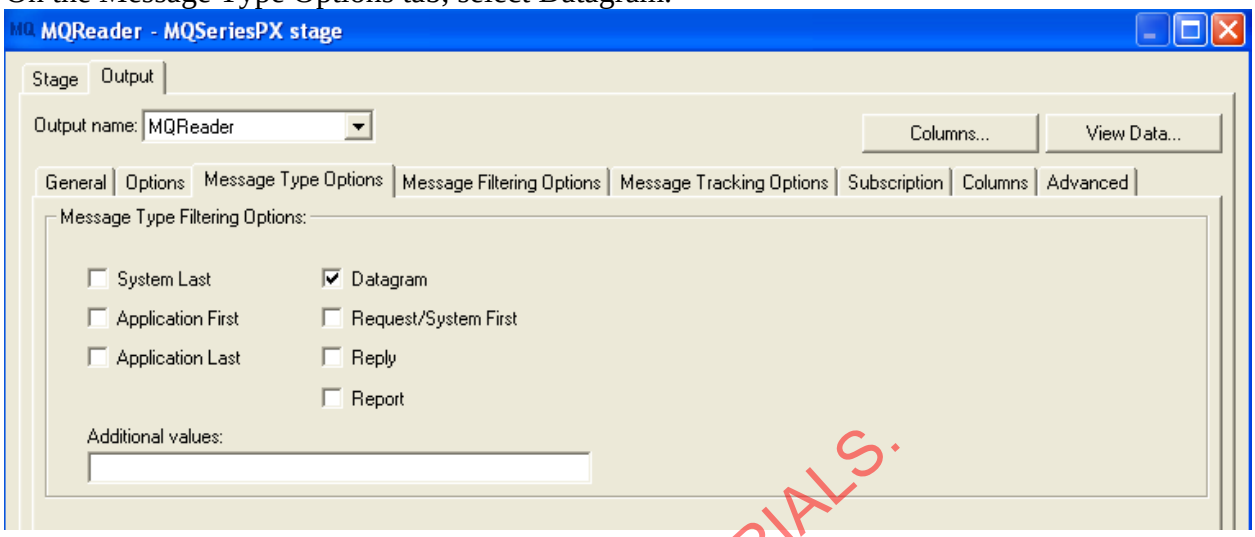
3. On the Output>General tab, specify the queue name and a message limit of 2.



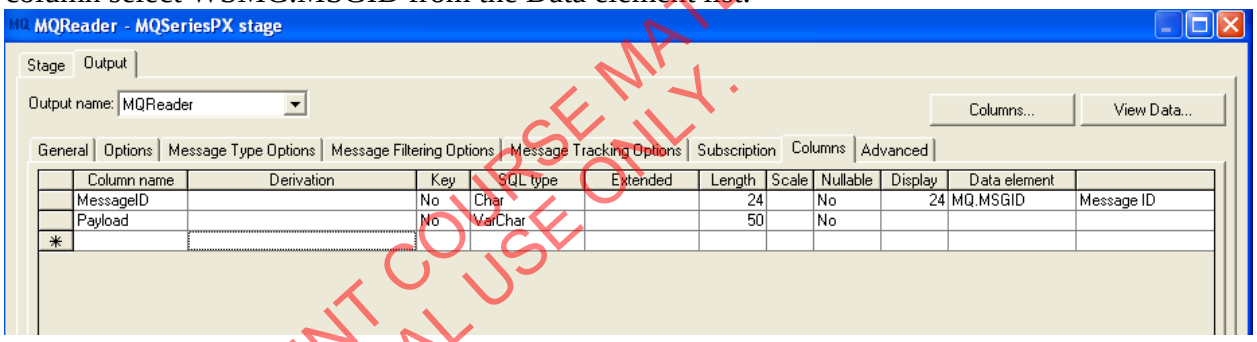
4. On the Options tab, specify Destructive Read.



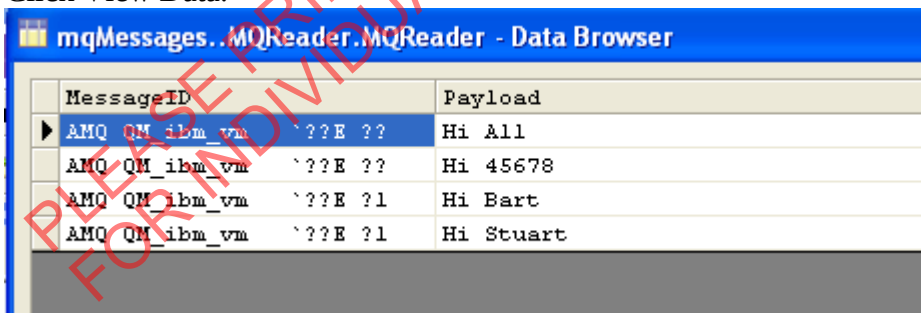
5. On the Message Type Options tab, select Datagram.



6. Click on the Columns tab. Add the two columns shown below. For the MessageID column select WSMG.MSGID from the Data element list.



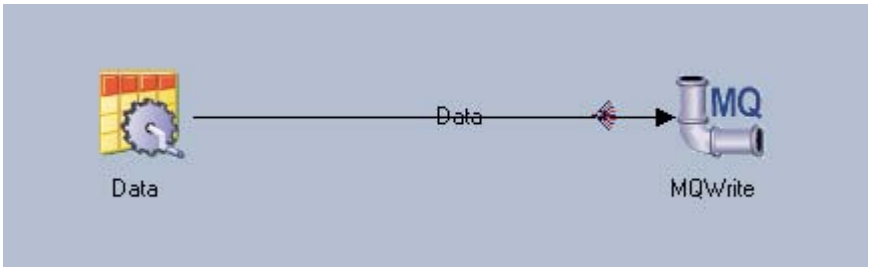
7. Click View Data.



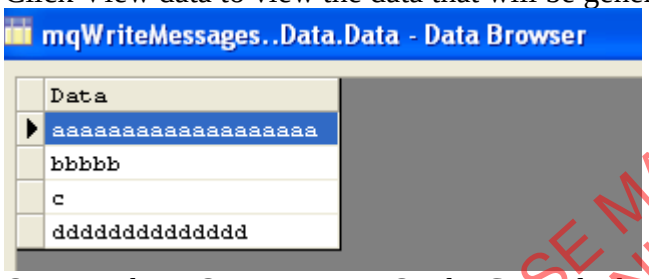
8. Compile and run.
9. View the job log. Find your Peek messages in the log.

Task: Write MQ messages

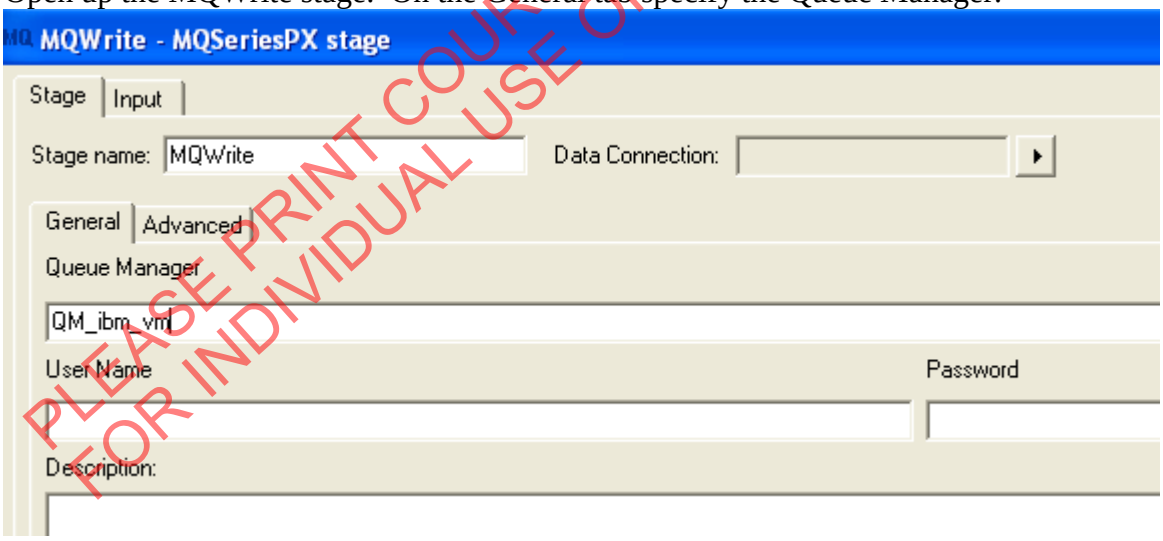
1. Create a new parallel job named mqWriteMessages. Add the stages and links as shown.



2. Open the Row Generator stage. The number of records equals 4. Add one character column named Data, VarChar(20).
3. Click View data to view the data that will be generated.

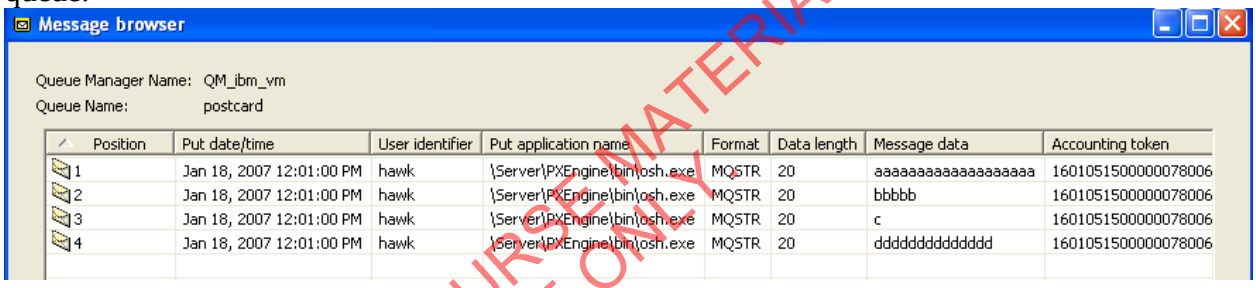


4. Open up the MQWrite stage. On the General tab specify the Queue Manager.



5. On the Input>General tab specify the name of the queue. Examine the defaults.

6. Compile and run your job. Check the Director log for errors.
7. Open up the MQ Exporter message browser and view the messages written to the queue.



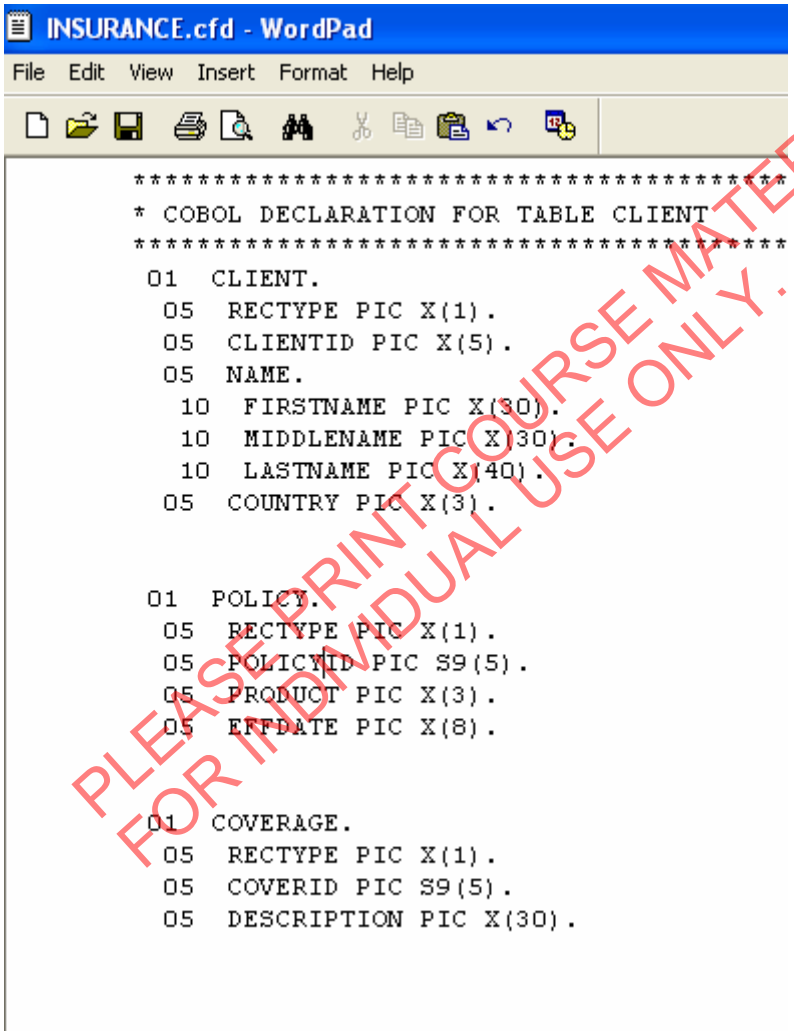
Special Topic 2: Complex Flat File

Assumptions

- The INSURANCE.cfd, INSURANCE.data.txt, and INSURANCE.data_reformatted.txt files are in your ISFiles>ComplexFlatFile directory.

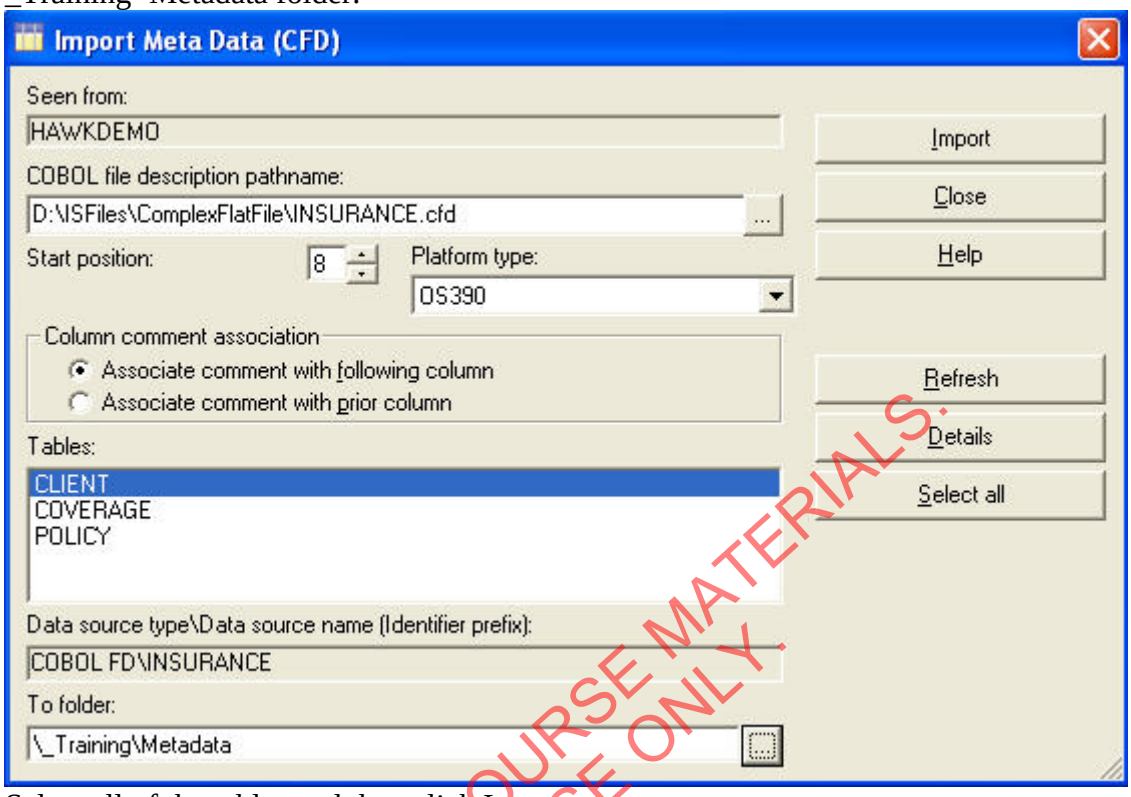
Task: Import a Cobol file definition

1. Open the INSURANCE.cfd file in WordPad in the ISFiles>ComplexFlatFile directory. Examine the file. Note the location of level 01 items, the number of level 01 items, and the names of the level 01 items.

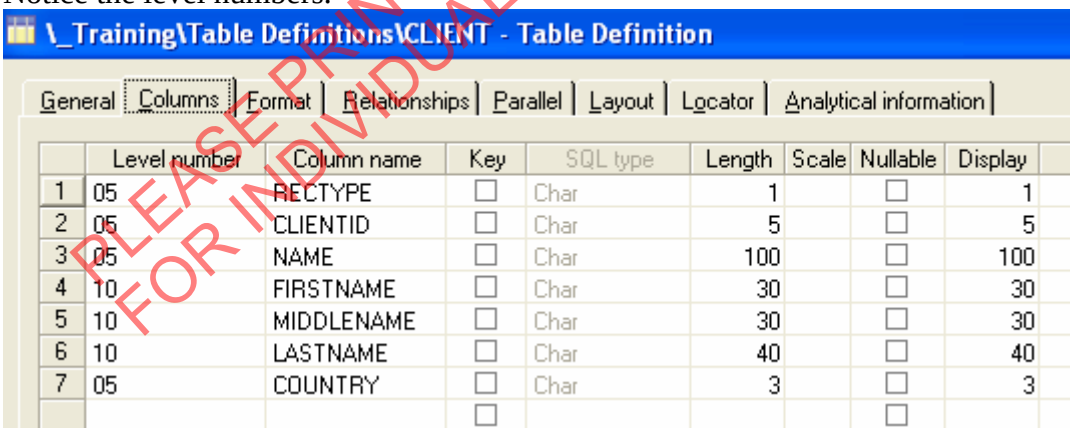


2. In DataStage Designer, click Import>Table Definitions and then select Cobol File Definitions.
3. Select the path to the INSURANCE.cfd file.
4. Specify a start position of 8.

5. In the To folder box, specify that you want to store the table definitions in your _Training>Metadata folder.

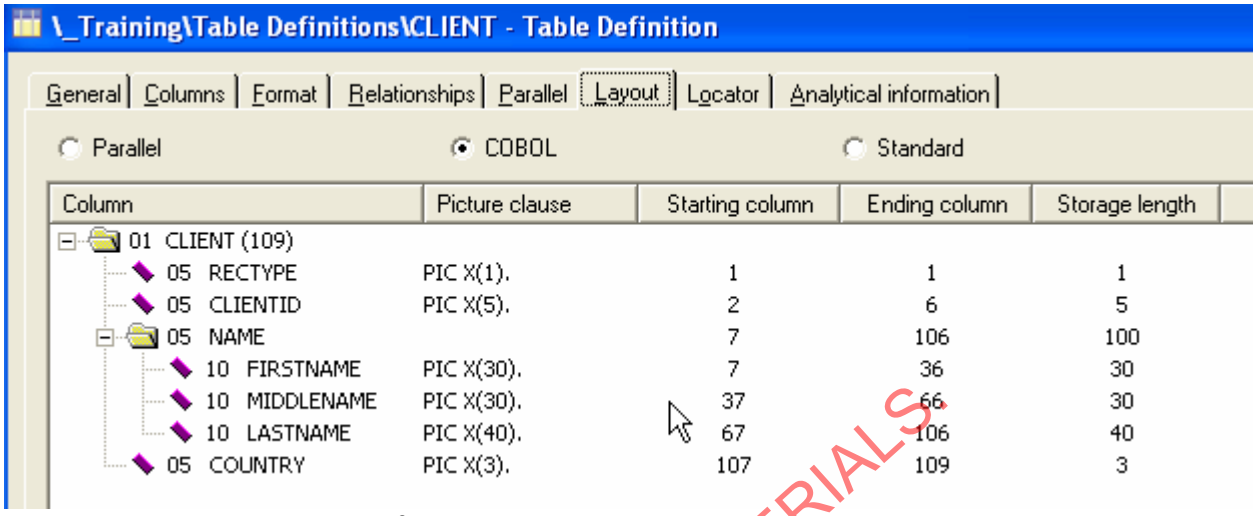


6. Select all of the tables and then click Import.
7. Open the CLIENT table definition. Click on the Columns tab. Examine the layout. Notice the level numbers.



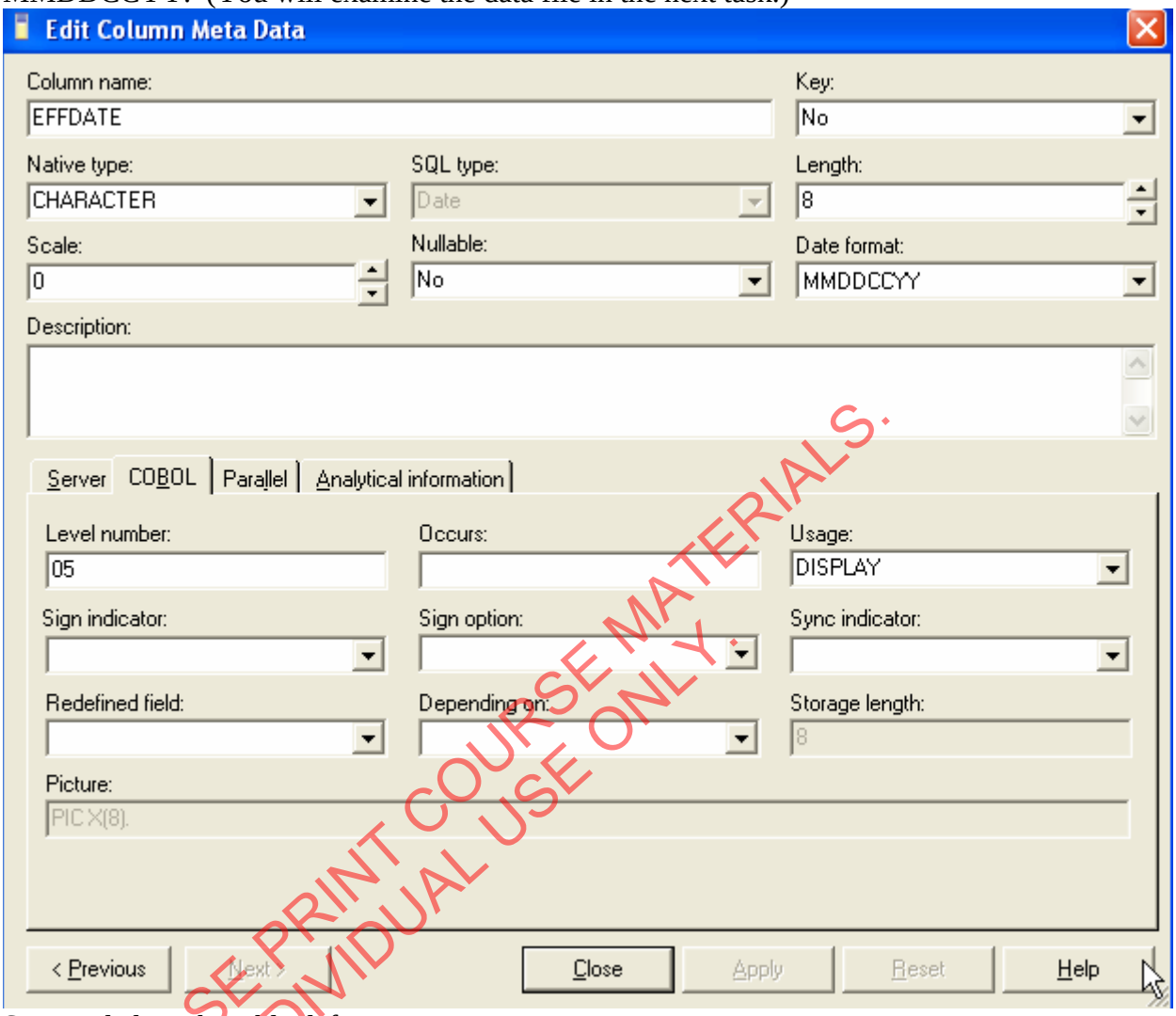
	Level number	Column name	Key	SQL type	Length	Scale	Nullable	Display
1	05	RECTYPE	☐	Char	1		☐	1
2	05	CLIENTID	☐	Char	5		☐	5
3	05	NAME	☐	Char	100		☐	100
4	10	FIRSTNAME	☐	Char	30		☐	30
5	10	MIDDLENAME	☐	Char	30		☐	30
6	10	LASTNAME	☐	Char	40		☐	40
7	05	COUNTRY	☐	Char	3		☐	3
			☐				☐	

8. Click on the Layout tab. Click the COBOL button. Examine the COBOL structure of the file definition.



9. Open up the POLICY table definition. Click on the Columns tab. Double-click to the left of the EFFDATE column to open the Edit Column Meta Data window.
10. Click on the COBOL tab at the bottom. Verify that the column name is EFFDATE. Select the date format that accurately describes the values in the data file, namely,

MMDDCCYY. (You will examine the data file in the next task.)

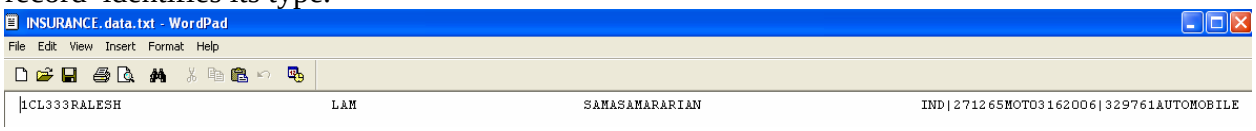


11. Save and close the table definition.
12. Open up and explore the COVERAGE table definition.

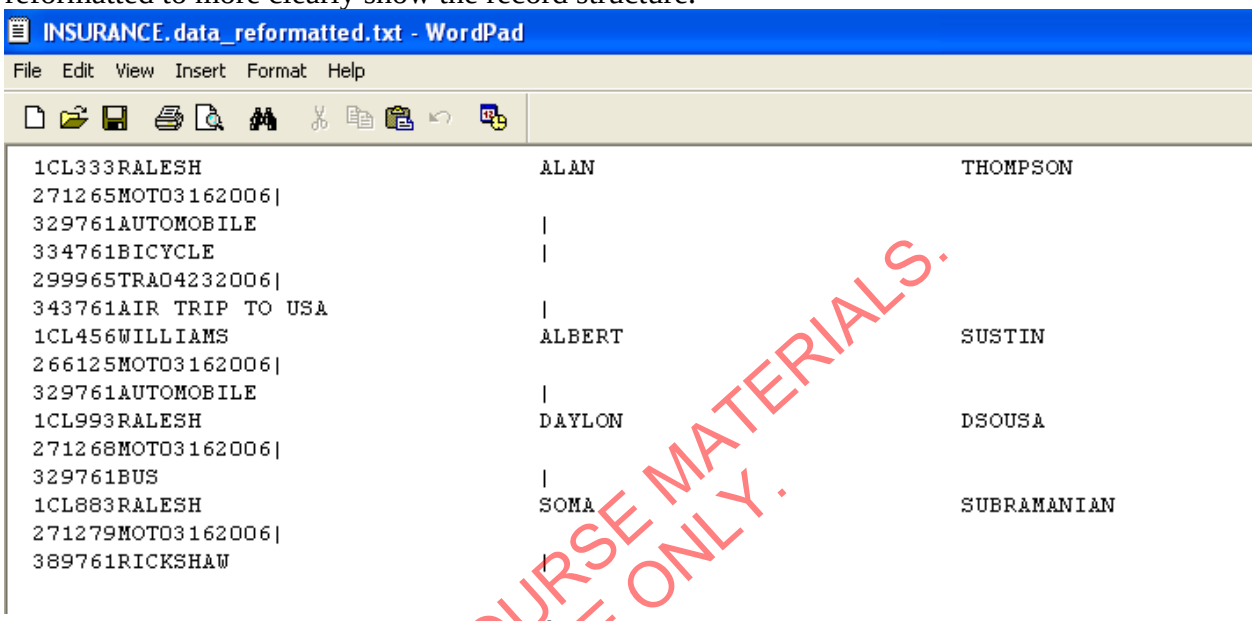
Task: Examine the multi-format data file

1. Open the Insurance.data.txt file in WordPad. This is the data file that will be read by the job we will build.
2. Examine at least the first three records. Here we see a CLIENT record (Record type '1') followed by a POLICY record (Record type '2') followed by a COVERAGE record (Record type '3'). The records are delimited by the pipe (|). The first character of each

record identifies its type.



3. For clarity, open up the INSURANCE.data_reformatted.txt file. This is the data file, reformatted to more clearly show the record structure.

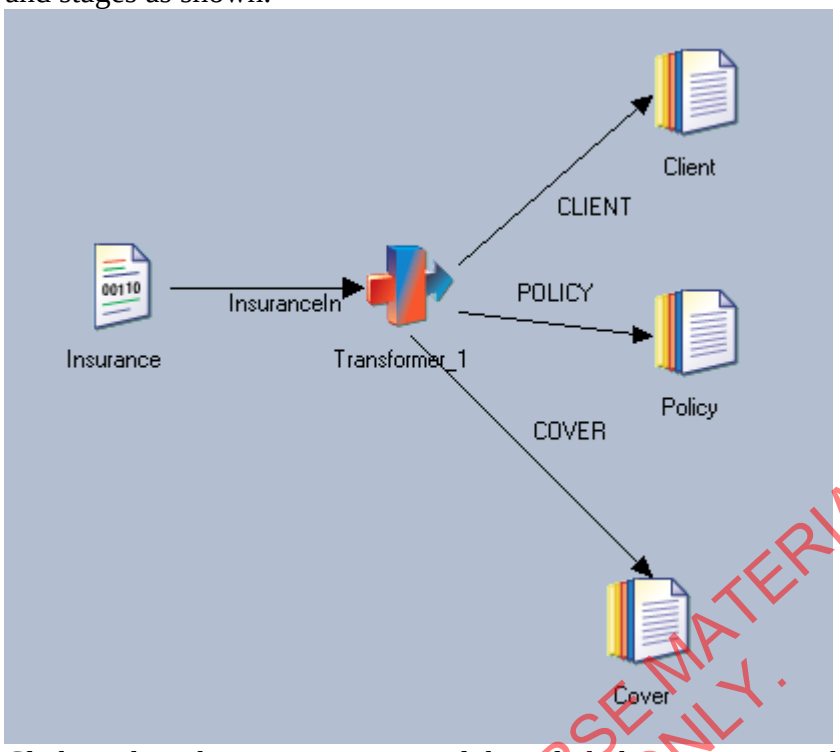


4. Compare the data records to their table definitions. Be sure you understand how the table definitions describe the three types of data records.

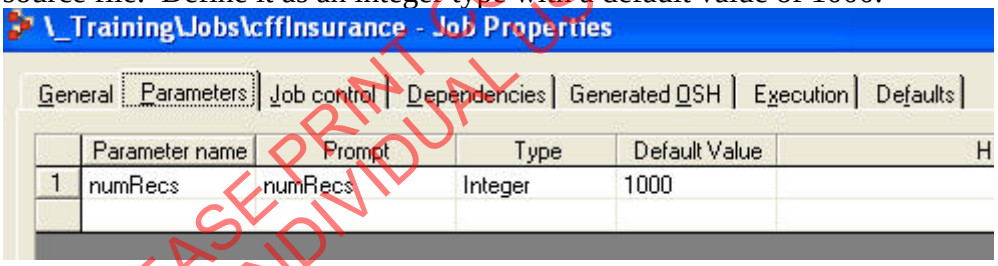
Task: Edit the Complex Flat File stage

1. Create a new parallel job named cffInsurance in your _Training>Jobs folder. The source stage is a Complex Flat File stage. The targets are Sequential File stages. Name the links

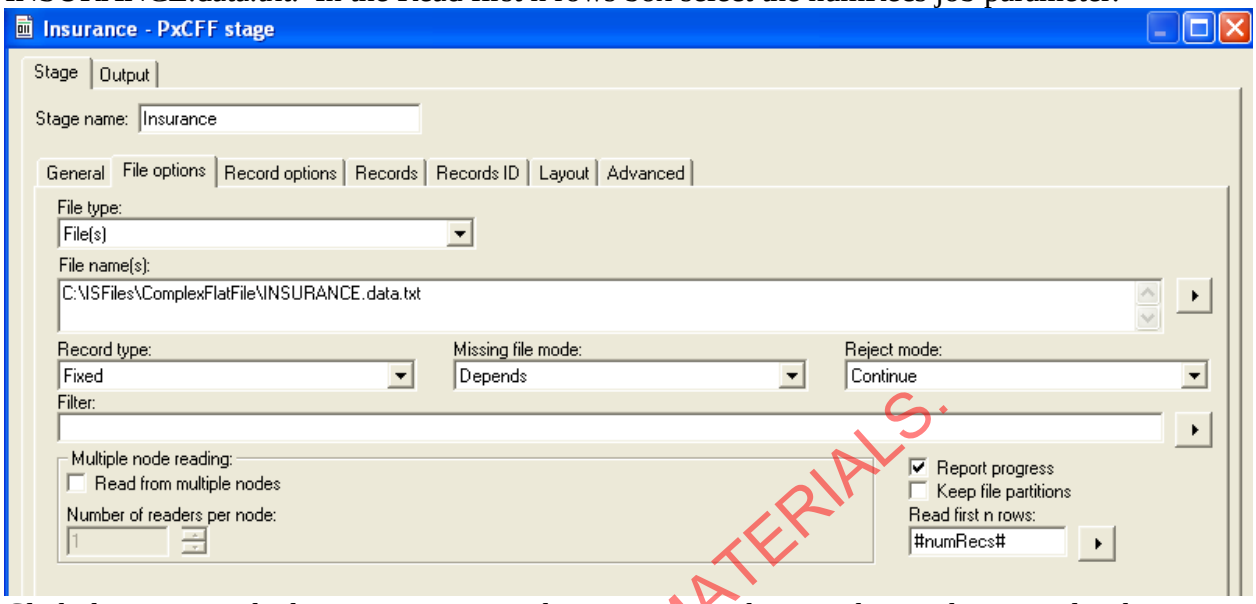
and stages as shown.



2. Click on the Job Properties icon and then click the Parameters tab. Define a job parameter named numRecs, which will contain the number of records to be read from the source file. Define it as an integer type with a default value of 1000.

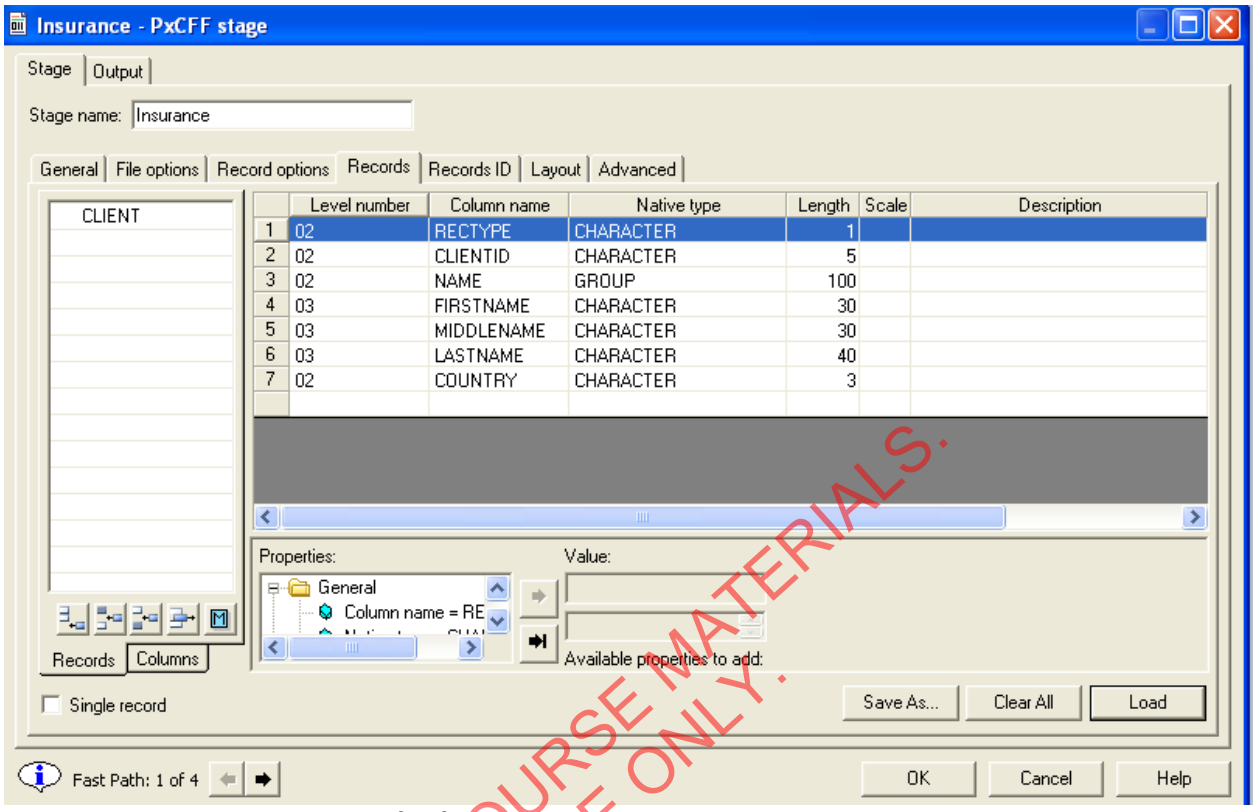


3. Open the Insurance CFF stage. On the File options tab, select the file to be read, namely, INSURANCE.data.txt. In the Read first n rows box select the numRecs job parameter.

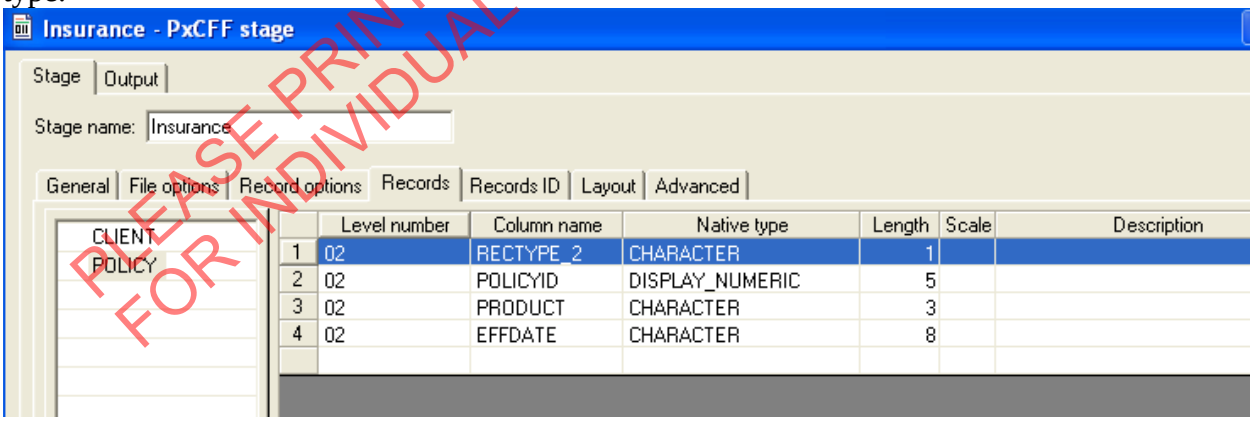


4. Click the arrow at the bottom to move to the next Fast Path page, that is, the Records tab. Remove the check from the Single record box.
5. Change the name of the default record type to CLIENT.
6. Click Load. Select all the columns from the CLIENT Table.

7. Click OK to load the columns.

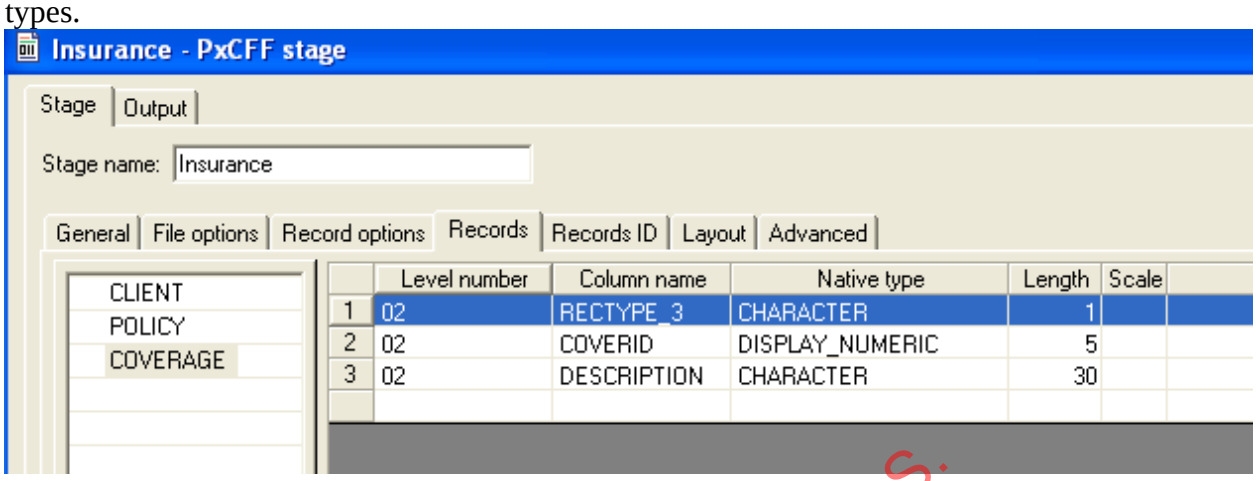


8. Click the icon at the bottom left of the Records tab to insert a new record type. Complete the process to define and load the Table Definition for the POLICY record type.

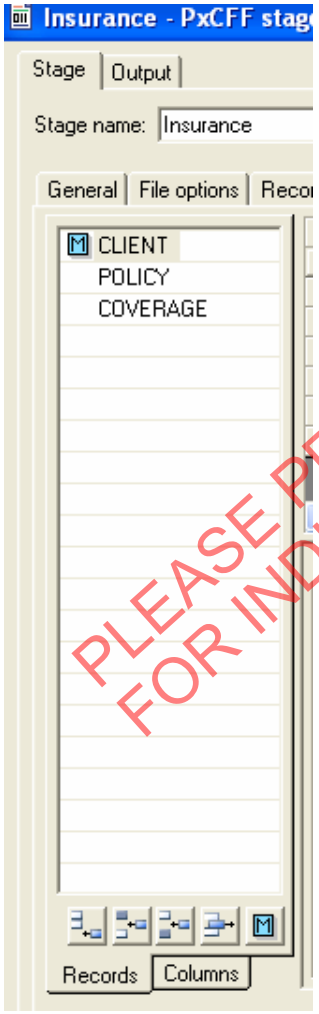


9. Click the icon at the bottom left of the Records tab to insert a new record type. Complete the process to define and load the Table Definition for the COVERAGE record

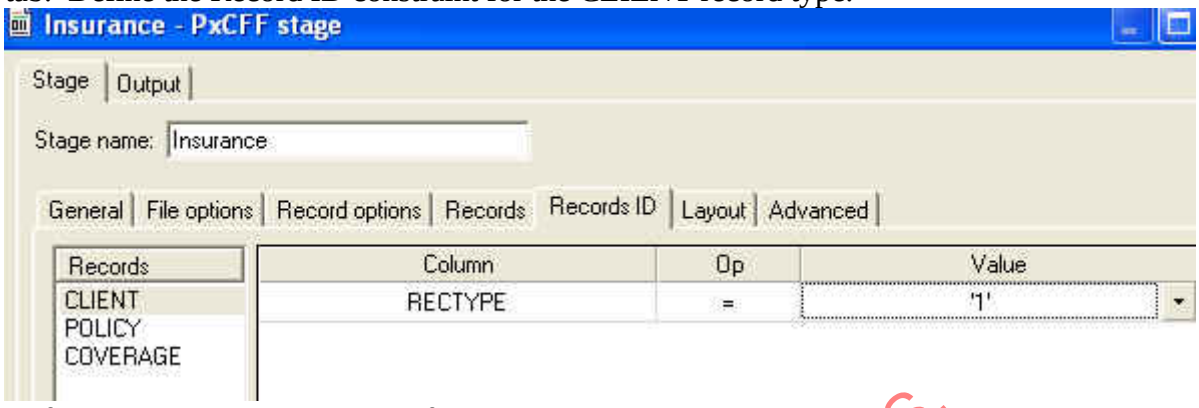
types.



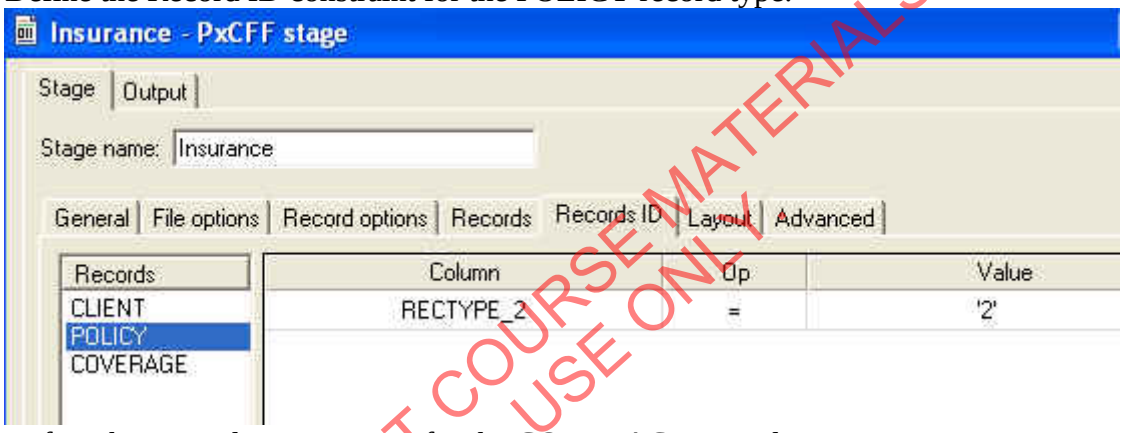
10. Select the CLIENT record type and then click the Master button (rightmost icon at the bottom of the records tab). This will make the CLIENT record type the master.



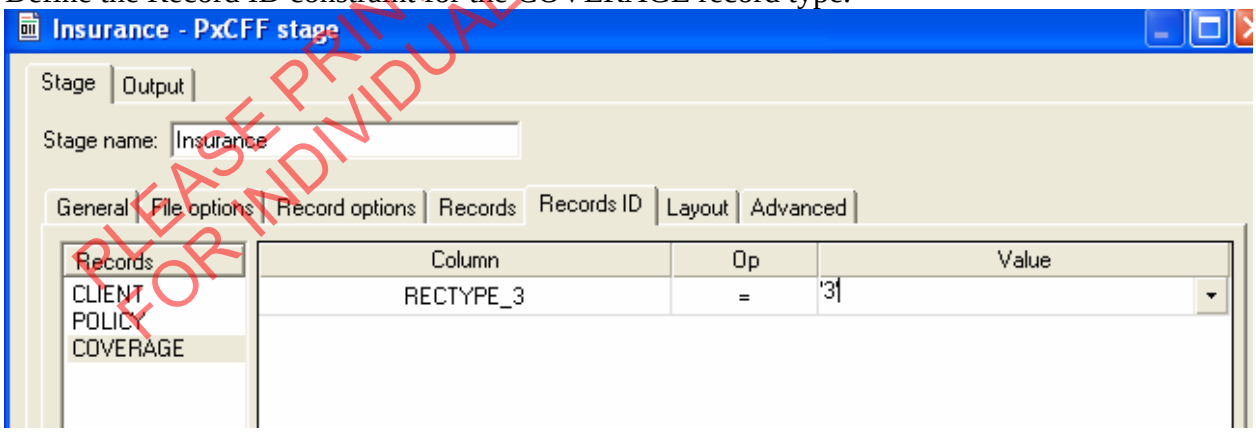
11. Click the arrow at the bottom to move to the next Fast Path page, that is, the Records ID tab. Define the Record ID constraint for the CLIENT record type.



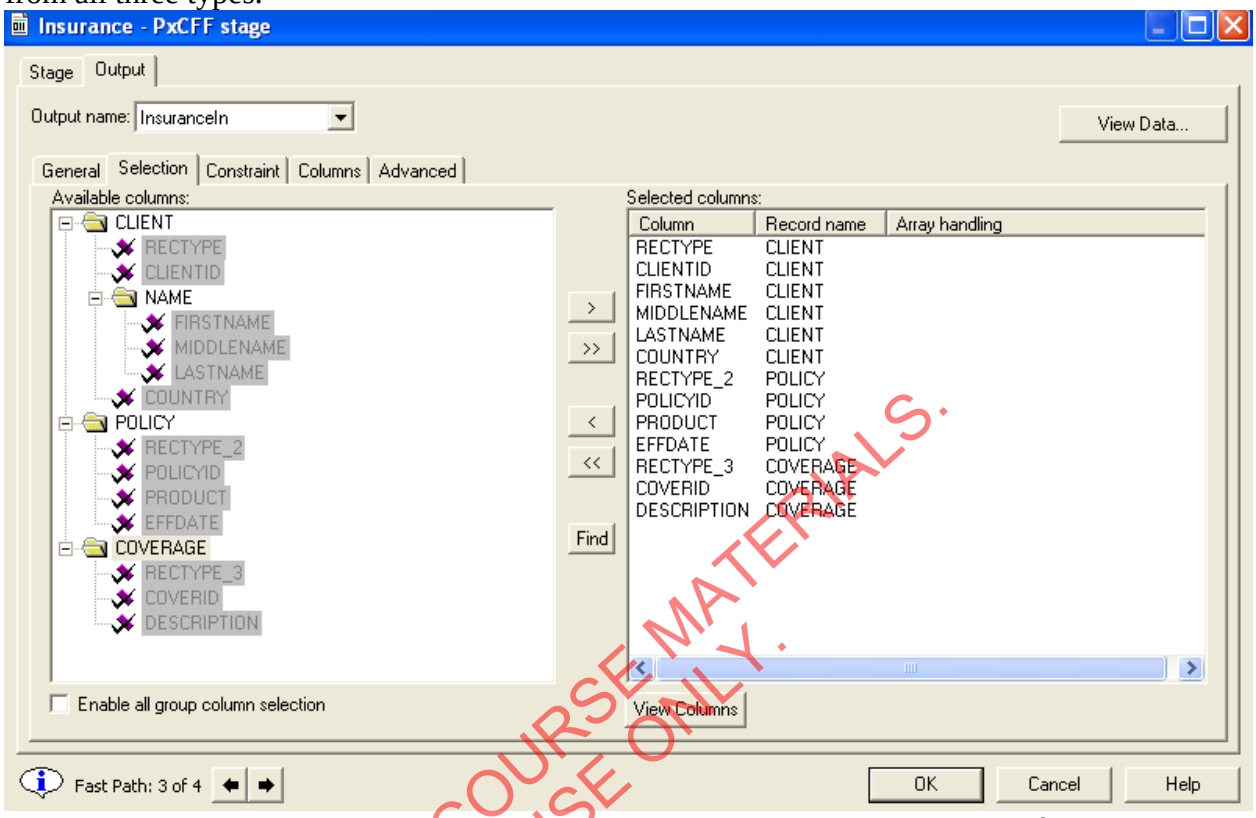
12. Define the Record ID constraint for the POLICY record type.



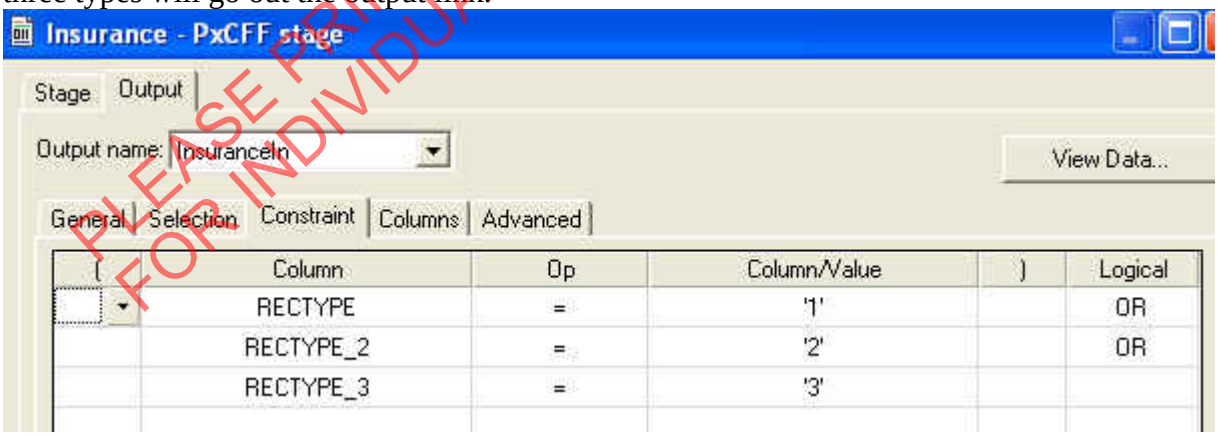
13. Define the Record ID constraint for the COVERAGE record type.



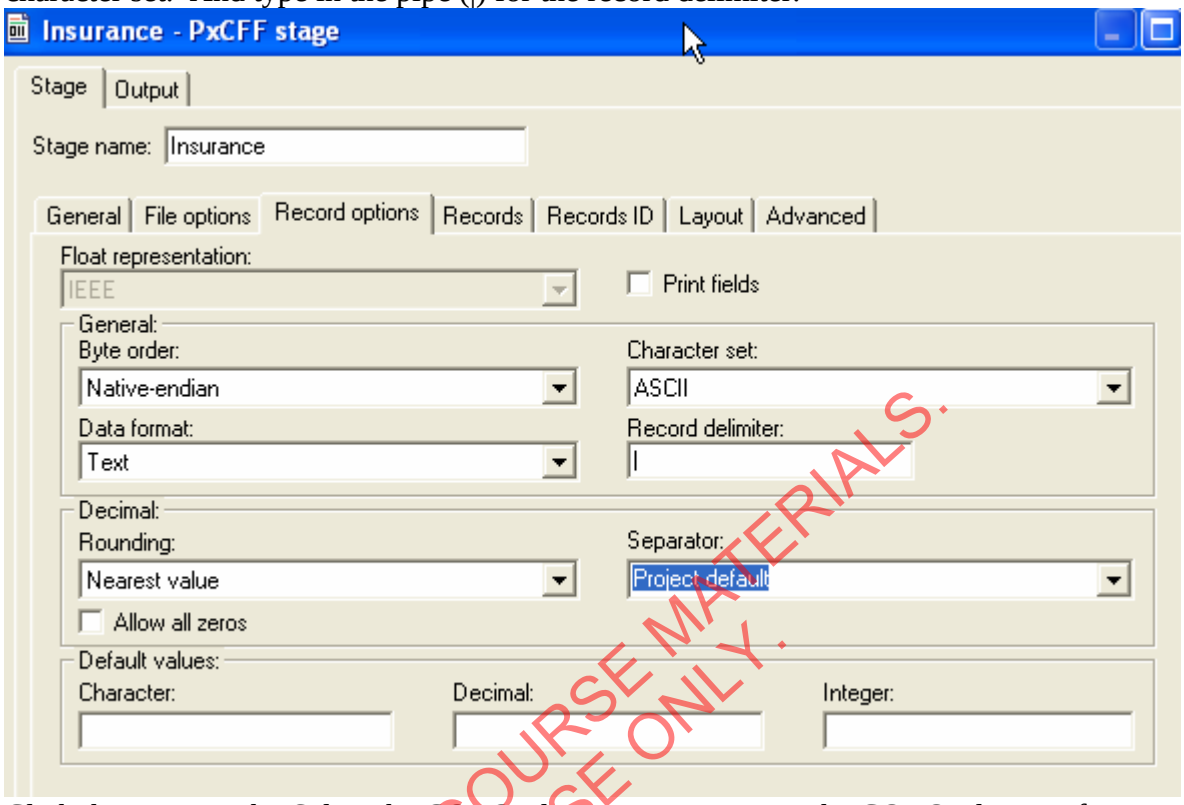
14. Move to the next Fast path page, that is, the Output>Selection tab. Select all columns from all three types.



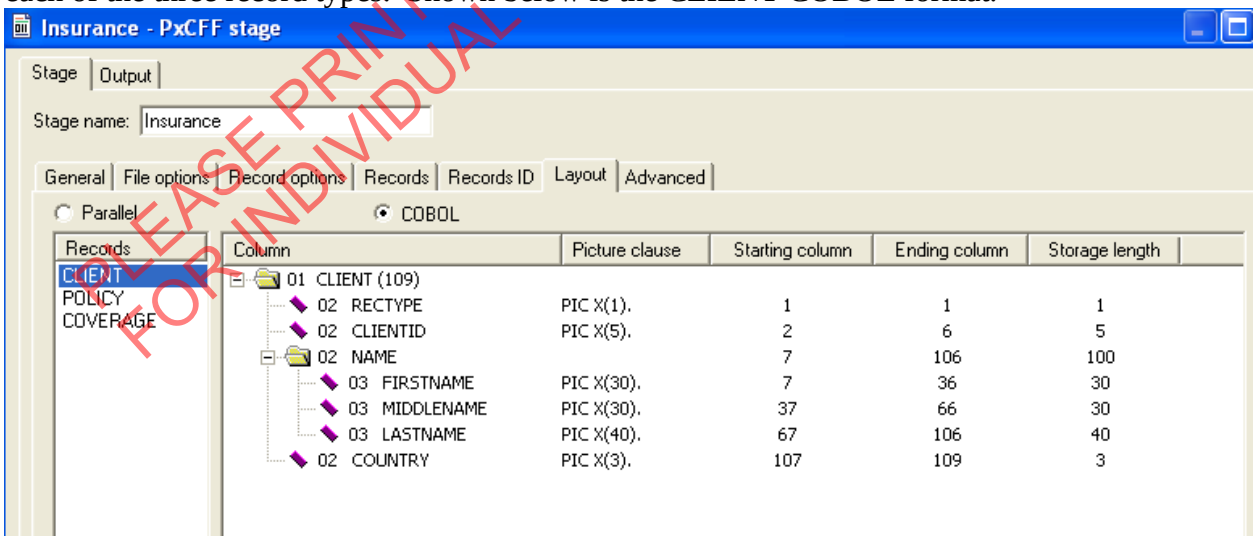
15. Move to the last Fast path page, that is, the Output>Constraint tab. Click the Default button to add the default output constraint. This will insure that only records of these three types will go out the output link.



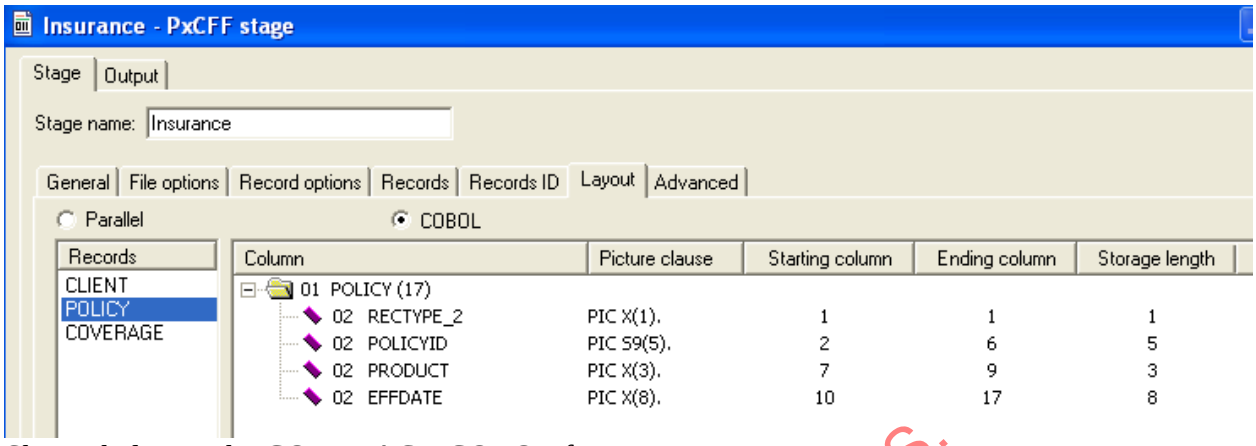
16. Click the Stage>Record options tab. Specify a Text data format. Select the ASCII character set. And type in the pipe (|) for the record delimiter.



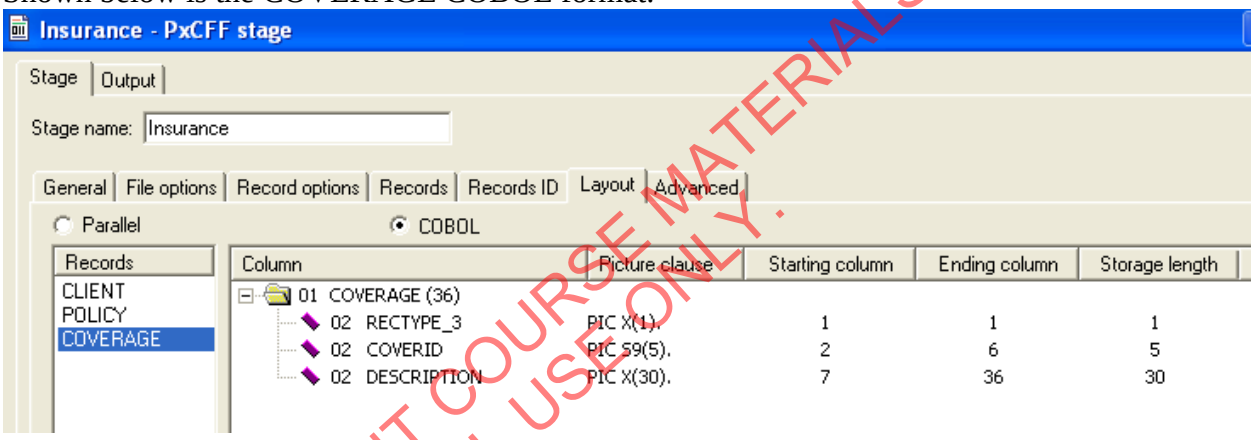
17. Click the Layout tab. Select the COBOL layout option. View the COBOL layouts for each of the three record types. Shown below is the CLIENT COBOL format.



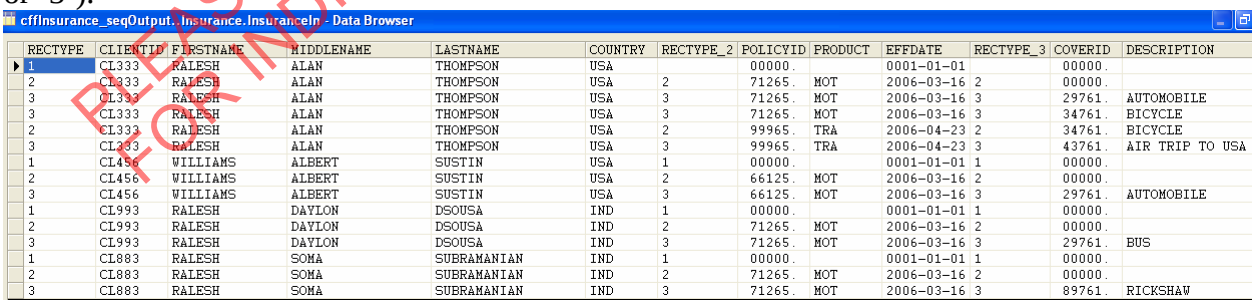
18. Shown below is the POLICY COBOL format.



19. Shown below is the COVERAGE COBOL format.



20. Click View Data. (You may have to move to another tab such as the Output>Selection tab to see it.) Notice that all the columns from all the record types are displayed with data in them. However, the data is only valid for the columns of a given record type when the RECTYPE field for the record contains the record identifier character ('1', '2', or '3').

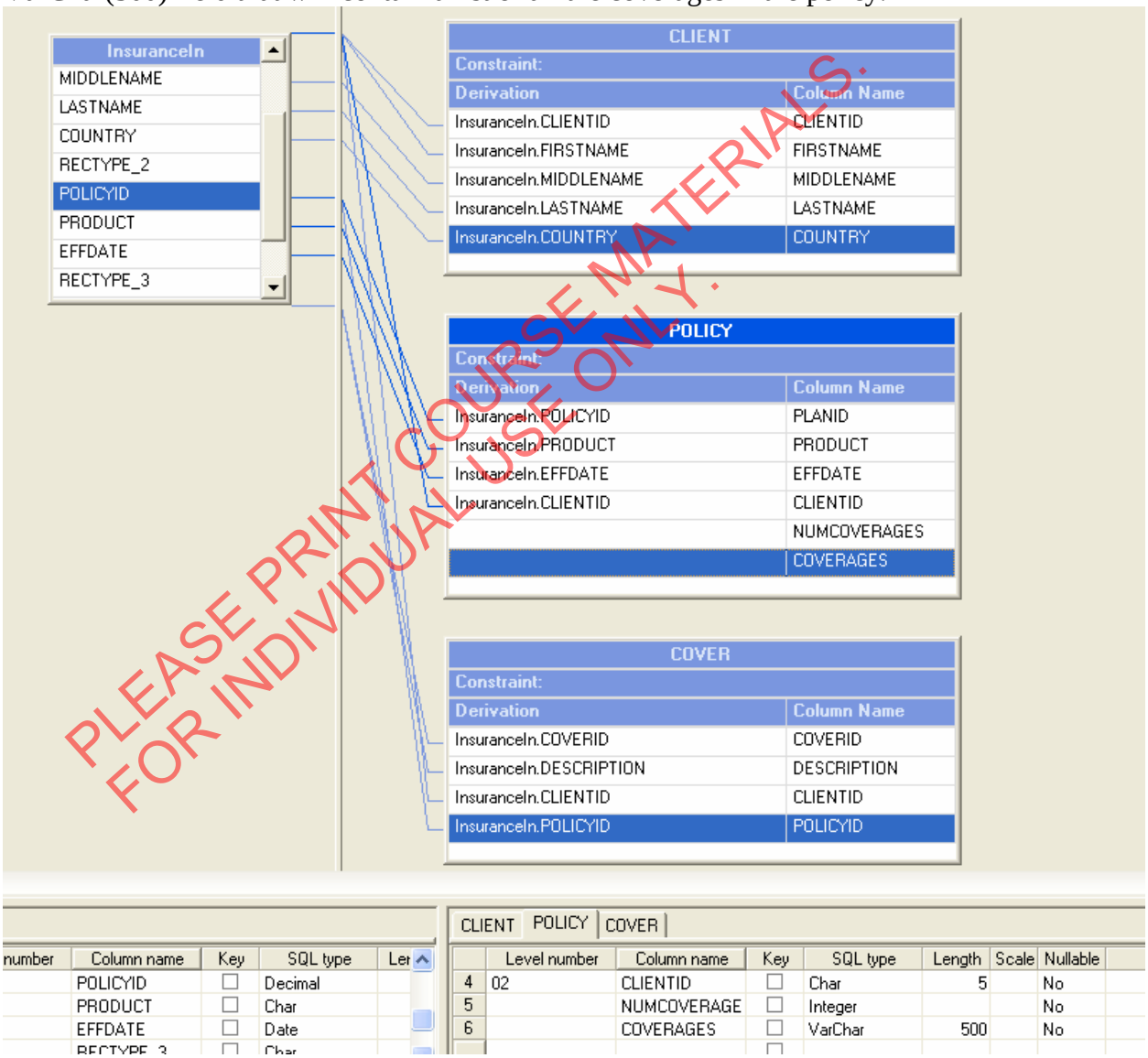


RECTYPE	CLIENTID	FIRSTNAME	MIDDLENAME	LASTNAME	COUNTRY	RECTYPE_2	POLICYID	PRODUCT	EFFDATE	RECTYPE_3	COVERID	DESCRIPTION
1	CL333	RALESH	ALAN	THOMPSON	USA		00000.		0001-01-01		00000.	
2	CL333	RALESH	ALAN	THOMPSON	USA	2	71265.	MOT	2006-03-16	2	00000.	
3	CL333	RALESH	ALAN	THOMPSON	USA	3	71265.	MOT	2006-03-16	3	29761.	AUTOMOBILE
3	CL333	RALESH	ALAN	THOMPSON	USA	3	71265.	MOT	2006-03-16	3	34761.	BICYCLE
2	CL333	RALESH	ALAN	THOMPSON	USA	2	99965.	TRA	2006-04-23	2	34761.	BICYCLE
3	CL333	RALESH	ALAN	THOMPSON	USA	3	99965.	TRA	2006-04-23	3	43761.	AIR TRIP TO USA
1	CL456	WILLIAMS	ALBERT	SUSTIN	USA	1	00000.		0001-01-01	1	00000.	
2	CL456	WILLIAMS	ALBERT	SUSTIN	USA	2	66125.	MOT	2006-03-16	2	00000.	
3	CL456	WILLIAMS	ALBERT	SUSTIN	USA	3	66125.	MOT	2006-03-16	3	29761.	AUTOMOBILE
1	CL993	RALESH	DAYLON	DSOUSA	IND	1	00000.		0001-01-01	1	00000.	
2	CL993	RALESH	DAYLON	DSOUSA	IND	2	71265.	MOT	2006-03-16	2	00000.	
3	CL993	RALESH	DAYLON	DSOUSA	IND	3	71265.	MOT	2006-03-16	3	29761.	BUS
1	CL883	RALESH	SOMA	SUBRAMANIAN	IND	1	00000.		0001-01-01	1	00000.	
2	CL883	RALESH	SOMA	SUBRAMANIAN	IND	2	71265.	MOT	2006-03-16	2	00000.	
3	CL883	RALESH	SOMA	SUBRAMANIAN	IND	3	71265.	MOT	2006-03-16	3	89761.	RICKSHAW

Task: Edit the Transformer stage

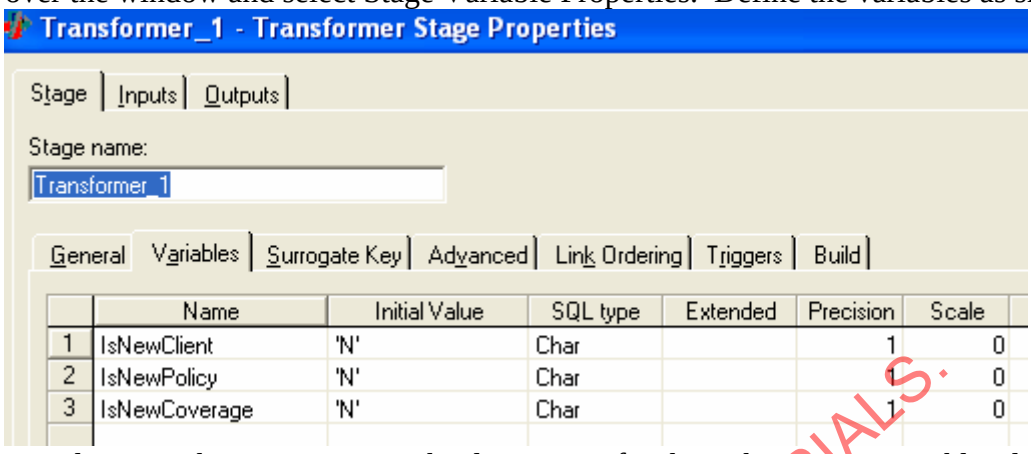
1. Save your job.
2. Open up the Transformer.

3. The CLIENT table output link should get all of the CLIENT record columns except for the RECTYPE field.
4. The POLICY table output link should get all of the POLICY columns except for the RECTYPE_2 field. In addition it should get the CLIENTID column from the CLIENT record.
5. The COVER table output link should get all of the COVERAGE record columns except for the RECTYPE_3 field. In addition it should get the CLIENTID column from the CLIENT record and POLICYID column from the POLICY record.
6. Add two additional columns to the POLICY output link. NUMCOVERAGES is an integer field that will contain the number of coverages in the policy. COVERAGES is a VarChar(500) field that will contain a list of all the coverages in the policy.



7. You need to know when a new client record is being processed, a new policy record is being processed, and a new coverage record is being processed. Create three stage

variables to track this. If necessary, click the Show/Hide stage variables icon at the top of the window to display the Stage Variables window. Then click the right mouse button over the window and select Stage Variable Properties. Define the variables as shown.



	Name	Initial Value	SQL type	Extended	Precision	Scale
1	IsNewClient	'N'	Char		1	0
2	IsNewPolicy	'N'	Char		1	0
3	IsNewCoverage	'N'	Char		1	0

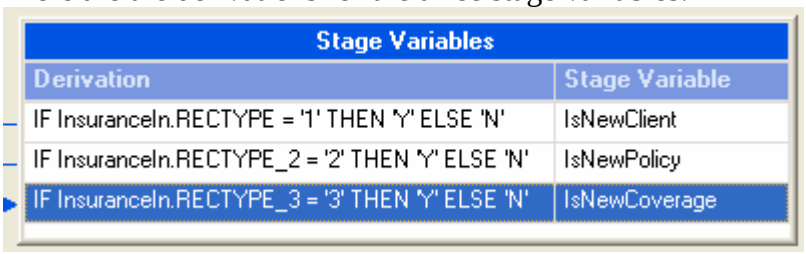
8. In order to see how to construct the derivations for these three stage variables, let's take a look at the data displayed by View Data. This shows how each of fields output from the source stage will be populated after each record is read. (Note: If the order of the rows

cffInsurance_seqOutput..Insurance.InsuranceIn - Data Browser

RECTYPE	CLIENTID	FIRSTNAME	MIDDLENAME	LASTNAME	COUNTRY	RECTYPE_2	POLICYID	PRODUCT	EFFDATE	RECTYPE_3	COVERID	DESCRIPTION
1	CL333	RALESH	ALAN	THOMPSON	USA		00000.		0001-01-01		00000.	
2	CL333	RALESH	ALAN	THOMPSON	USA	2	71265.	MOT	2006-03-16	2	00000.	
3	CL333	RALESH	ALAN	THOMPSON	USA	3	71265.	MOT	2006-03-16	3	29761.	AUTOMOBILE
3	CL333	RALESH	ALAN	THOMPSON	USA	3	71265.	MOT	2006-03-16	3	34761.	BICYCLE
2	CL333	RALESH	ALAN	THOMPSON	USA	2	99965.	TRA	2006-04-23	2	34761.	BICYCLE
3	CL333	RALESH	ALAN	THOMPSON	USA	3	99965.	TRA	2006-04-23	3	43761.	AIR TRIP TO USA
1	CL456	WILLIAMS	ALBERT	SUSTIN	USA	1	00000.		0001-01-01	1	00000.	
2	CL456	WILLIAMS	ALBERT	SUSTIN	USA	2	66125.	MOT	2006-03-16	2	00000.	
3	CL456	WILLIAMS	ALBERT	SUSTIN	USA	3	66125.	MOT	2006-03-16	3	29761.	AUTOMOBILE
1	CL993	RALESH	DAYLON	DSOUSA	IND	1	00000.		0001-01-01	1	00000.	
2	CL993	RALESH	DAYLON	DSOUSA	IND	2	71265.	MOT	2006-03-16	2	00000.	
3	CL993	RALESH	DAYLON	DSOUSA	IND	3	71265.	MOT	2006-03-16	3	29761.	BUS
1	CL883	RALESH	SOMA	SUBRAMANIAN	IND	1	00000.		0001-01-01	1	00000.	
2	CL883	RALESH	SOMA	SUBRAMANIAN	IND	2	71265.	MOT	2006-03-16	2	00000.	
3	CL883	RALESH	SOMA	SUBRAMANIAN	IND	3	71265.	MOT	2006-03-16	3	89761.	RICKSHAW

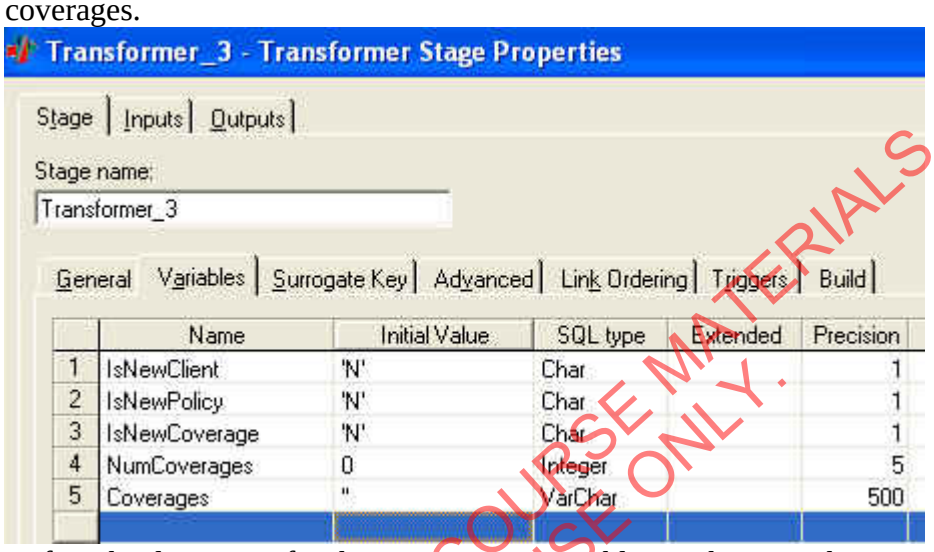
9. When you look at the data, keep in mind the following:
   a. Each read reads a single record. It will be of one of the three types.
   b. Each time a record is read, a record is output from the source stage and sent to the Transformer if the output constraint is satisfied. This record has columns coming from all three record types. Some of the columns may not contain valid information. For example, after the first record is read (which is a CLIENT record) all non-client record output columns contain "garbage", that is, non-valid data.
   c. When the RECTYPE field contains '1', the current record read from the source file is a CLIENT record. When RECTYPE_2 contains '2', the current record read from the source file is a POLICY record. When RECTYPE_3 contains '3', the current record read from the source file is a COVERAGE record.

10. Here are the derivations for the three stage variables.



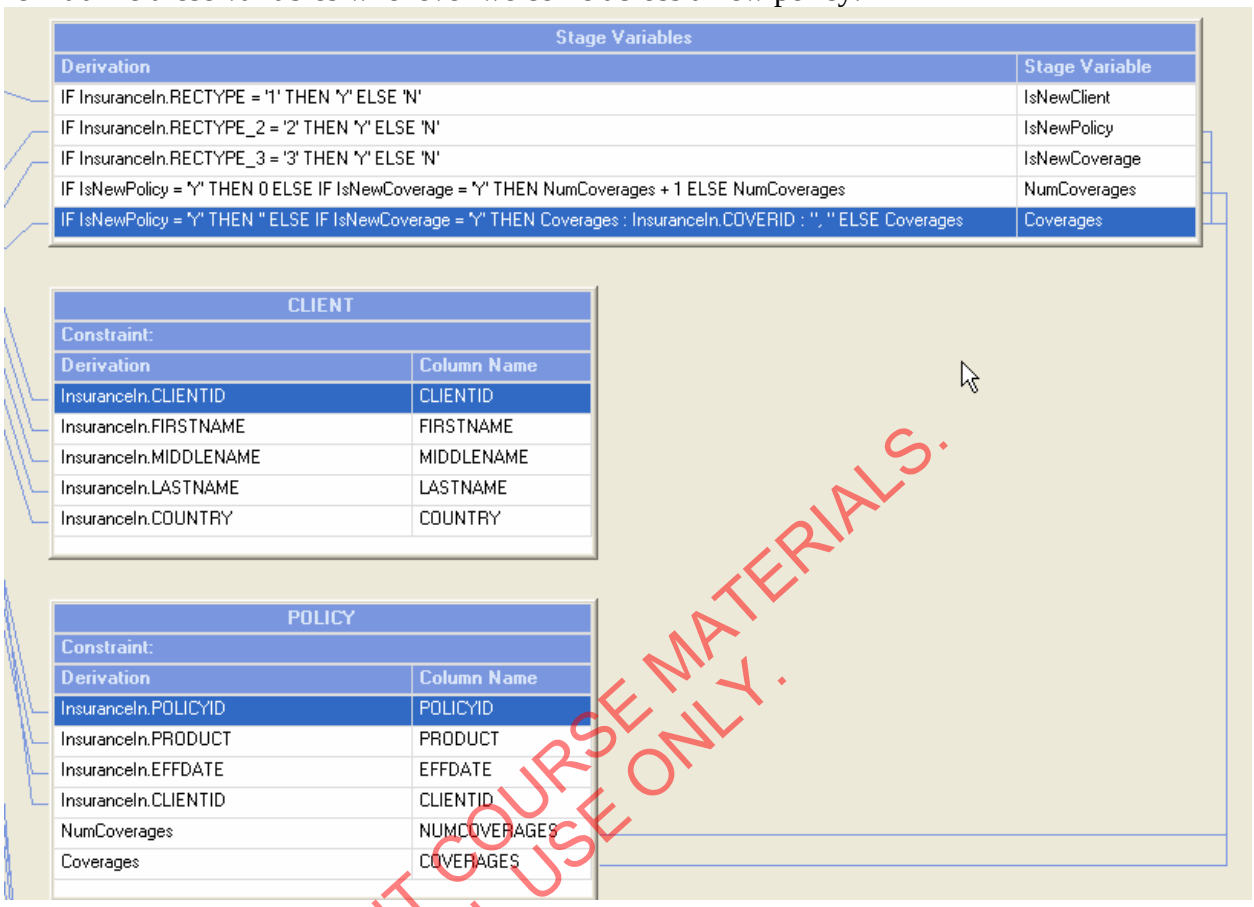
Stage Variables	
Derivation	Stage Variable
IF InsuranceIn.RECTYPE = '1' THEN 'Y' ELSE 'N'	IsNewClient
IF InsuranceIn.RECTYPE_2 = '2' THEN 'Y' ELSE 'N'	IsNewPolicy
IF InsuranceIn.RECTYPE_3 = '3' THEN 'Y' ELSE 'N'	IsNewCoverage

11. Define two more stage variables to track the number of coverages and the list of coverages.



12. Define the derivation for these new stage variables. Whenever there is a new coverage we want to add 1 to NumCoverages and add the Coverage ID to the list. We want to

reinitialize these variables whenever we come across a new policy.



Stage Variables

Derivation	Stage Variable
IF InsuranceIn.RECTYPE = '1' THEN 'Y' ELSE 'N'	IsNewClient
IF InsuranceIn.RECTYPE_2 = '2' THEN 'Y' ELSE 'N'	IsNewPolicy
IF InsuranceIn.RECTYPE_3 = '3' THEN 'Y' ELSE 'N'	IsNewCoverage
IF IsNewPolicy = 'Y' THEN 0 ELSE IF IsNewCoverage = 'Y' THEN NumCoverages + 1 ELSE NumCoverages	NumCoverages
IF IsNewPolicy = 'Y' THEN '' ELSE IF IsNewCoverage = 'Y' THEN Coverages : InsuranceIn.COVERID : ',' ELSE Coverages	Coverages

CLIENT

Constraint:

Derivation	Column Name
InsuranceIn.CLIENTID	CLIENTID
InsuranceIn.FIRSTNAME	FIRSTNAME
InsuranceIn.MIDDLENAME	MIDDLENAME
InsuranceIn.LASTNAME	LASTNAME
InsuranceIn.COUNTRY	COUNTRY

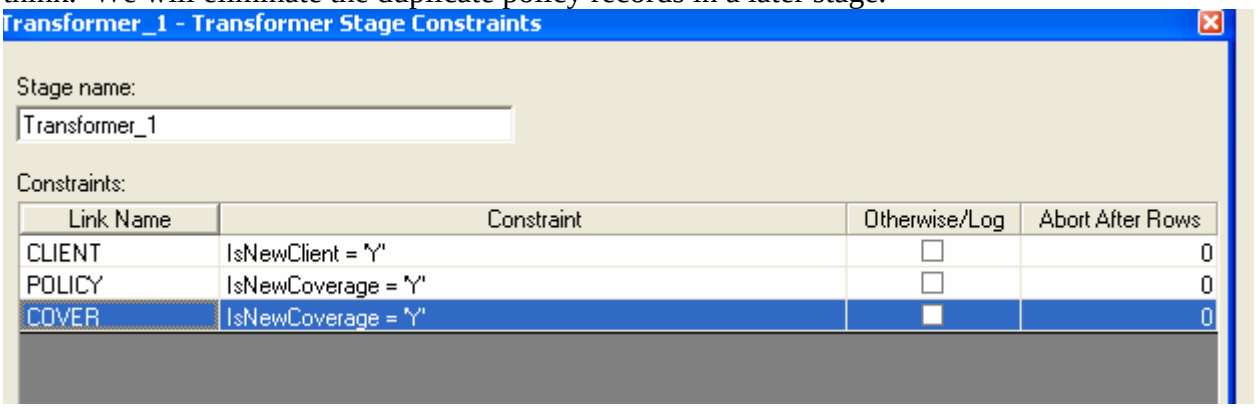
POLICY

Constraint:

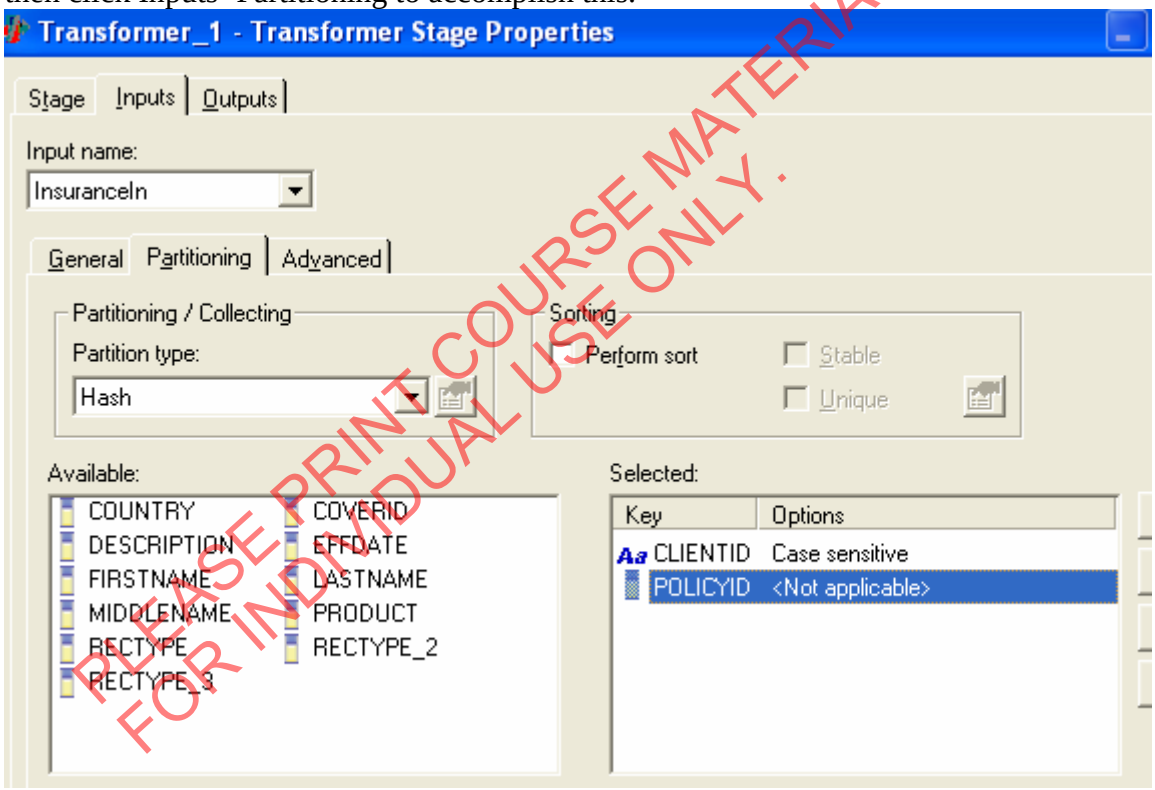
Derivation	Column Name
InsuranceIn.POLICYID	POLICYID
InsuranceIn.PRODUCT	PRODUCT
InsuranceIn.EFFDATE	EFFDATE
InsuranceIn.CLIENTID	CLIENTID
NumCoverages	NUMCOVERAGES
Coverages	COVERAGES

13. Map the NumCoverages and Coverages stage variables, respectively, to the NumCoverages and Coverages output columns.
14. Now, write the constraints. POLICY is tricky because you cannot write the record until you have processed all the coverage records in the policy. If we come across another new policy then we know we have processed all the records in the previous policy. However, this doesn't work when we are at the end of the file. To handle this difficulty we will output a record every time we process a coverage record. That is, our constraint for POLICY is IsNewCoverage = 'Y', not IsNewPolicy = 'Y', as you might initially

think. We will eliminate the duplicate policy records in a later stage.



15. To insure that all client policies are in the same input link, we hash partition by CLIENTID and POLICYID. Click the icon on the top far left of the Transformer and then click Inputs>Partitioning to accomplish this.



16. Define the three output Sequential File stages. Write to output files named ClientOut.txt, PolicyOut.txt, and CoverageOut.txt, respectively.
17. Compile and run your job. Note that we have not removed the duplicate records yet. We are just testing things so far.

18. Verify the data in the CLIENT file.

cffInsurance_..Client.CLIENT - Data Browser

CLIENTID	FIRSTNAME	MIDDLENAME	LASTNAME	COUNTRY
CL333	RALESH	ALAN	THOMPSON	USA
CL456	WILLIAMS	ALBERT	SUSTIN	USA
CL993	RALESH	DAYLON	DSOUSA	IND
CL883	RALESH	SOMA	SUBRAMANIAN	IND

19. Verify the data in the POLICY file.

cffInsurance_..Policy.POLICY - Data Browser

POLICYID	PRODUCT	EFFDATE	CLIENTID	NUMCOVERAGES	COVERAGES
71265.	MOT	2006-03-16	CL333	1	29761.,
71265.	MOT	2006-03-16	CL333	2	29761., 34761.,
99965.	TRA	2006-04-23	CL333	1	43761.,
66125.	MOT	2006-03-16	CL456	1	29761.,
71268.	MOT	2006-03-16	CL993	1	29761.,
71279.	MOT	2006-03-16	CL883	1	89761.,

20. Verify the data in the COVERAGE file.

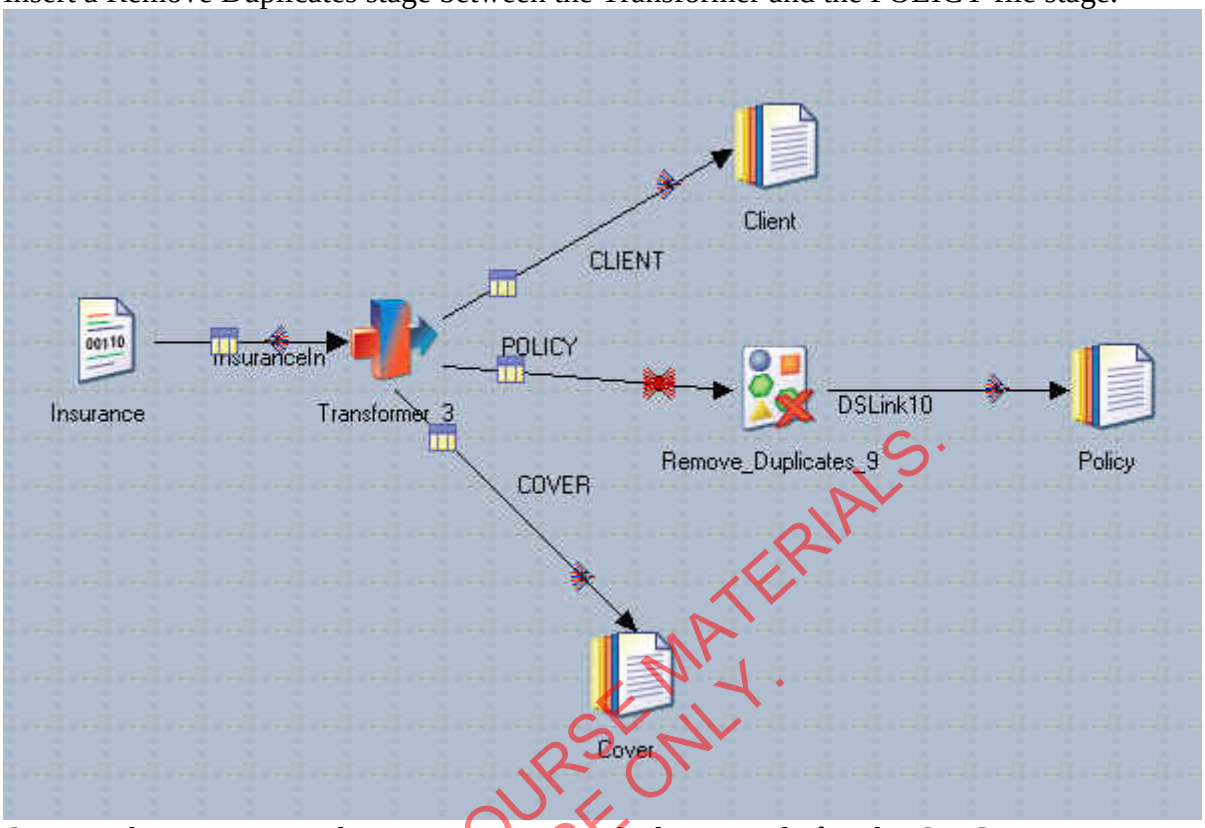
cffInsurance_..Cover.COVER - Data Browser

COVERID	DESCRIPTION	CLIENTID	POLICYID
29761.	AUTOMOBILE	CL333	71265.
34761.	BICYCLE	CL333	71265.
43761.	AIR TRIP TO USA	CL333	99965.
29761.	AUTOMOBILE	CL456	66125.
29761.	BUS	CL993	71268.
89761.	RICKSHAW	CL883	71279.

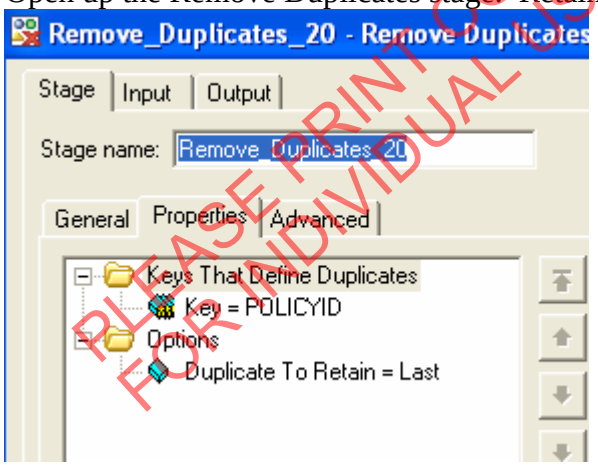
Task: Remove the extra policies

1. Save your job as cffInsurance2.

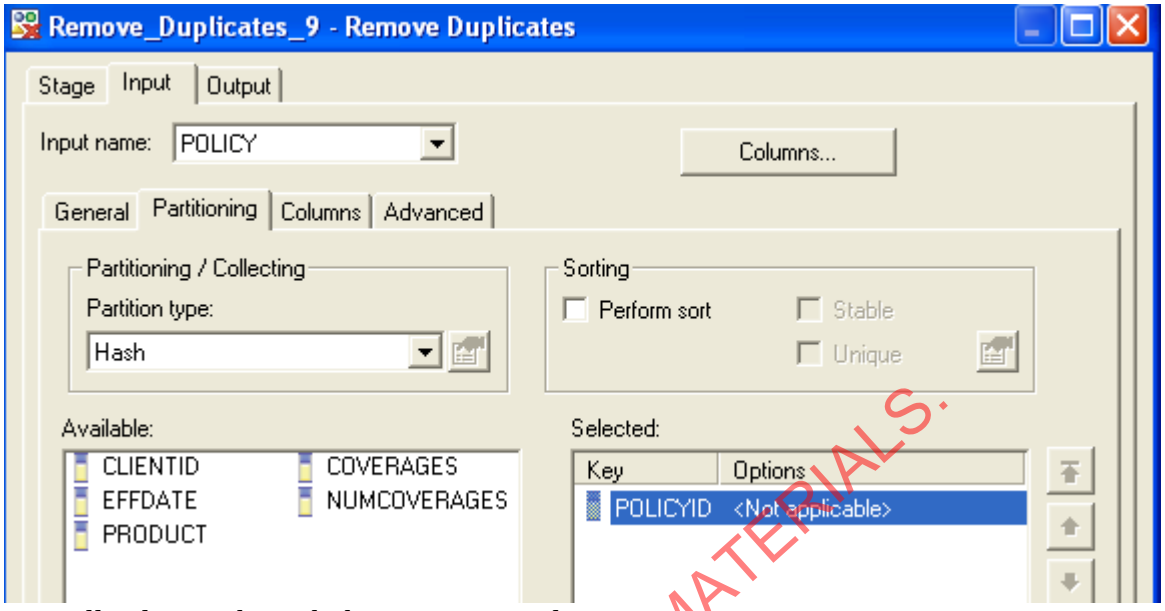
2. Insert a Remove Duplicates stage between the Transformer and the POLICY file stage.



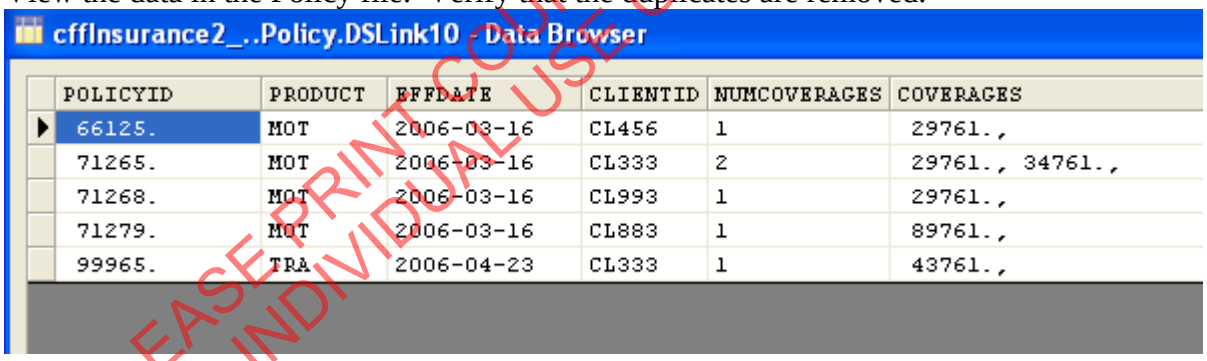
3. Open up the Remove Duplicates stage. Retain the last record of each POLICYID group.



4. Specify that the data going into the Remove Duplicates stage is to be hash partitioned by POLICYID.



5. Map all columns through the Remove Duplicates stage.
6. Edit the target POLICY sequential stage as needed.
7. Compile and run.
8. View the data in the Policy file. Verify that the duplicates are removed.



POLICYID	PRODUCT	EFFDATE	CLIENTID	NUMCOVERAGES	COVERAGES
66125.	MOT	2006-03-16	CL456	1	29761.,
71265.	MOT	2006-03-16	CL333	2	29761., 34761.,
71268.	MOT	2006-03-16	CL993	1	29761.,
71279.	MOT	2006-03-16	CL883	1	89761.,
99965.	TRA	2006-04-23	CL333	1	43761.,

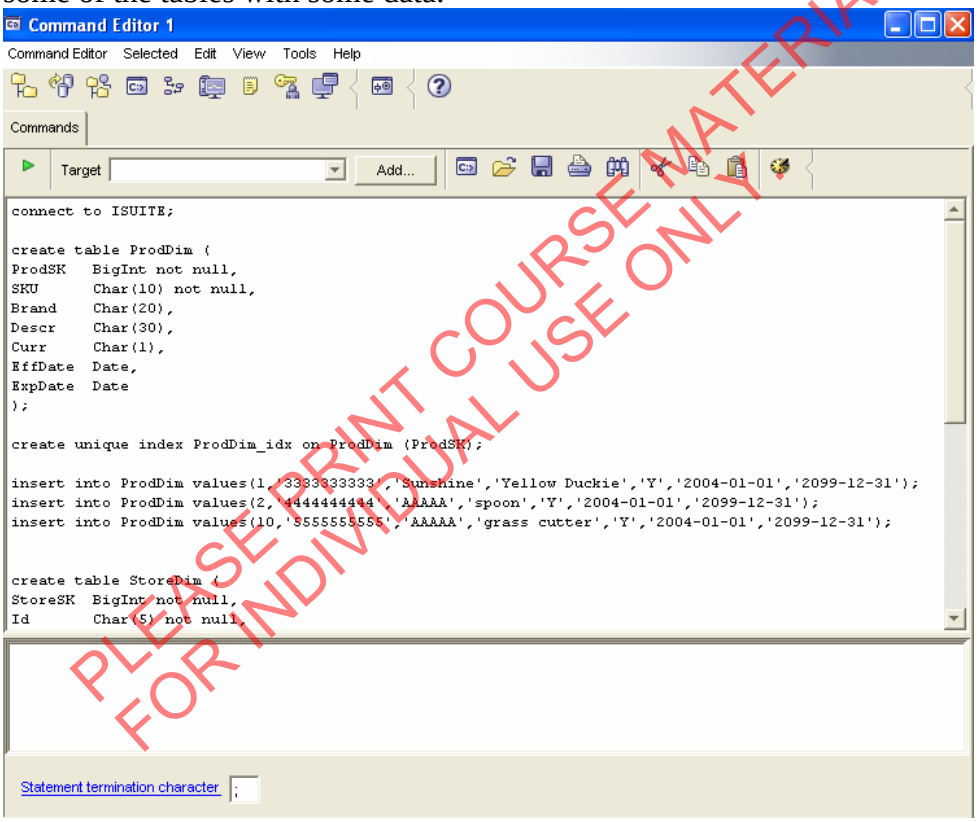
Special Topic 3: Slowly Changing Dimensions

Assumptions

- The CreateSCDTables.sql file exists in the ISFiles>SCD directory.
- You have access to a DB2 database named ISUITE.

Task: Create the start schema tables

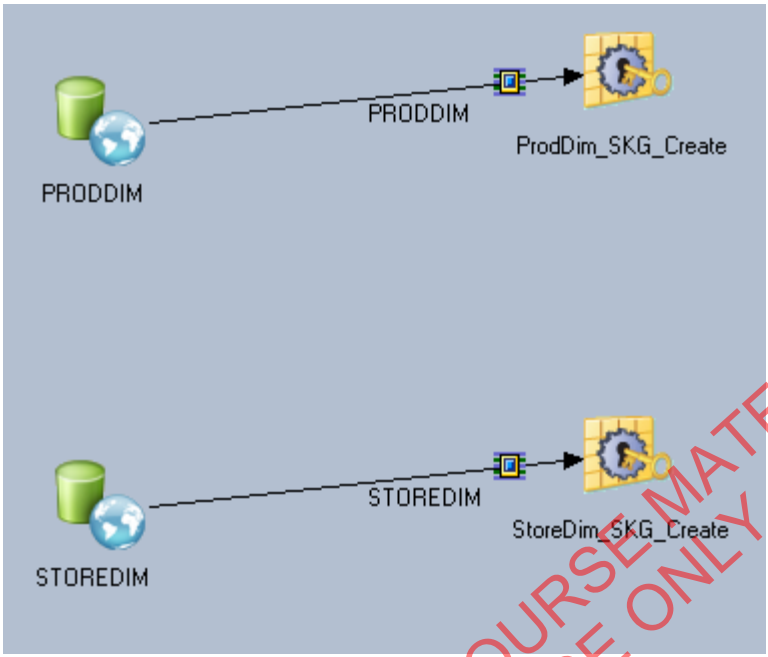
1. Open the DB2 Control Center.
2. Click on the Command Editor icon to open the DB2 Command Line Processor (CLP).
3. Click Selected>Open. Select the CreateSCDTables.sql file in the ISFiles>SCD directory and then click OK. This loads the SQL statements in the file into the window. These statements connect to the ISUITE database, create the dimension and fact tables, and load some of the tables with some data.



4. Click the Execute button. Check the output window to verify that all commands executed successfully.
5. In the Control Center window, refresh and then view the existence of the dimension and fact tables (SUPER.FACTTBL, SUPER.PRODDIM, SUPER.STOREDIM). Verify that the PRODDIM and STOREDIM tables have a few rows of data in them.

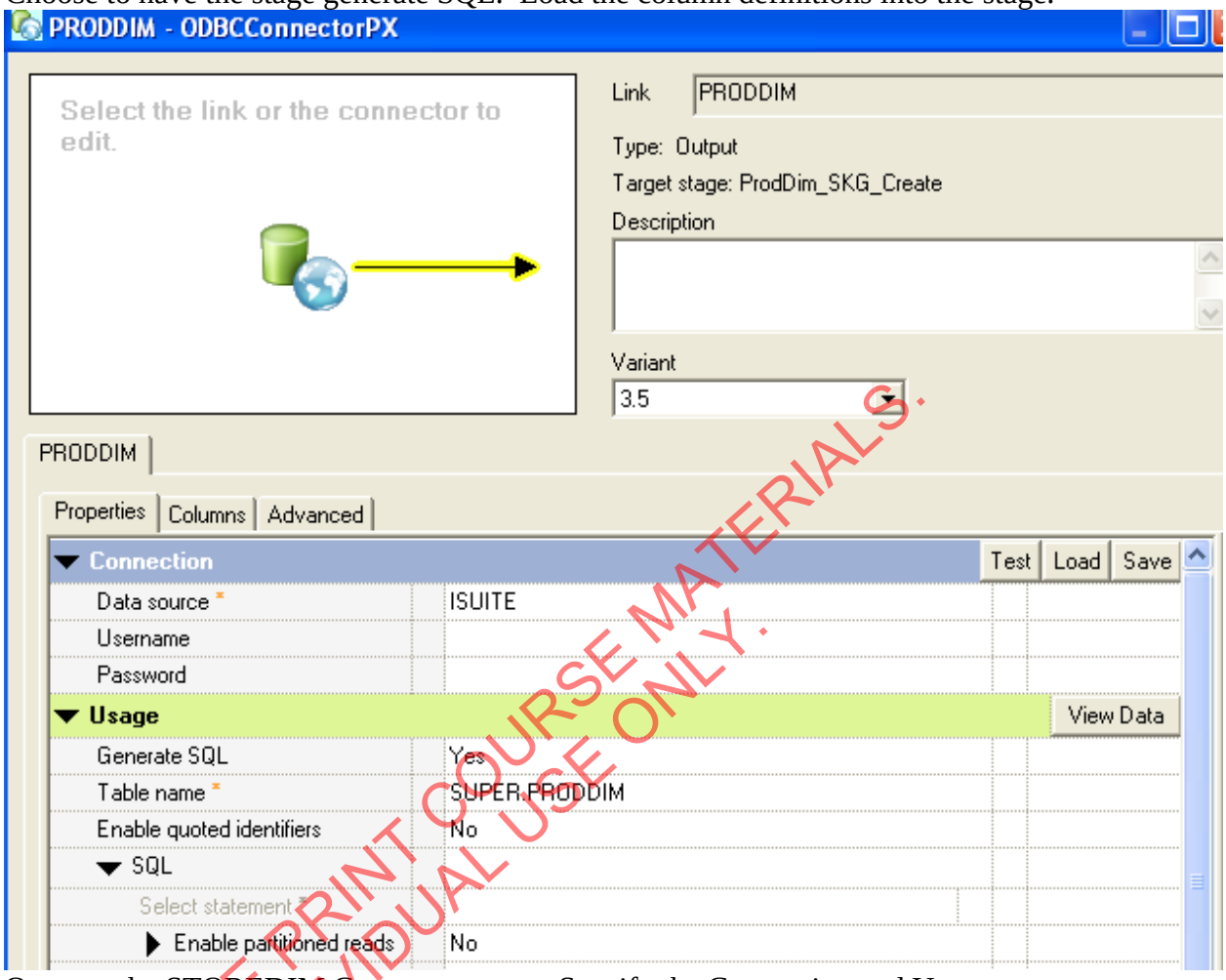
Task: Create the surrogate key source files

1. In DataStage Designer, create a new parallel job named scdCreateSurrogateKeySourceFiles. Save it into your _Training>Jobs folder.
2. From the Processing folder add two Surrogate Key Generator stages to the canvas. Name them as shown.



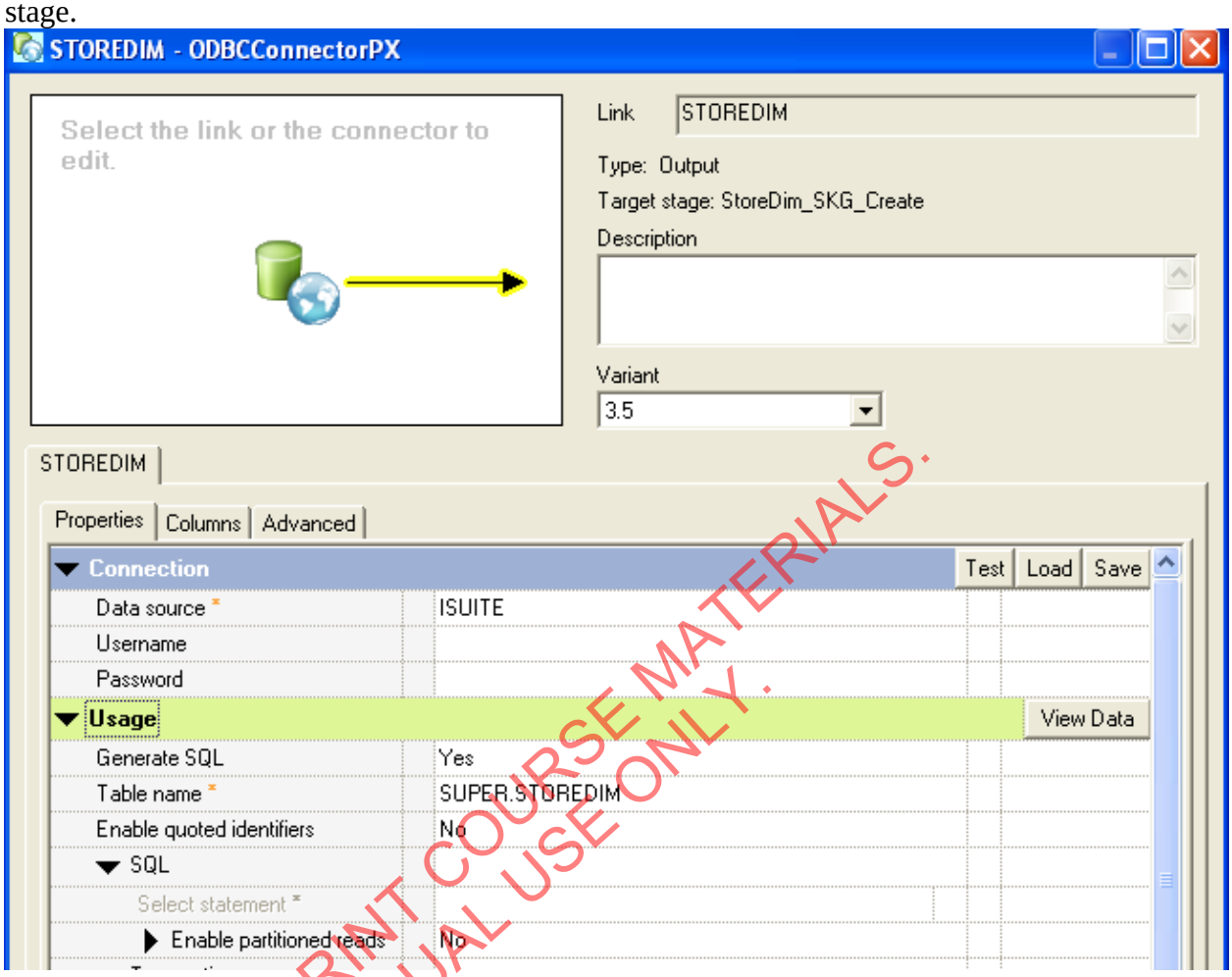
1. Click Import>Table Definitions>Orchestrate Schema Definitions. Import the column definitions of the three star schema tables created in the ISUITE database, namely, SUPER.PRODDIM, SUPER.STOREDIM, and SUPER.FACTTBL. Save the Table Definitions in your _Training>Metadata folder.
2. Edit the Locator tab of each of these table definitions. Make sure the information is accurate and complete.

3. Open up the PRODDIM Connector stage. Specify the Connection and Usage properties. Choose to have the stage generate SQL. Load the column definitions into the stage.

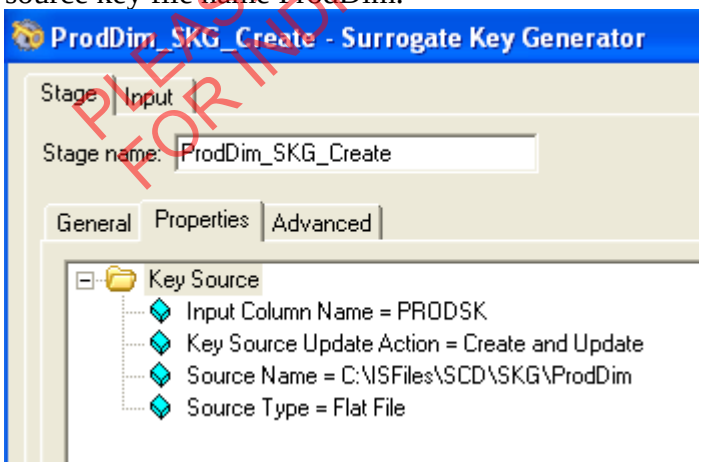


4. Open up the STOREDIM Connector stage. Specify the Connection and Usage properties. Choose to have the stage generate SQL. Load the column definitions into the

stage.

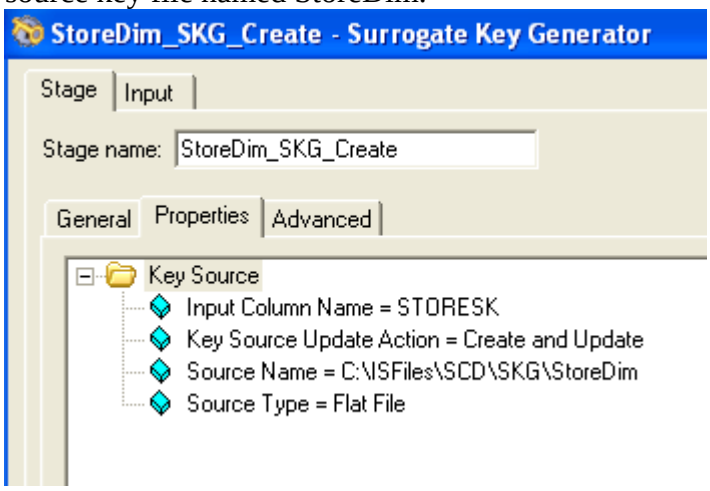


5. Open the ProdDim_SKG_Update stage properties. The Key Source Update Action is Create and Update. Select PRODSK for the input column name. Specify a path to a source key file name ProdDim.



6. Open the StoreDim_SKG_Update stage properties. The Key Source Update Action is Create and Update. Select STORESK for the input column name. Specify a path to a

source key file named StoreDim.

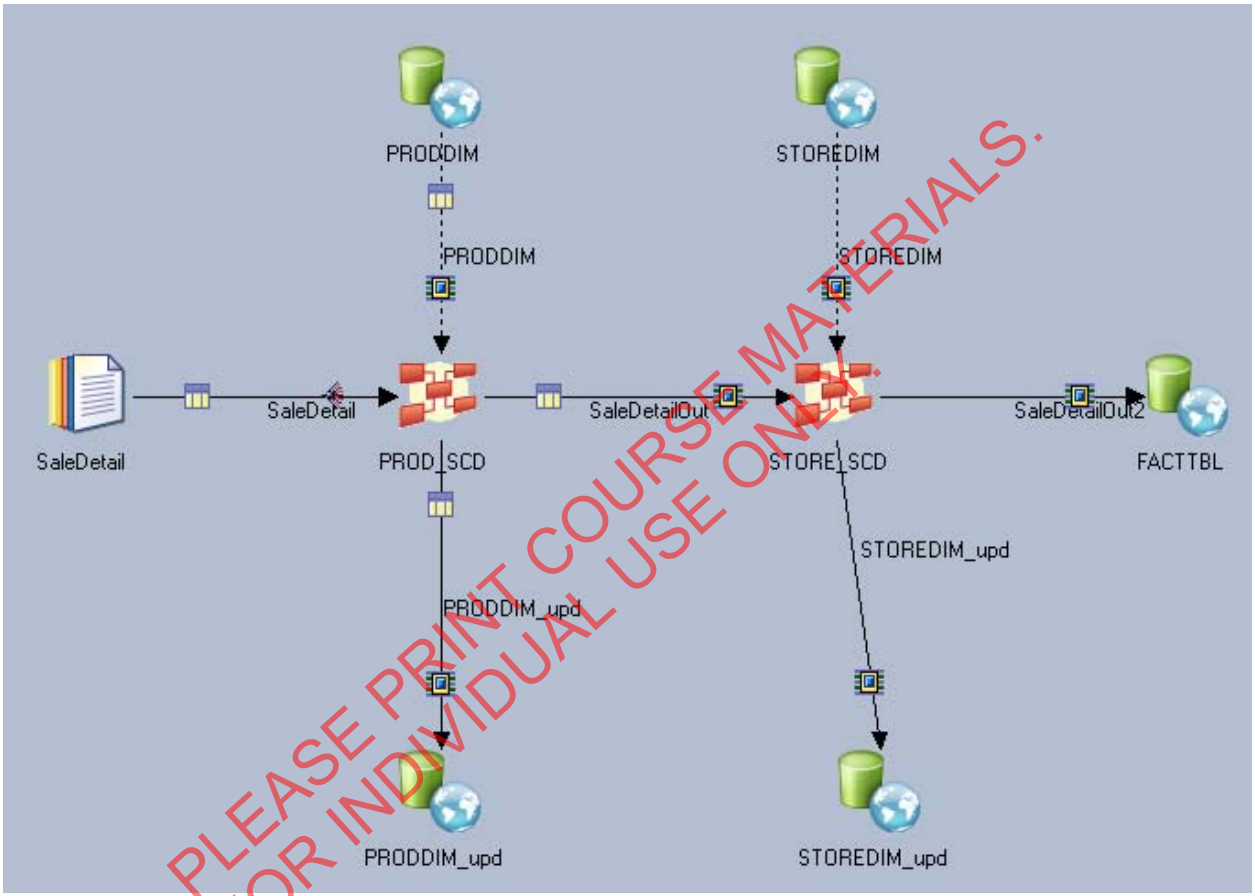


7. Compile and run your job. Check the job log.
8. Verify that the files have been created in the ISFiles>SCD>SKG directory and that they are not empty.

Build an SCD job with two dimensions

Introduction

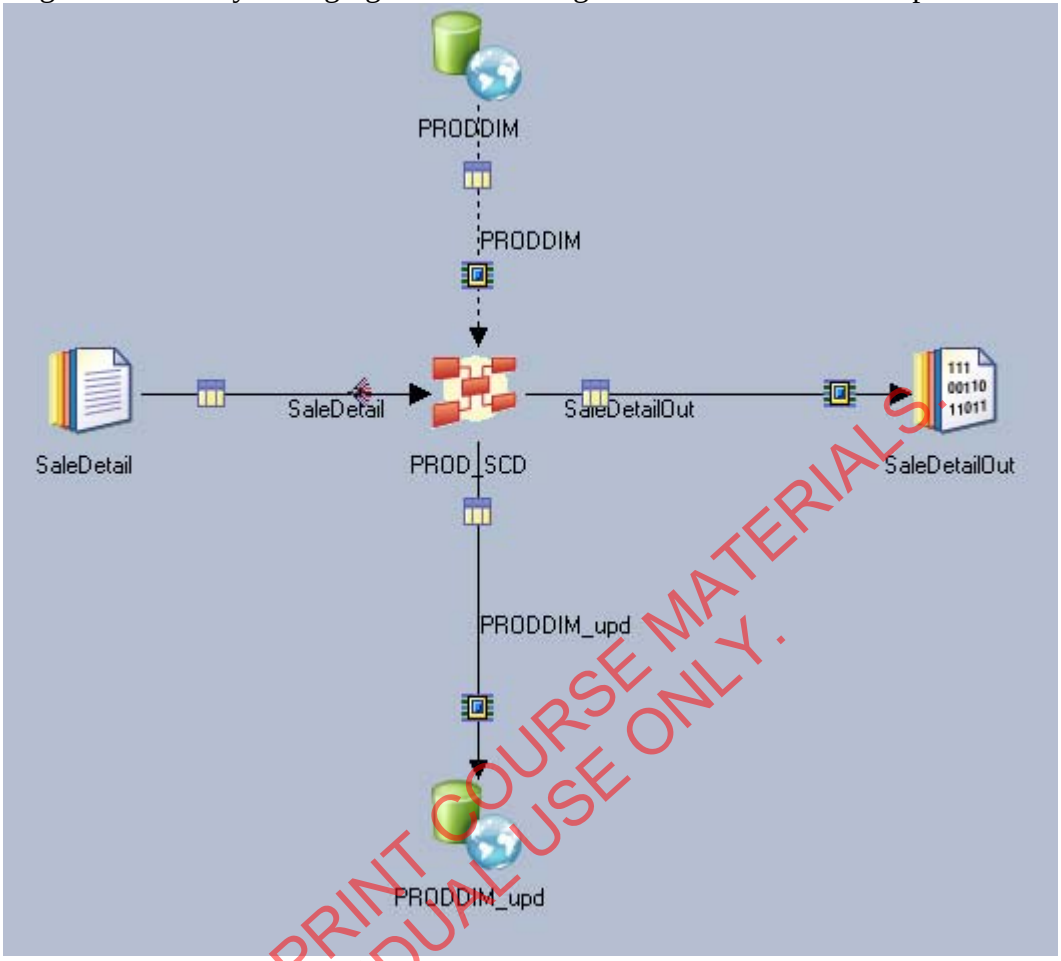
In this section you will build and SCD with two dimensions. The completed job will look like the following. However, to ease the development and debugging two separate jobs will be built. The first will process the PRODDIM dimension table and write its output to a dataset. The second will read the data from the dataset, process the STOREDIM dimension table, and write the results to the fact table.



Task: Build an SCD job to process the first dimension

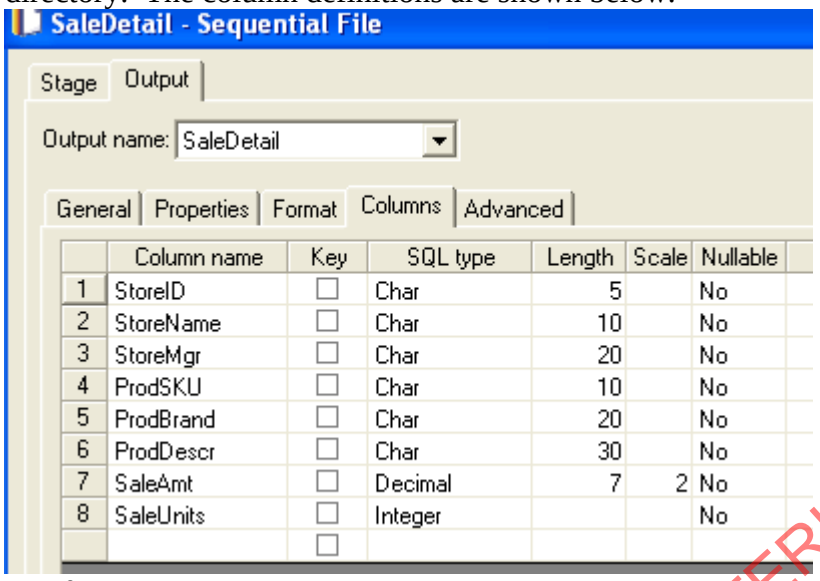
1. Create a new parallel job named scdLoadFactTable_1.
2. ***Important*** Open the Job Properties window and make sure that Runtime Column Propogation is not enabled. Otherwise, you will get runtime errors when source columns such as StoreID are written to the PRODDIM_upd link.

3. Add the stages and links as shown. Notice that the link from the PRODDIM Connector stage to the Slowly Changing Dimension stage in the middle is a lookup reference link.



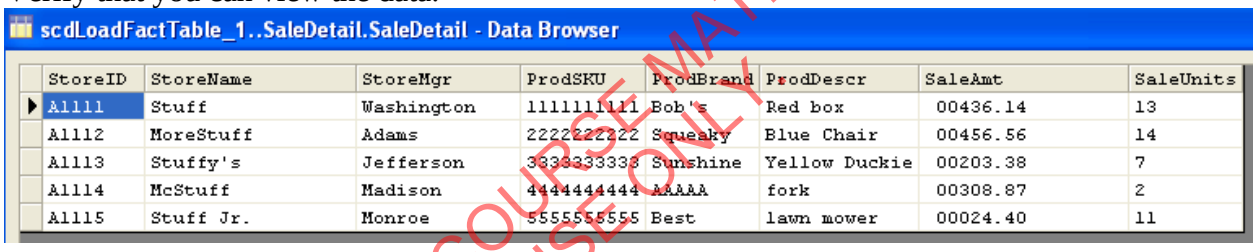
4. Edit the SaleDetail stage. Extract data from the SaleDetail.txt file in the ISFiles>SCD directory. Import the table definition named SaleDetailTableDef.dsx in the ISFiles>SCD

directory. The column definitions are shown below.



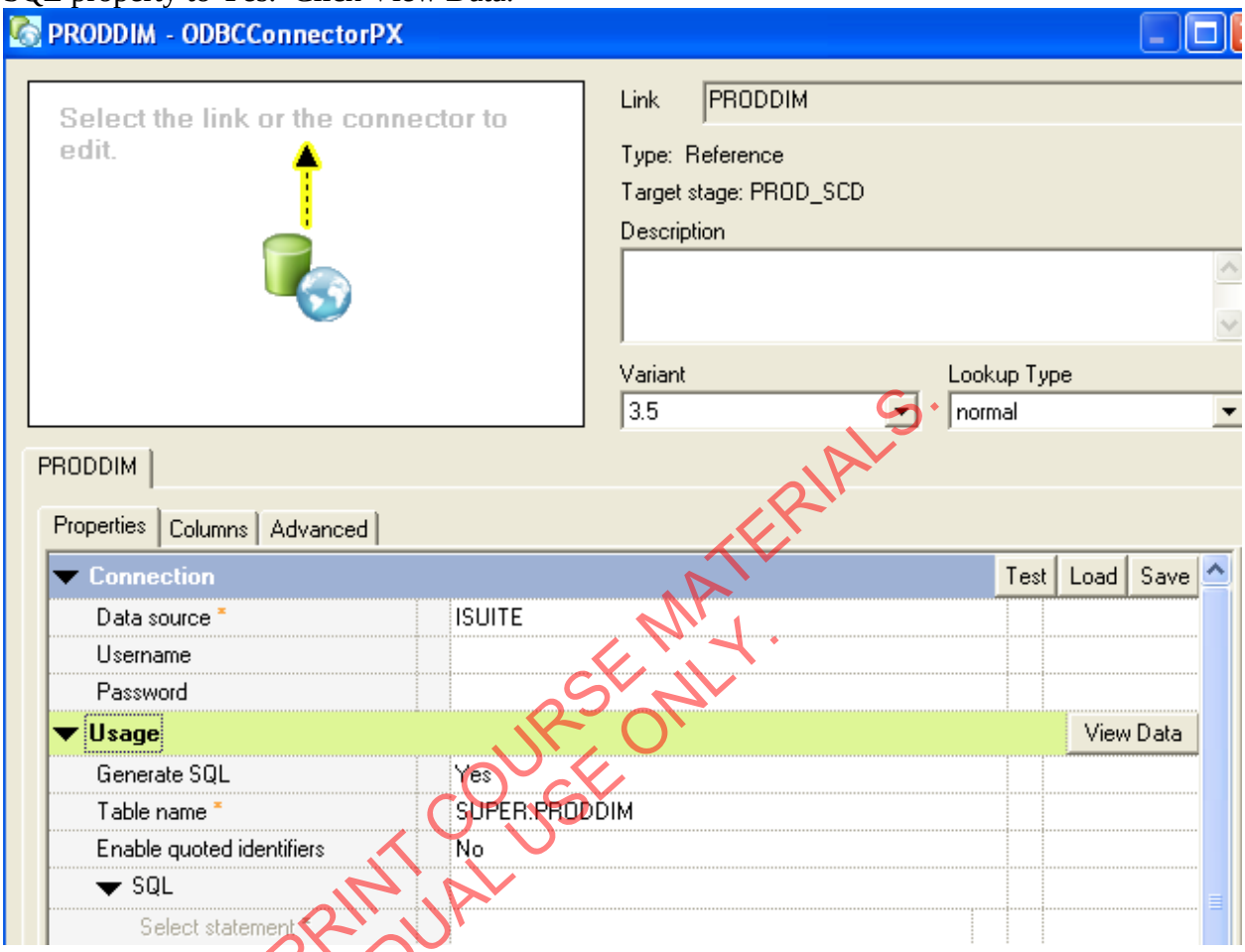
	Column name	Key	SQL type	Length	Scale	Nullable
1	StoreID	☐	Char	5		No
2	StoreName	☐	Char	10		No
3	StoreMgr	☐	Char	20		No
4	ProdSKU	☐	Char	10		No
5	ProdBrand	☐	Char	20		No
6	ProdDescr	☐	Char	30		No
7	SaleAmt	☐	Decimal	7	2	No
8	SaleUnits	☐	Integer			No
		☐				

5. Verify that you can view the data.

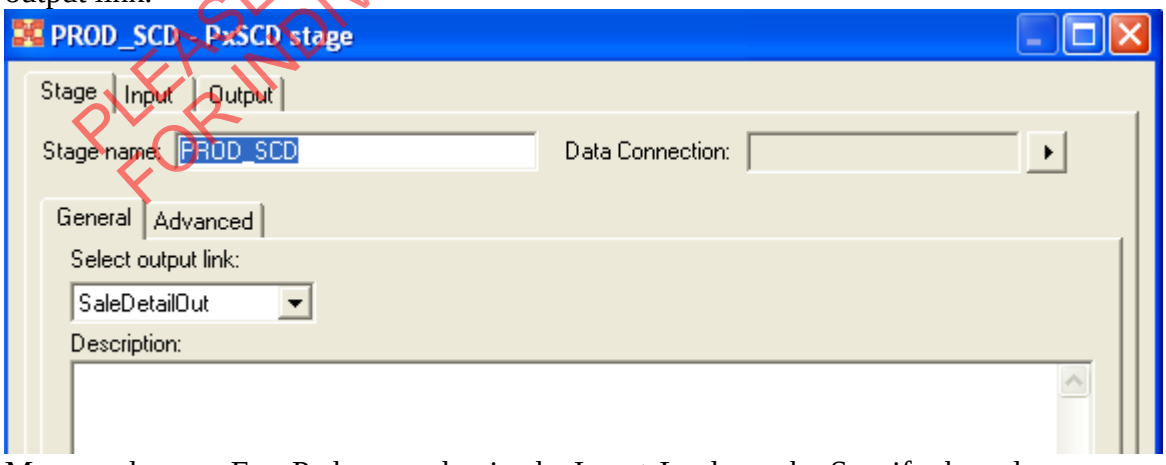


StoreID	StoreName	StoreMgr	ProdSKU	ProdBrand	ProdDescr	SaleAmt	SaleUnits
Allll	Stuff	Washington	lllllllll1	Bob's	Red box	00436.14	13
Alll2	MoreStuff	Adams	2222222222	Squeaky	Blue Chair	00456.56	14
Alll3	Stuffy's	Jefferson	3333333333	Sunshine	Yellow Duckie	00203.38	7
Alll4	McStuff	Madison	4444444444	AAAAA	fork	00308.87	2
Alll5	Stuff Jr.	Monroe	5555555555	Best	lawn mower	00024.40	11

6. Edit the PRODDIM reference link stage. Load column definitions. Set the Generate SQL property to Yes. Click View Data.

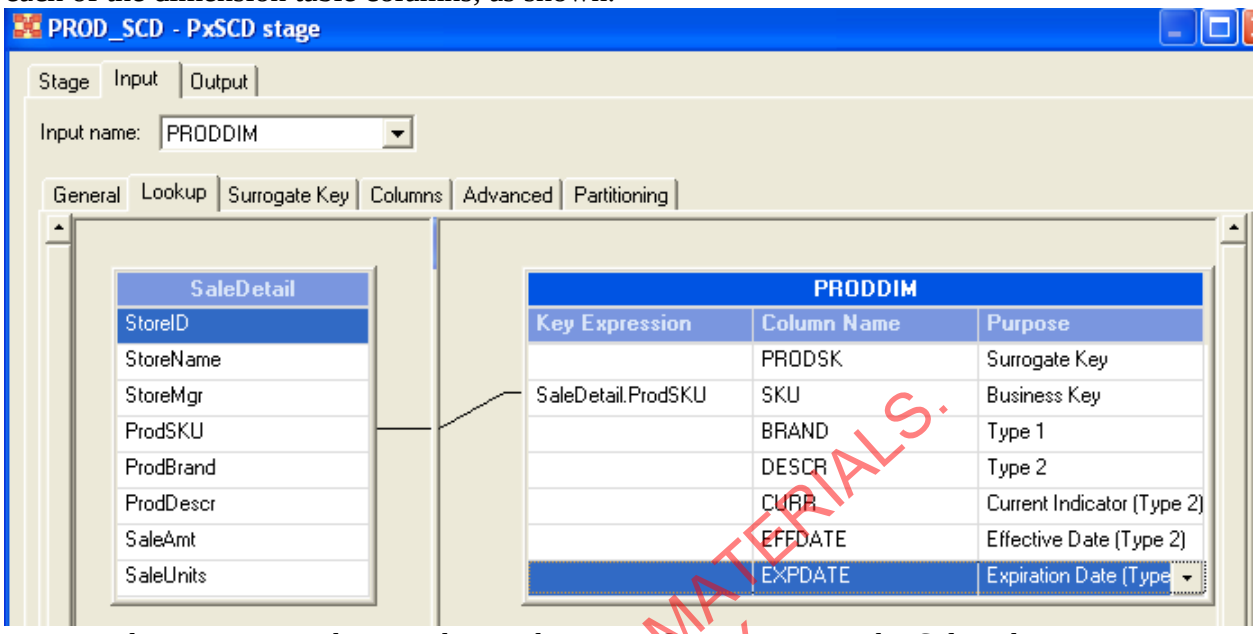


7. Open the PROD_SCD stage. On the Stage>General tab, select SaleDetailOut as the output link.

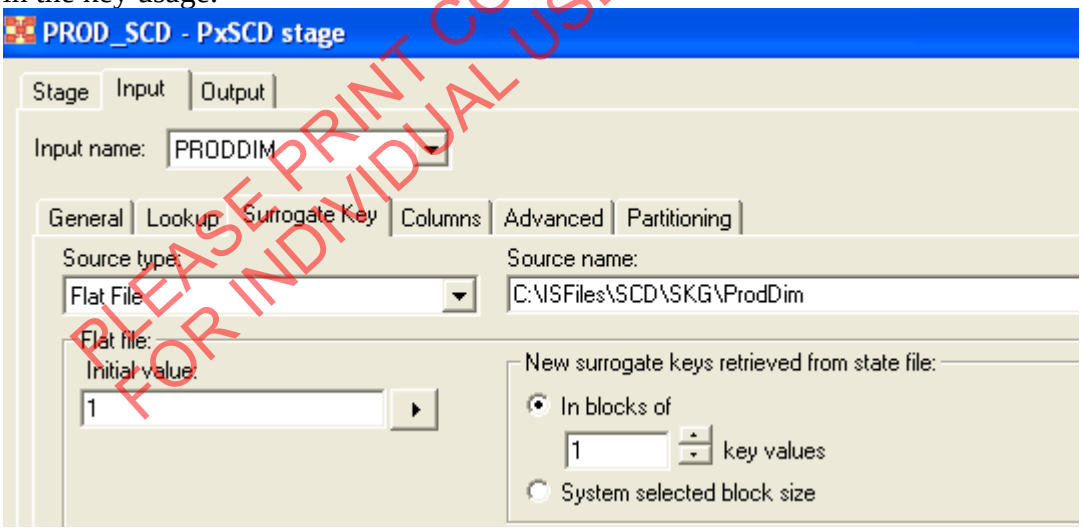


8. Move to the next Fast Path page, that is, the Input>Lookup tab. Specify the column matching to use to lookup a matching dimension row. Here we want to retrieve the row

with the matching PRODDIM business (natural) key. Also select the purpose codes for each of the dimension table columns, as shown.

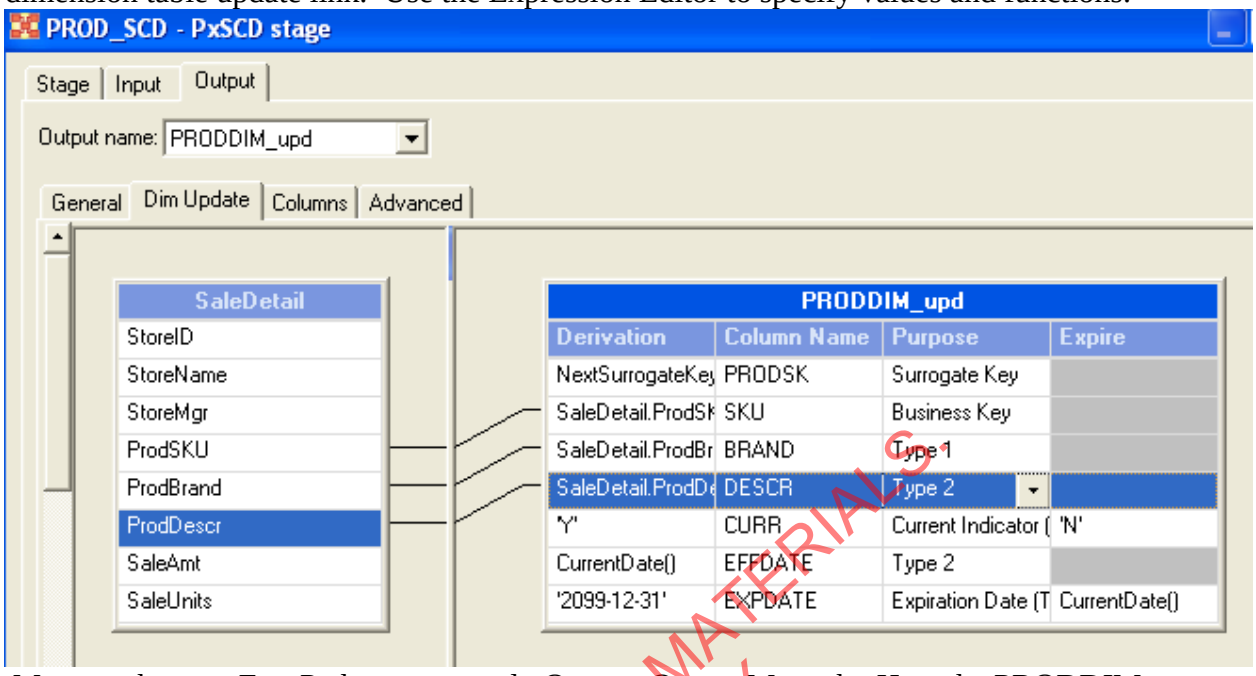


9. Move to the next Fast Path page, that is, the Input>Surrogate Key tab. Select the surrogate key source file (ProdDim) from the SKG directory. Specify the surrogate key initial value, 1. Also specify how many surrogate key values to retrieve from the state file in a single block read. Specifying a block size of 1 ensures that there will be no gaps in the key usage.

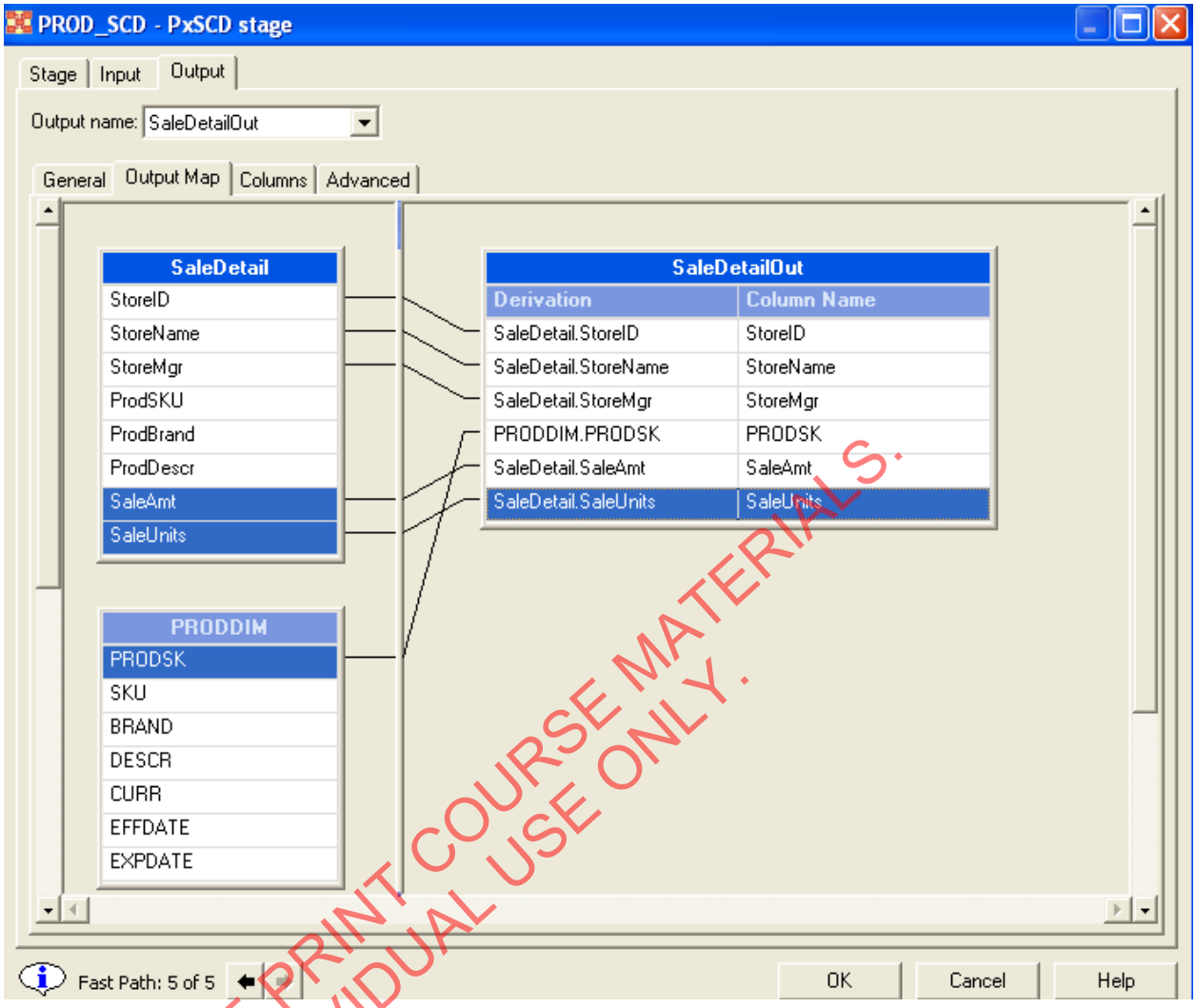


10. Move to the next Fast Path page, that is, the Output>Dim Update tab. Here specify how to create a new dimension record and how to expire a dimension record that has Type 2 columns in it. Be sure Output name is PRODDIM_Upd, that is, the name of the

dimension table update link. Use the Expression Editor to specify values and functions.

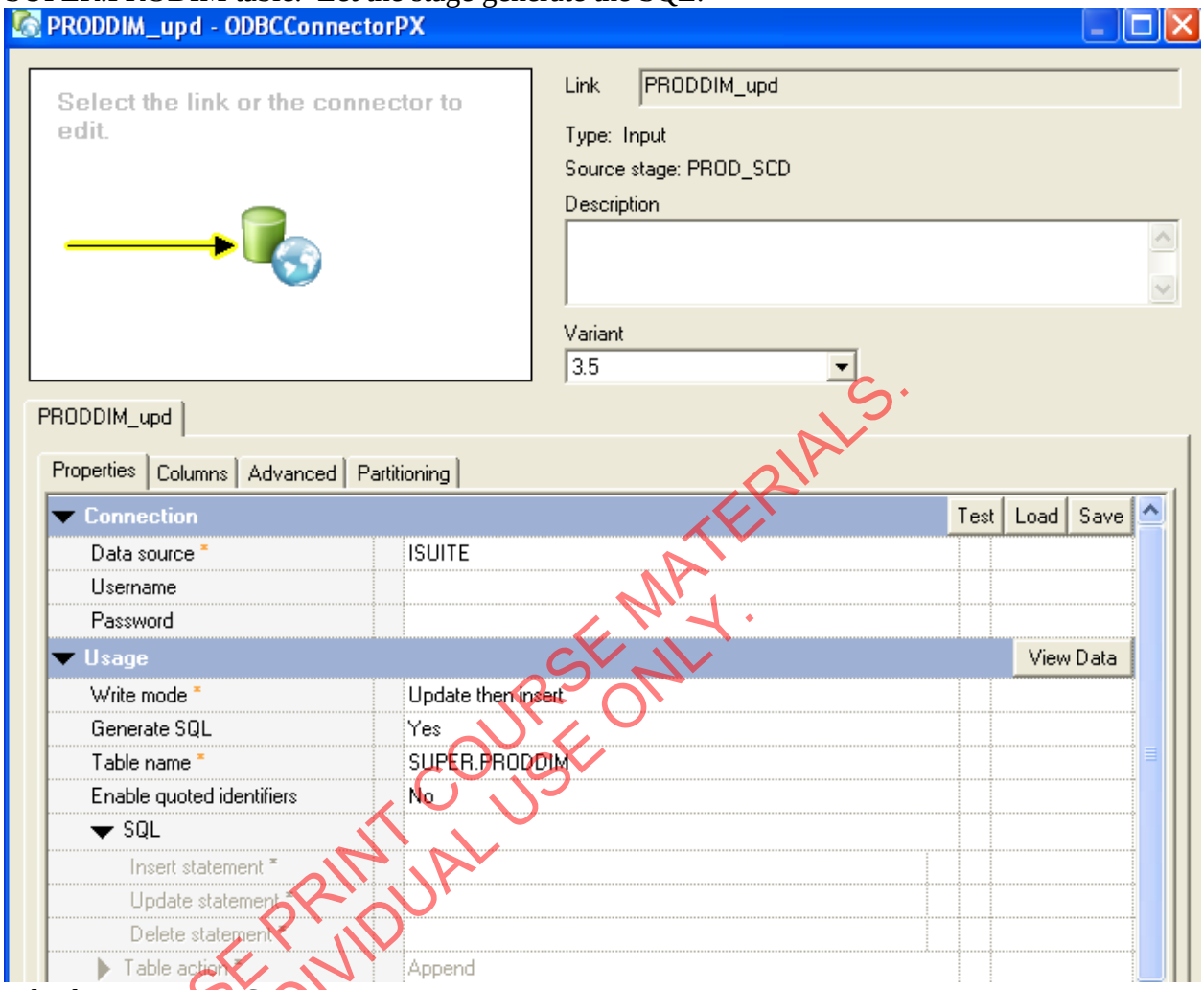


11. Move to the next Fast Path page, namely Output>Output Map tab. Here the PRODDIM surrogate key field (PRODSK) replaces the business key field in the source file.

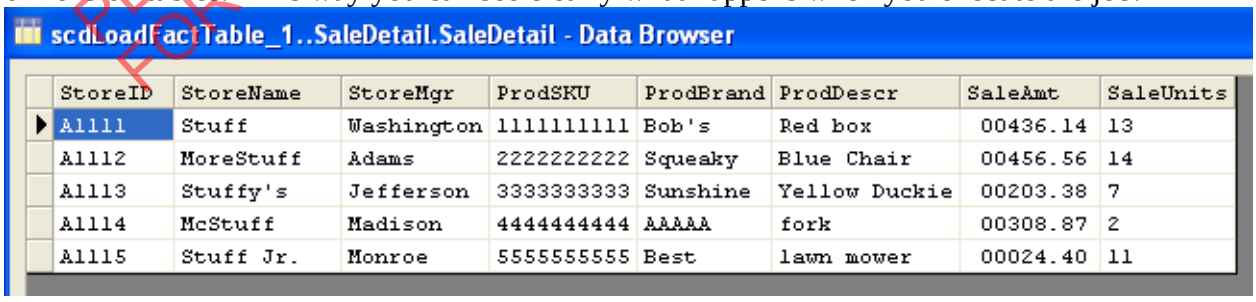


12. Click OK to close the SCD stage.

13. Open up the PRODDIM_upd stage. Use Update then Insert to write to the target SUPER.PRODIM table. Let the stage generate the SQL.



14. Edit the target DataSet stage.
15. Compile. Before you run the job, view the data from the SaleDetail.txt file and the two dimension tables. This way you can see clearly what happens when you execute the job.



StoreID	StoreName	StoreMgr	ProdSKU	ProdBrand	ProdDescr	SaleAmt	SaleUnits
Allll	Stuff	Washington	lllllllll	Bob's	Red box	00436.14	13
Alll2	MoreStuff	Adams	2222222222	Squeaky	Blue Chair	00456.56	14
Alll3	Stuffy's	Jefferson	3333333333	Sunshine	Yellow Duckie	00203.38	7
Alll4	McStuff	Madison	4444444444	AAAAA	fork	00308.87	2
Alll5	Stuff Jr.	Monroe	5555555555	Best	lawn mower	00024.40	11

Open Table - PRODDIM

HAWKVM - DB2 - ISUITE - SUPER.PRODDIM

Edits to these results are performed as searched UPDATEs and DELETEs. Use the Tools Settings notebook to change the form of editing.

PRODSK	SKU	BRAND	DESCR	CURR	EFFDATE	EXPDATE
1	3333333333	Sunshine	Yellow Duckie	Y	Jan 1, 2004	Dec 31, 2099
2	4444444444	AAAAA	spoon	Y	Jan 1, 2004	Dec 31, 2099
10	5555555555	AAAAA	grass cutter	Y	Jan 1, 2004	Dec 31, 2099

16. Run the job. Check the job log for errors. View the data in PRODIM to see if the table was updated properly. SKU 3... doesn't change. SKU 1 and 2 are new inserts. SKU 4 and 5 new Type 2 updates. The original records are preserved as historical records (CURR=N) PRODSK=2 and 10 are kept as historical records. (***Note***: There appears to be a bug or user error in my job. The two historical rows, PRODSK=2 and 10, lose their original SKU, DESCR, and EFFDATE values.)

Open Table - PRODDIM

HAWKVM - DB2 - ISUITE - SUPER.PRODDIM

Edits to these results are performed as searched UPDATEs and DELETEs. Use the Tools Settings notebook to change the form of editing.

Add Row

Delete Row

PRODSK	SKU	BRAND	DESCR	CURR	EFFDATE	EXPDATE
1	3333333333	Sunshine	Yellow Duckie	Y	Jan 1, 2004	Dec 31, 2099
5	4444444444	AAAAA	fork	Y	Dec 8, 2006	Dec 31, 2099
6	5555555555	Best	lawn mower	Y	Dec 8, 2006	Dec 31, 2099
3	1111111111	Bob's	Red box	Y	Dec 8, 2006	Dec 31, 2099
4	2222222222	Squeaky	Blue Chair	Y	Dec 8, 2006	Dec 31, 2099
2		AAAAA		N		Dec 8, 2006
10		AAAAA		N		Dec 8, 2006

17. View the data in the target dataset. This appears correct. A1111 and A1112 are assigned new surrogate key values since they are inserts. A1113 was not changed, so it has the same surrogate key value. A1114 and A1115 are new Type 2 updates. They received new surrogate key values and are inserted into the target.

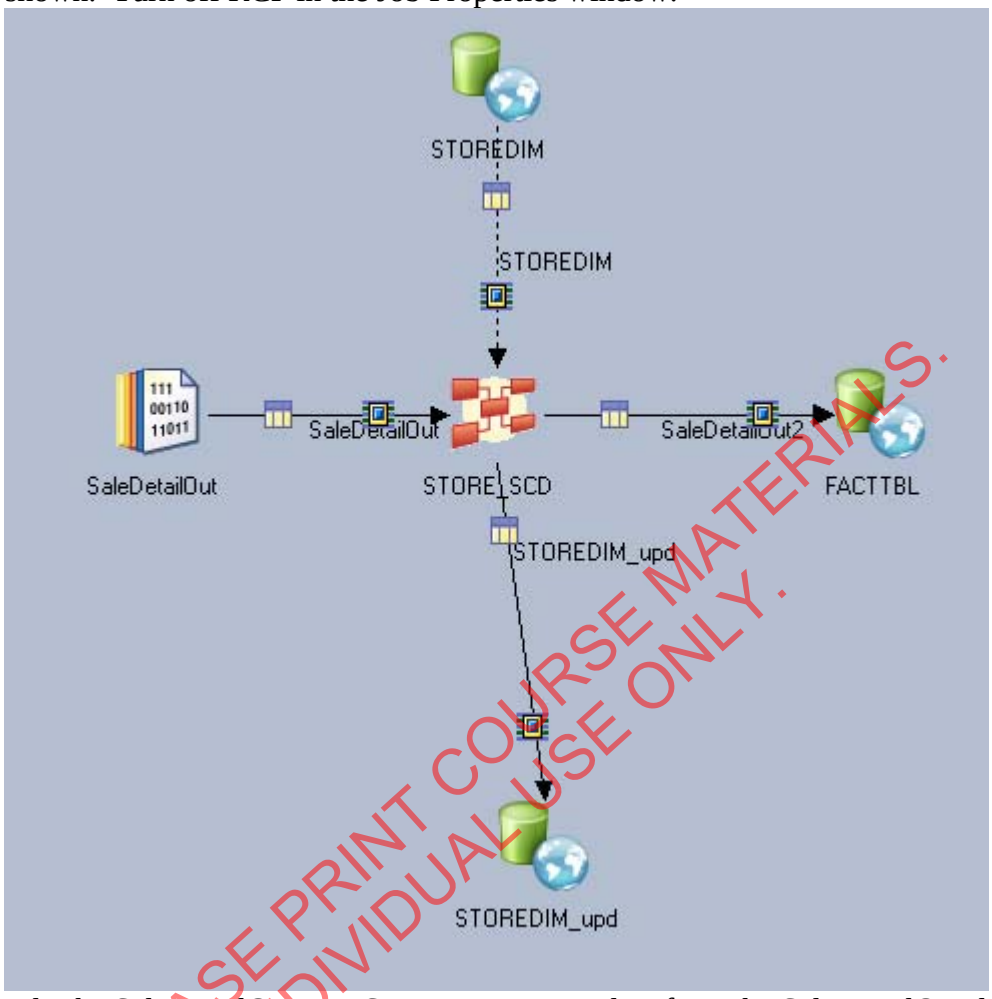
scdLoadFactTable_1..SaleDetailOut.SaleDetailOut - Data Browser

StoreID	StoreName	StoreMgr	PRODSK	SaleAmt	SaleUnits
A1111	Stuff	Washington	3	00436.1	13
A1112	MoreStuff	Adams	4	00456.5	14
A1113	Stuffy's	Jefferson	1	00203.3	7
A1114	McStuff	Madison	5	00308.8	2
A1115	Stuff Jr.	Monroe	6	00024.4	11

18. If you want to rerun your job. Drop the three star schema tables and then re-run the sql file that creates the tables. Delete the surrogate key source files and then re-run the job that creates and updates them.

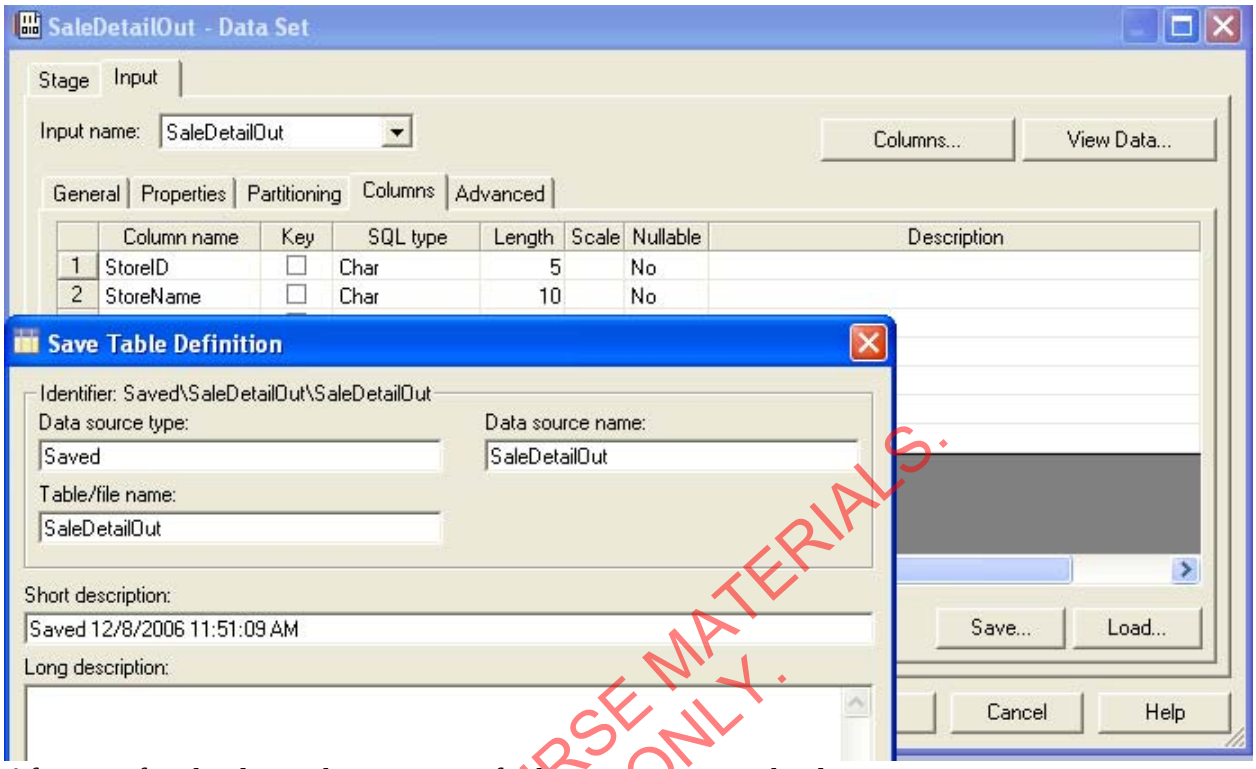
Task: Build an SCD job to process the second dimension

1. Create a new parallel job named scdLoadFactTable_2. Add the stages and links as shown. Turn off RCP in the Job Properties window.

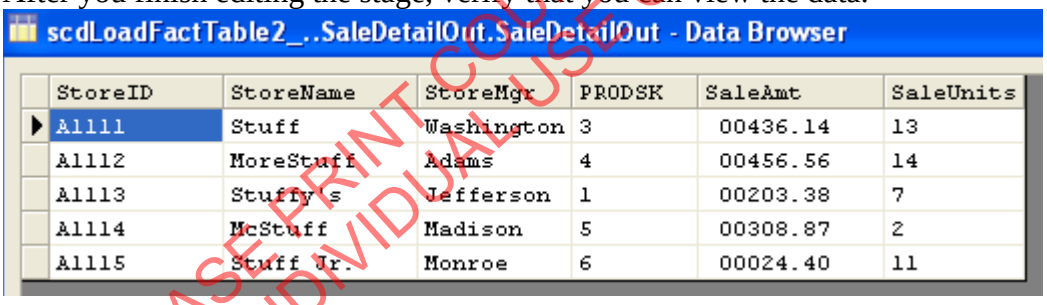


2. Edit the SaleDetailOut DataSet stage. Extract data from the SaleDetailOut.ds file that you created in the previous job. To get the Table Definition go to the Columns tab of the target DataSet stage in your previous job. Click the Save button to save the columns as a

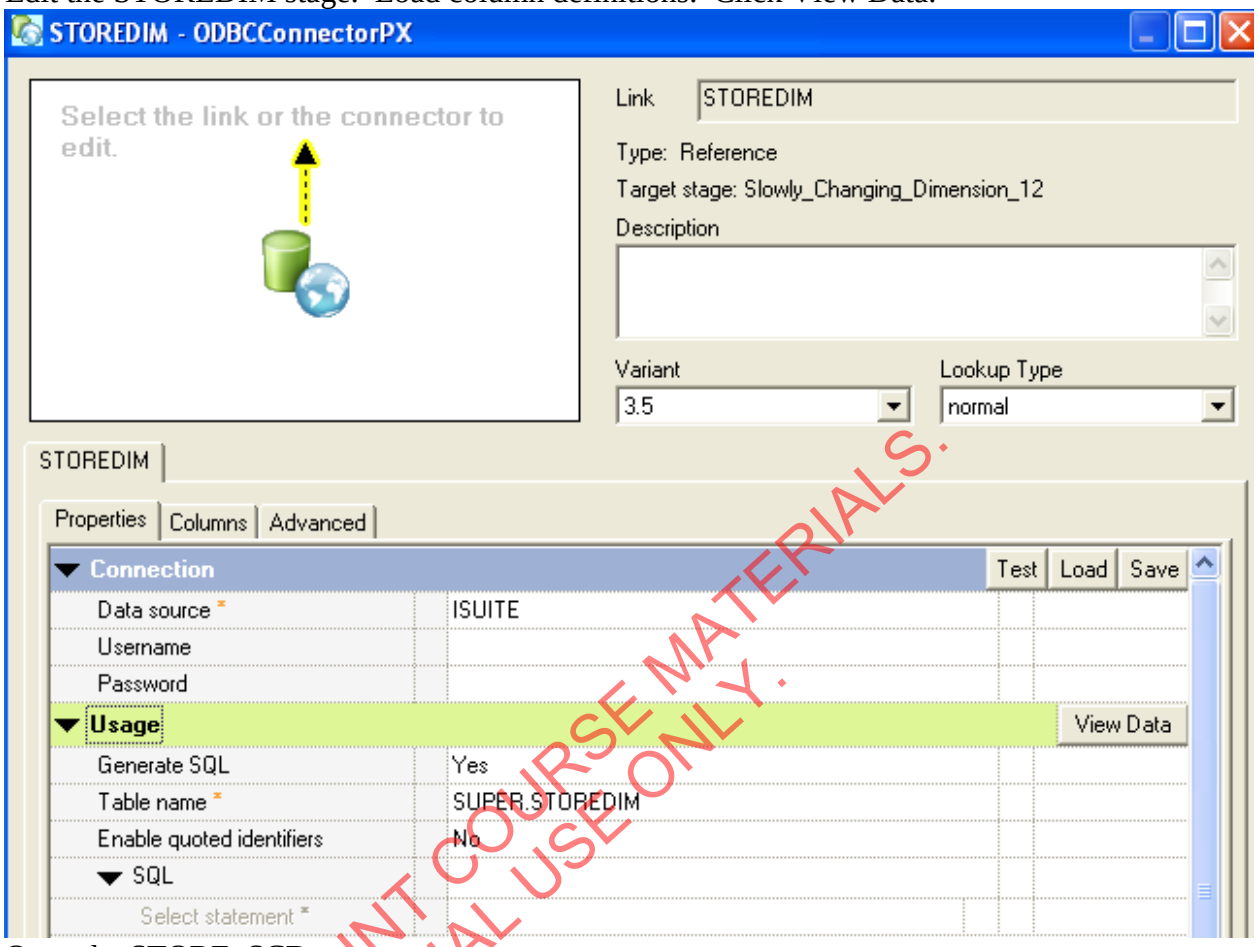
new Table Definition.



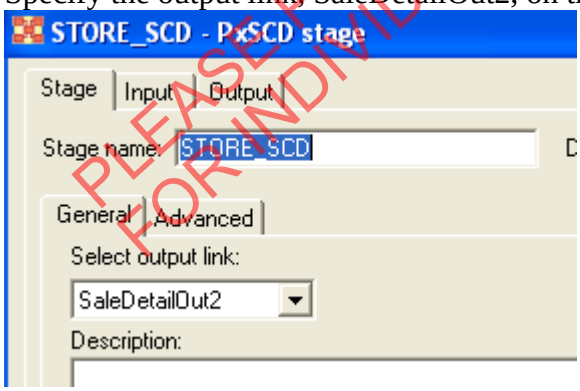
3. After you finish editing the stage, verify that you can view the data.



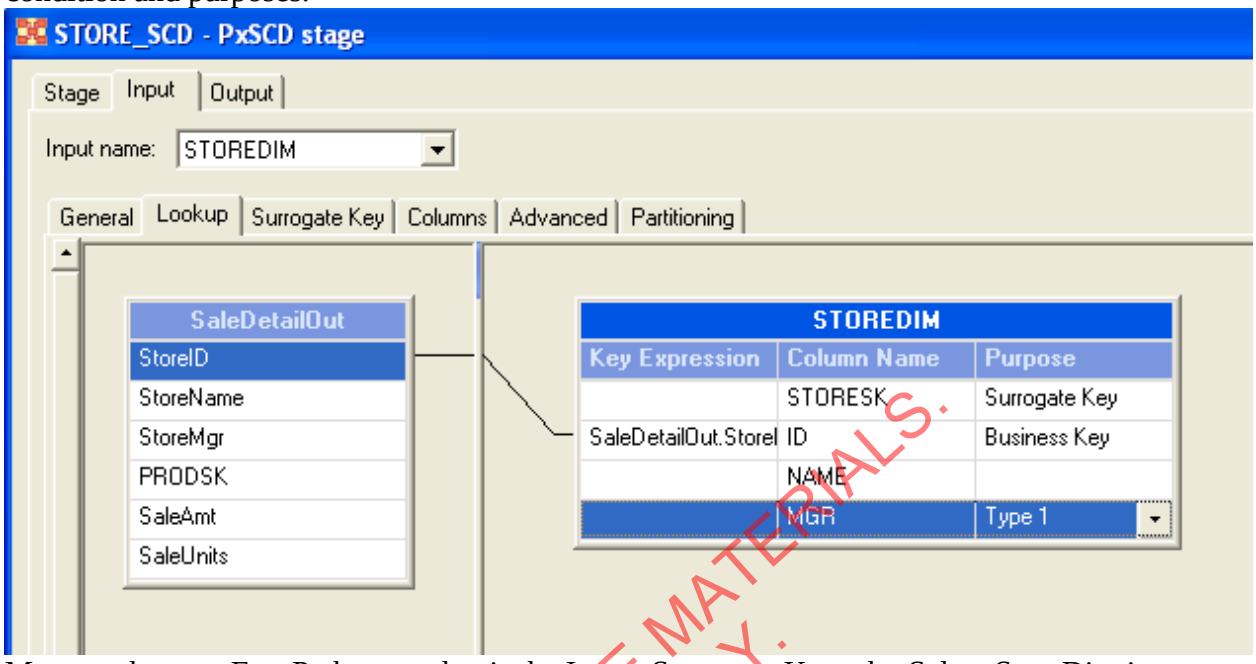
4. Edit the STOREDIM stage. Load column definitions. Click View Data.



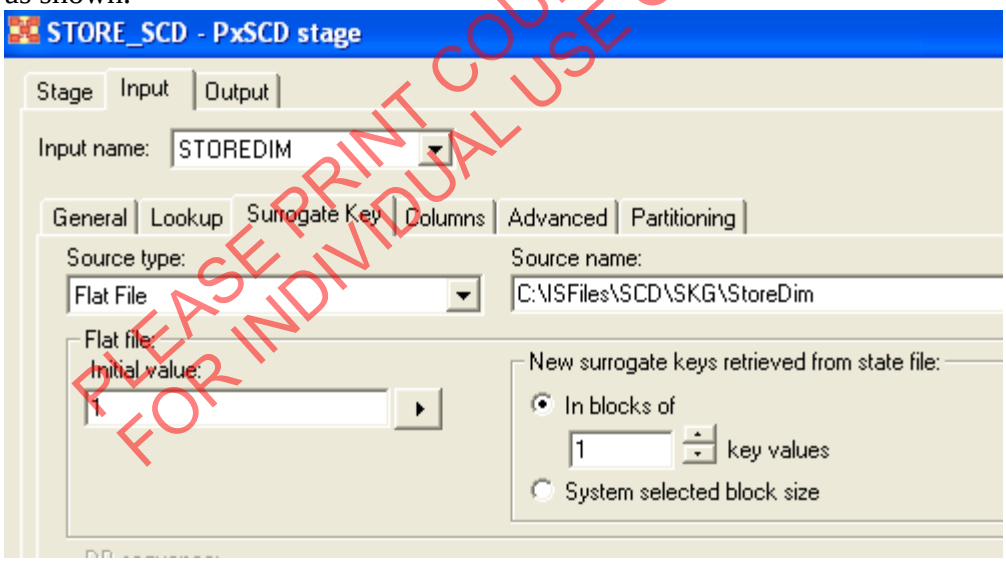
5. Open the STORE_SCD stage.
6. Specify the output link, SaleDetailOut2, on the first Fast Path page.



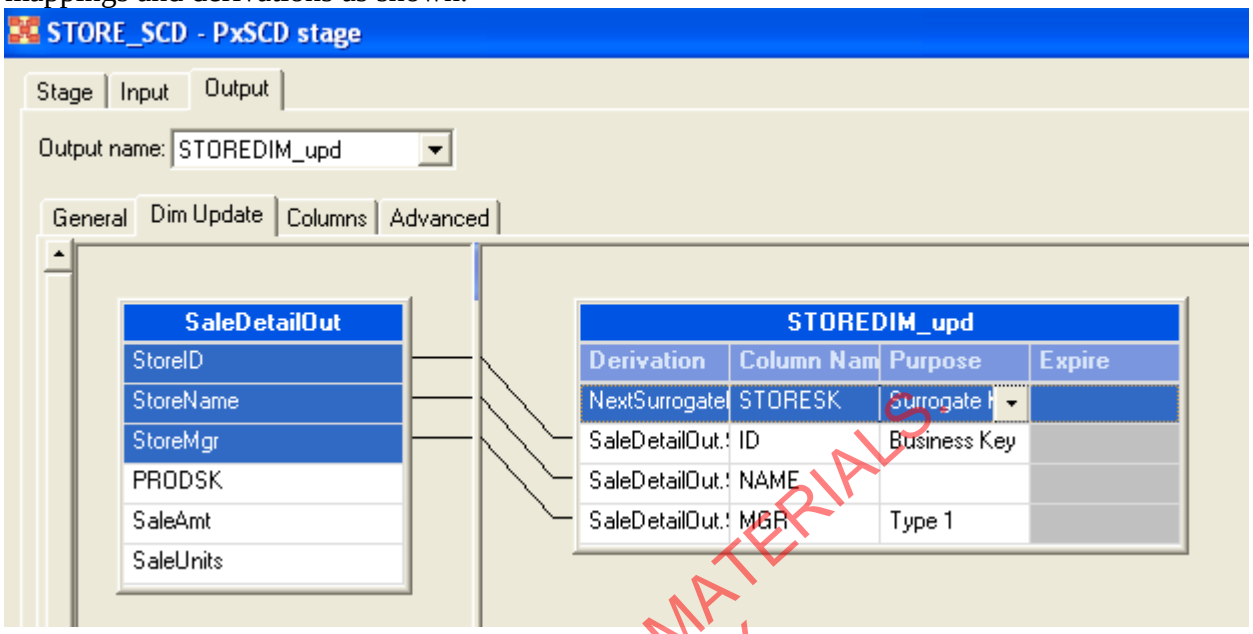
7. Move to the next Fast path page, that is, the Input>Lookup tab. Specify the lookup condition and purposes.



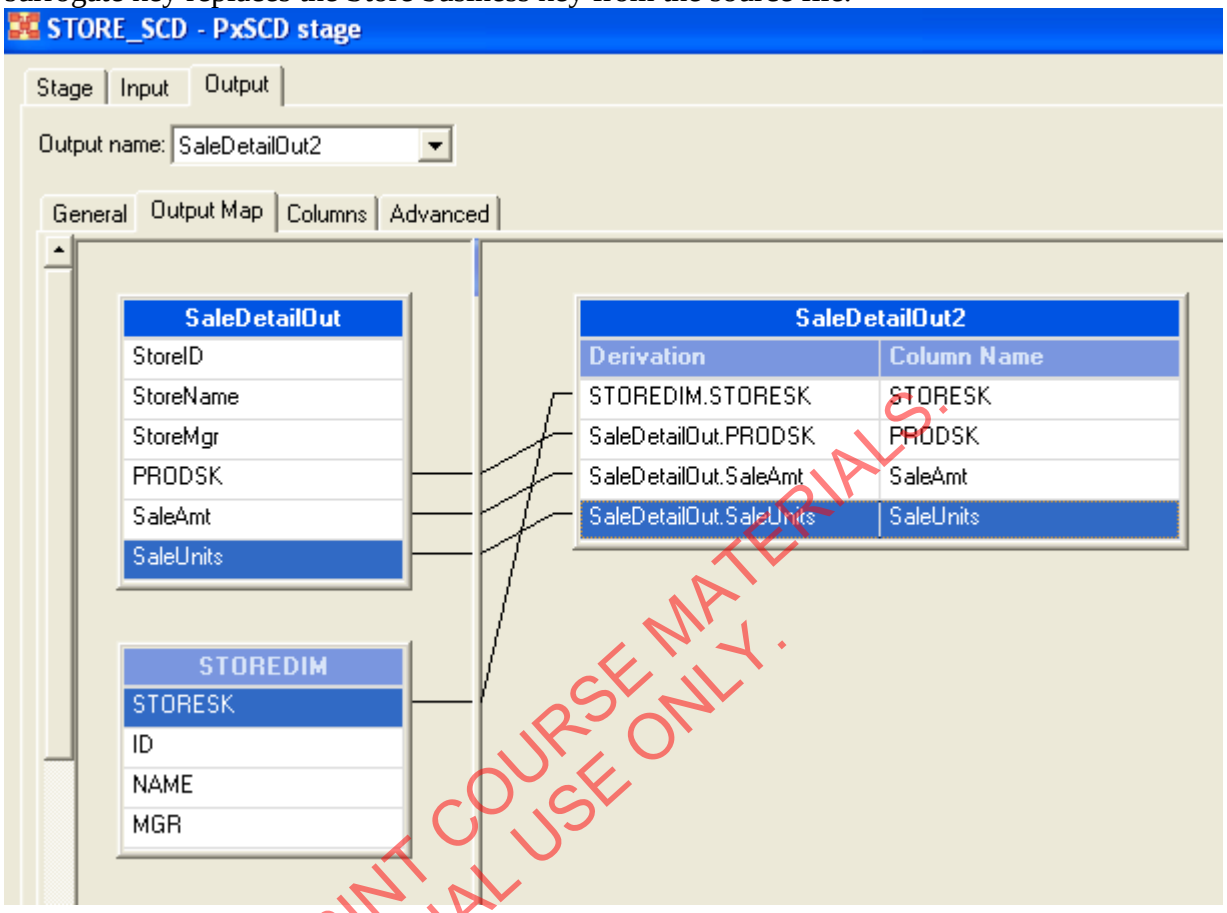
8. Move to the next Fast Path page, that is the Input>Surrogate Key tab. Select StoreDim in the SCD>SKG directory as the source key file to be used. Specify the other information as shown.



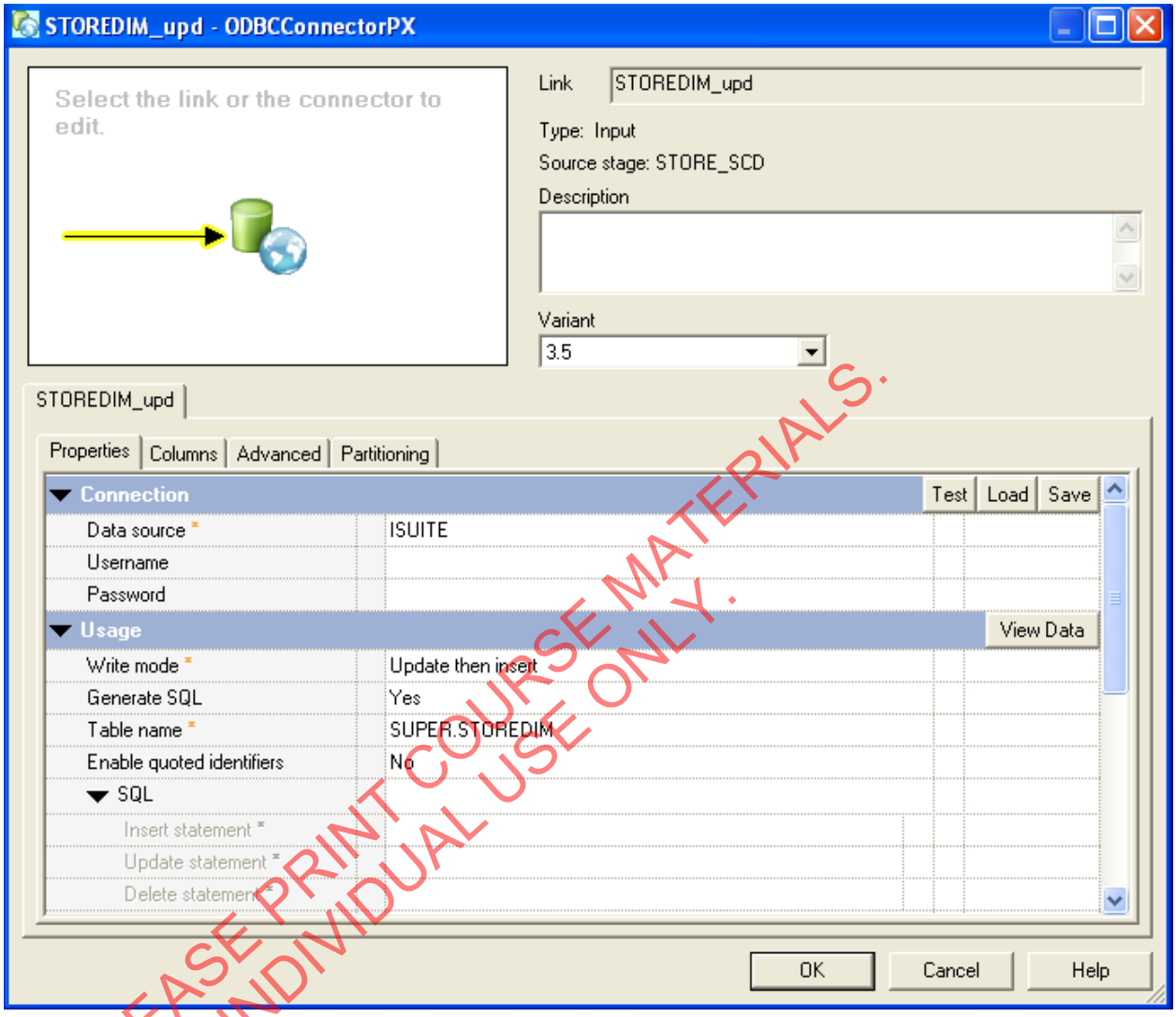
9. Move to the next Fast Path page, that is the Output>Dim Update tab. Specify the mappings and derivations as shown.



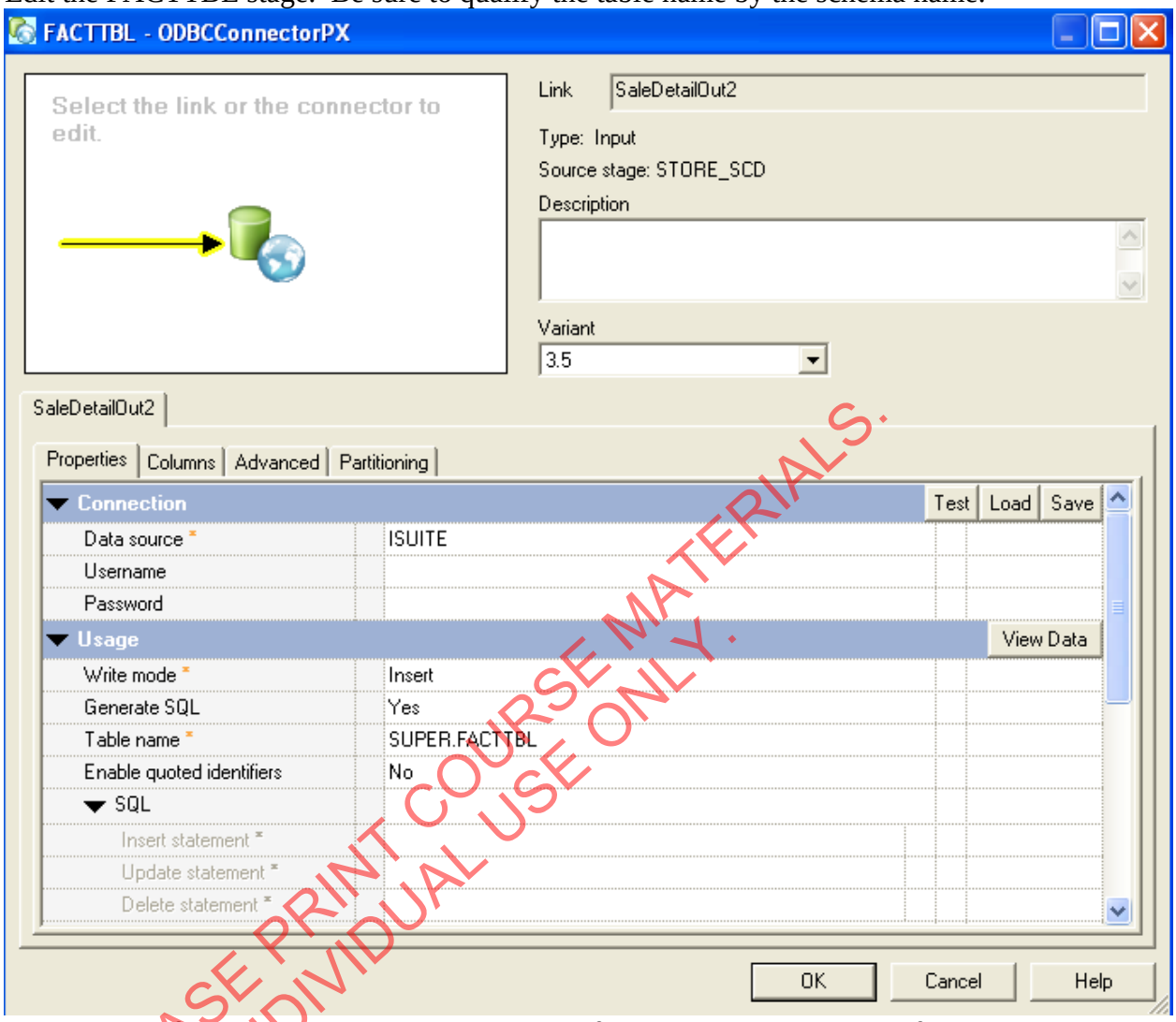
10. Move to the next Fast Path page, that is the Output>Output Map tab. Here the STORE surrogate key replaces the Store business key from the source file.



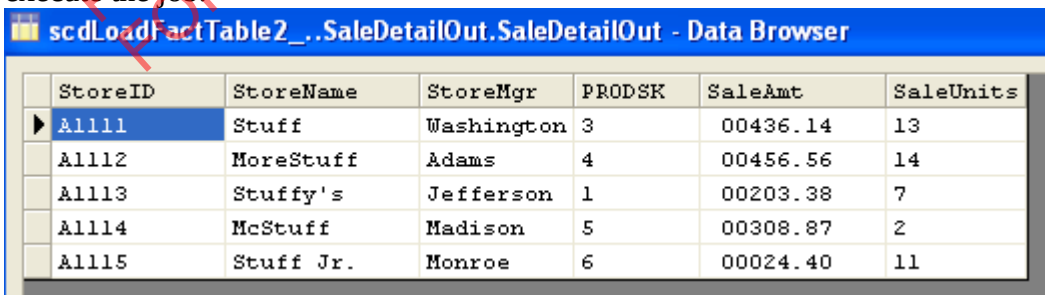
11. Edit the STOREDIM_upd stage. Be sure to qualify the table name by the schema name, as shown.



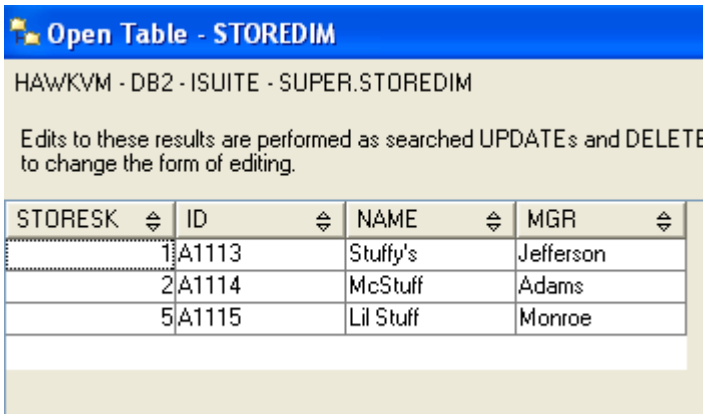
12. Edit the FACTTBL stage. Be sure to qualify the table name by the schema name.



13. Compile. Before you run the job, view the data from the SaleDetailOut.ds file and the STOREDIM dimension table. This way you can see clearly what happens when you execute the job.



StoreID	StoreName	StoreMgr	PRODSK	SaleAmt	SaleUnits
Allll	Stuff	Washington	3	00436.14	13
Alll2	MoreStuff	Adams	4	00456.56	14
Alll3	Stuffy's	Jefferson	1	00203.38	7
Alll4	McStuff	Madison	5	00308.87	2
Alll5	Stuff Jr.	Monroe	6	00024.40	11



14. Run the job. Check the job log for errors. View the data in the updated STOREDIM table and in the FACTTBL.

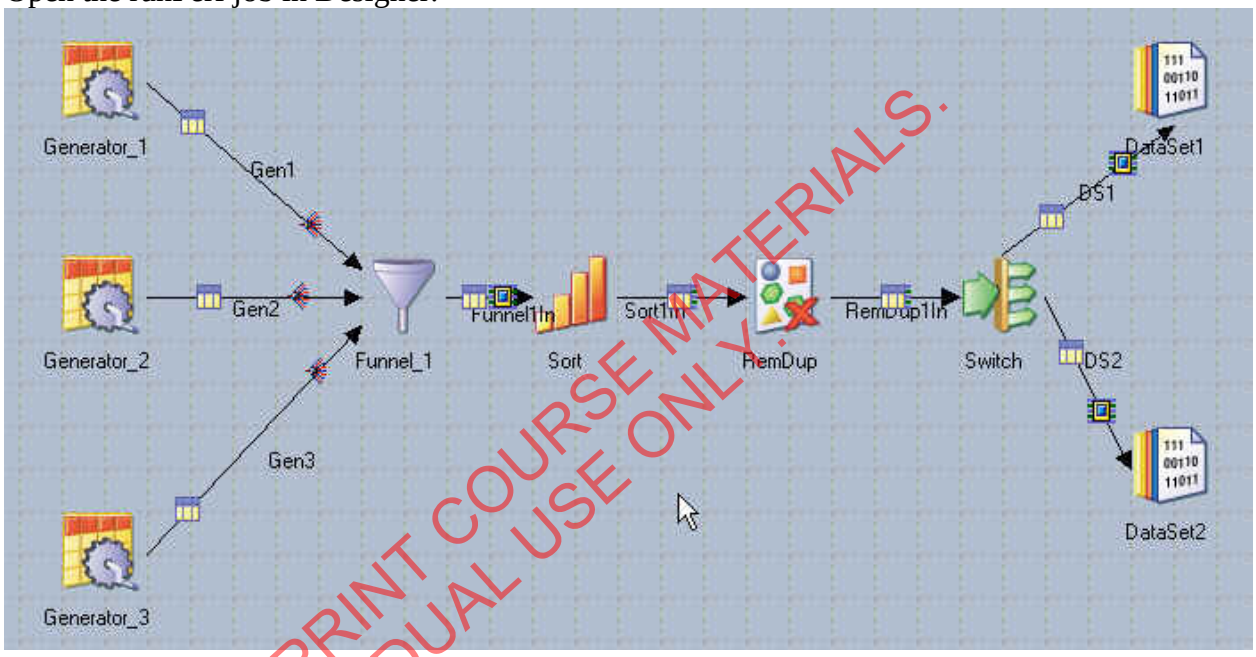
Special Topic 4: DataStage Utilities

Assumptions

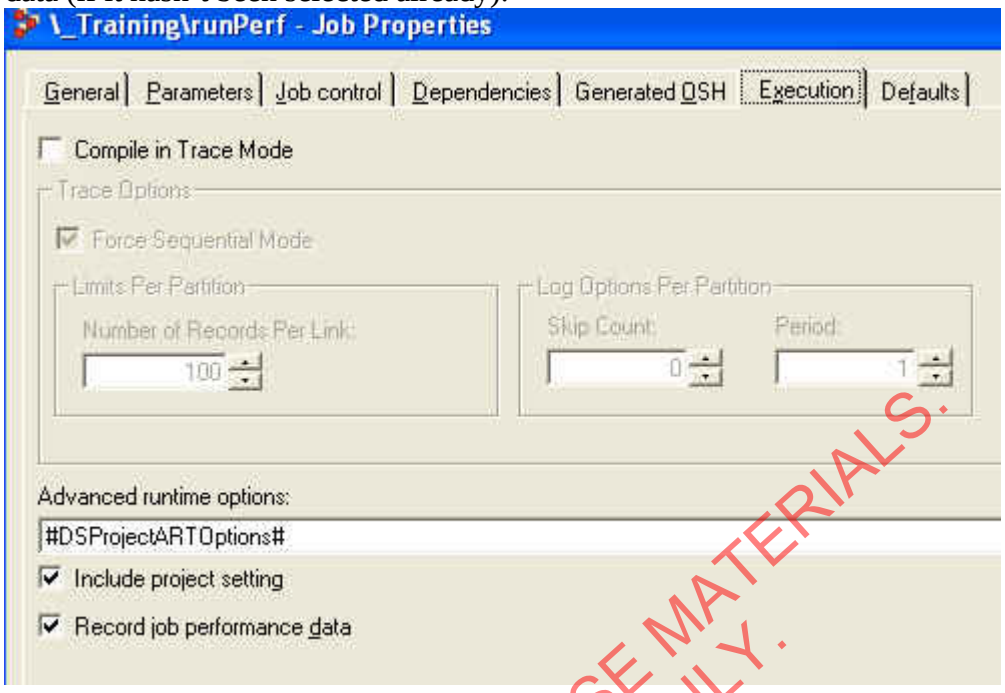
- The runPerf_Job.dsx file exists in your ISFiles directory

Task: Analyze the performance of a job

1. Import the runPerf_Job.dsx file from your ISFiles>Dsx directory.
2. Open the runPerf job in Designer.

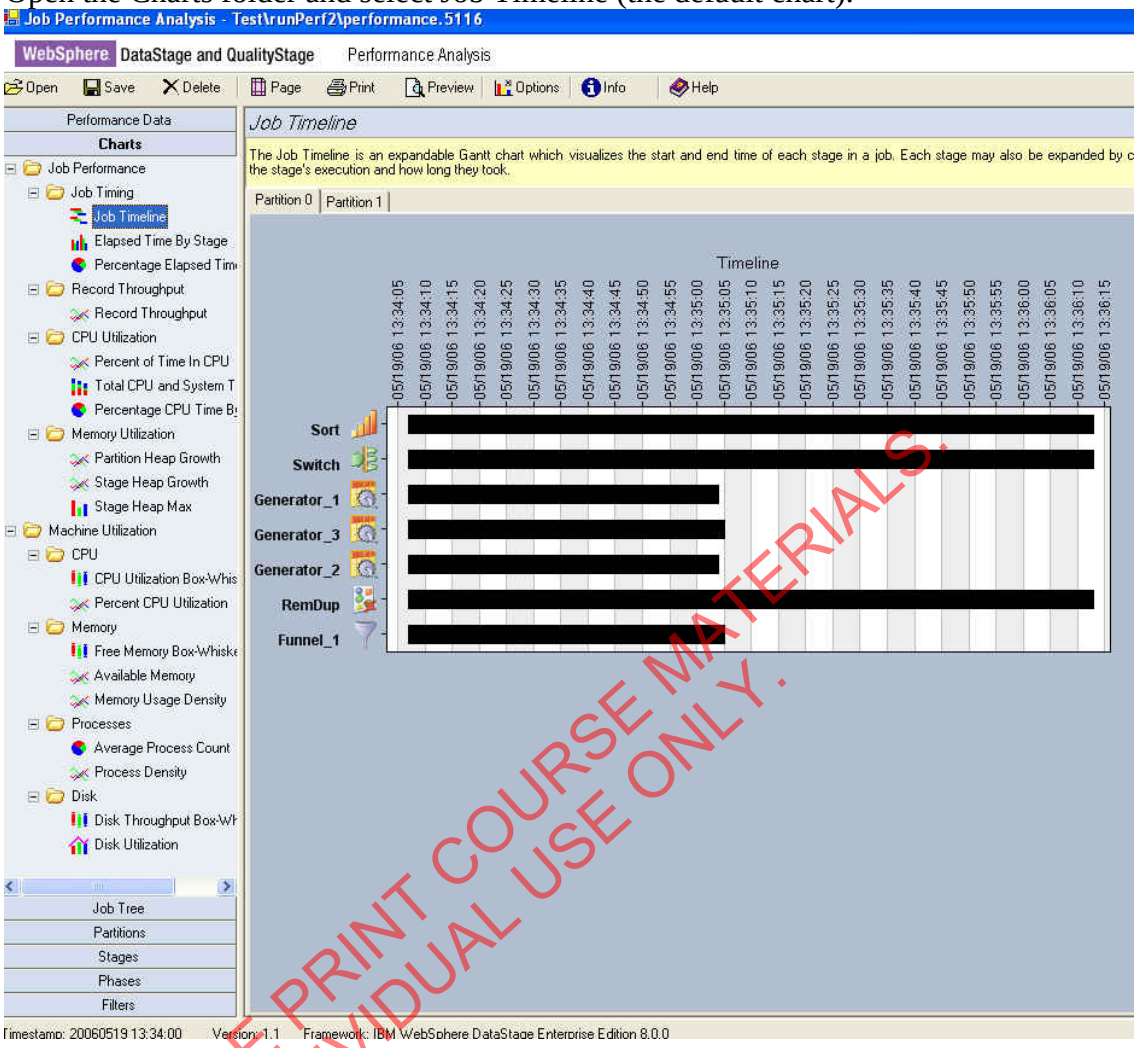


3. Open up Job Properties and click on the Execution tab. Select Record job performance data (if it hasn't been selected already).



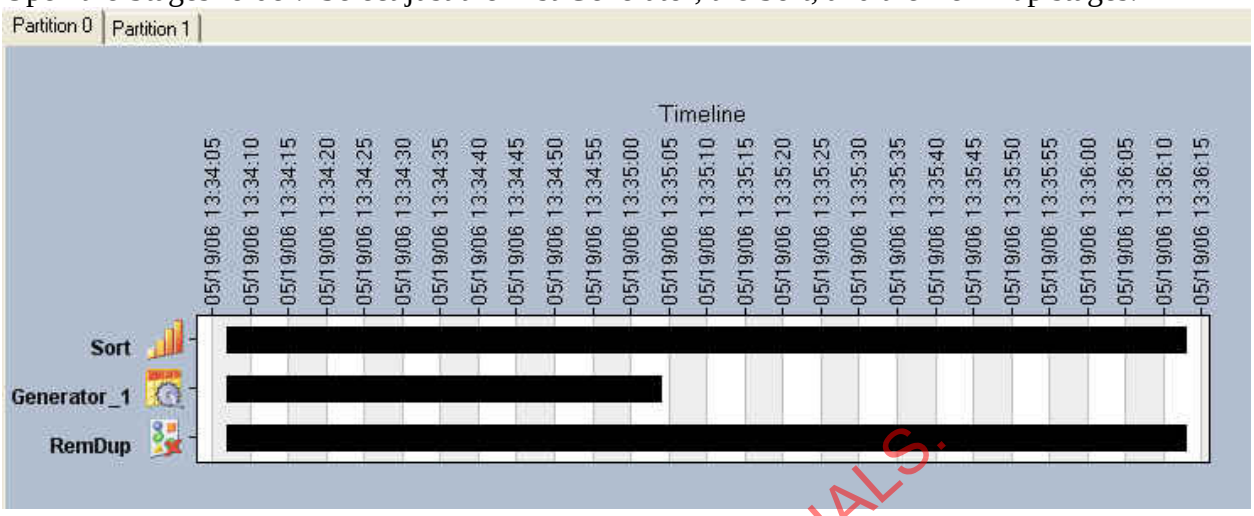
4. Compile and run your job. Verify in Director that it runs to successful completion.
5. Click on the Performance Analyzer icon in the toolbar.

6. Open the Charts folder and select Job Timeline (the default chart).

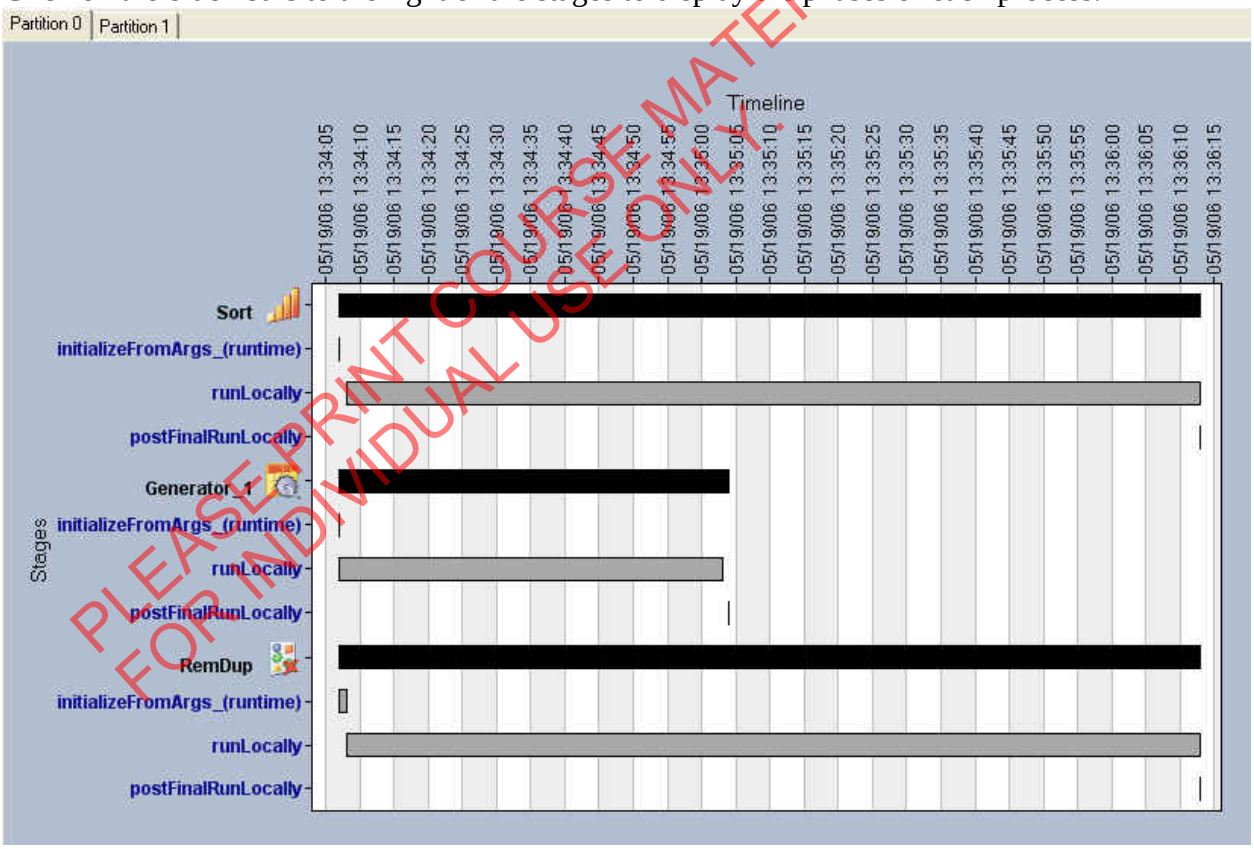


7. Open the Partitions folder. Deselect one of the Partitions. Notice that the corresponding tab disappears on the chart. Reselect the partition.

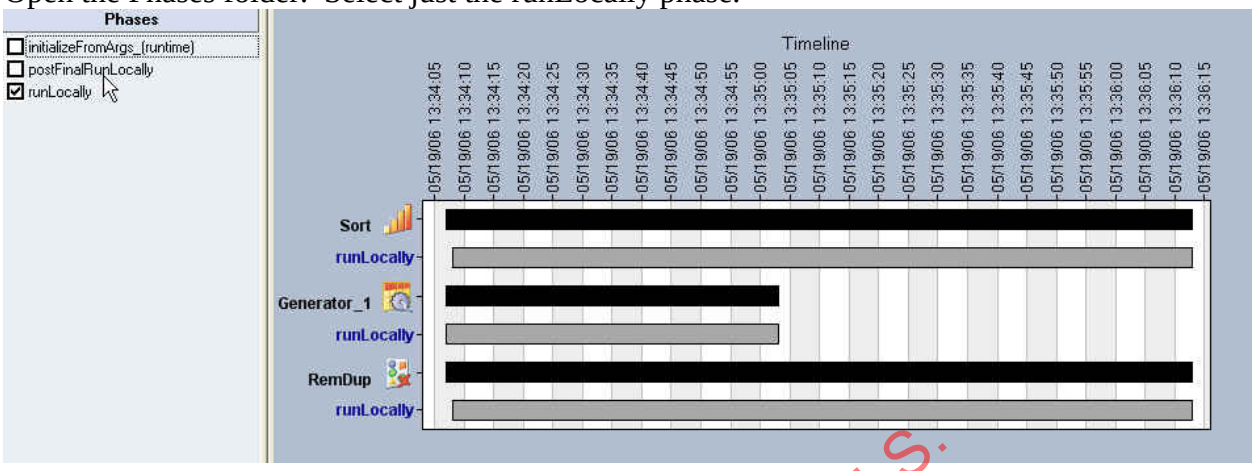
8. Open the Stages folder. Select just the first Generator, the Sort, and the RemDup stages.



9. Click on the black bars to the right of the stages to display the phases of each process.



10. Open the Phases folder. Select just the runLocally phase.



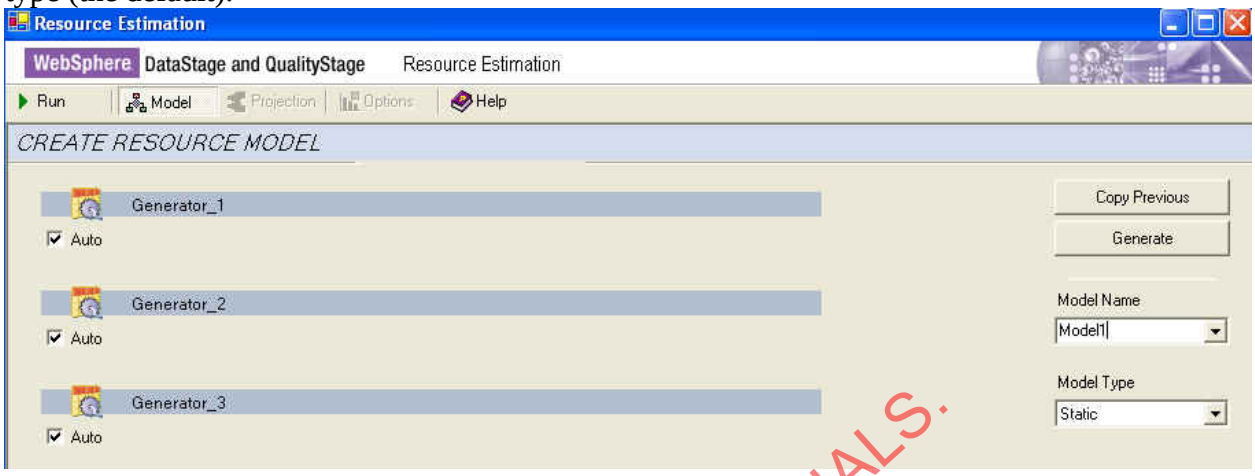
11. Open the Filters tab. Deselect each box one at a time and examine the effect on the chart. Shown below is the effect of deselecting the Hide Startup Phases box.

12. Open up the Charts folder. Examine each chart in the Job Timing, Record Throughput, CPU Utilization, Memory Utilization, and Machine Utilization folders.

Task: Estimate the resources of a job

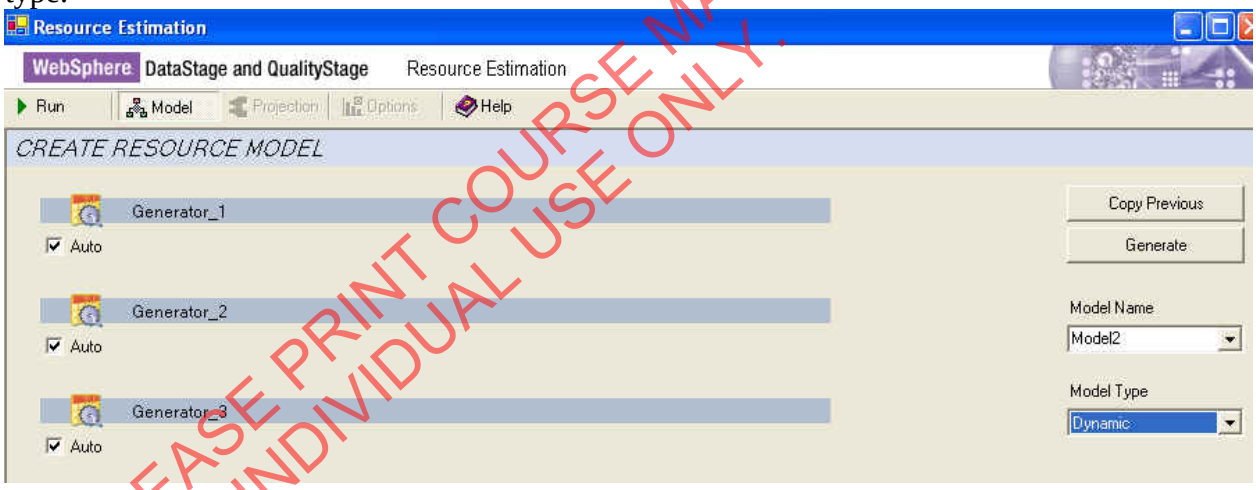
1. Click on the Resource Estimator icon. Run your job when prompted.

2. On the Model tab, enter Model1 in the Model Name box. Specify Static as the model type (the default).



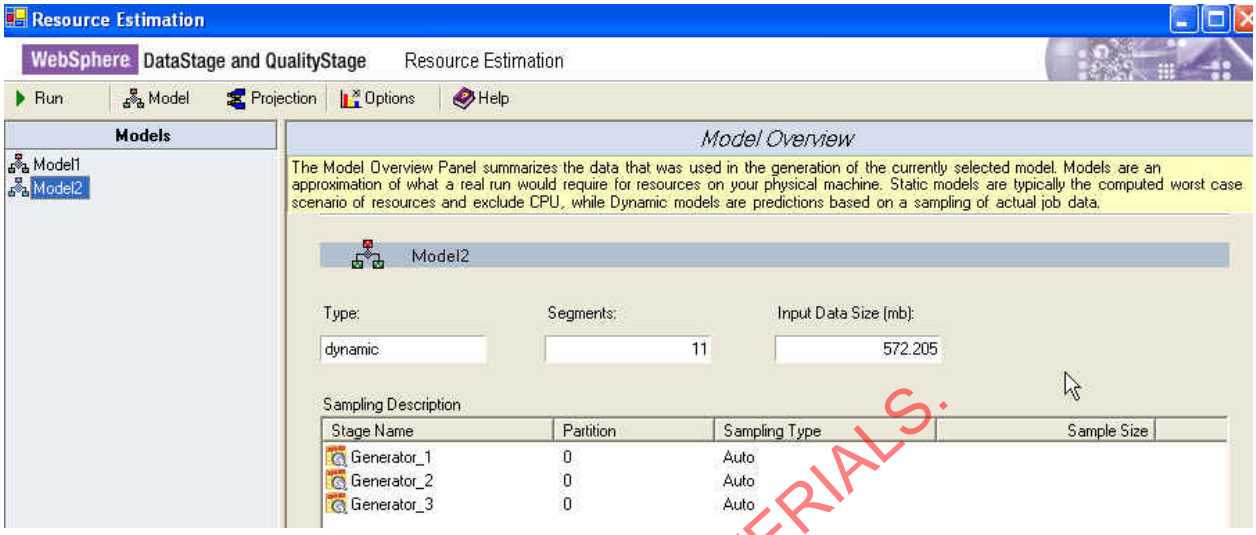
3. Click Generate.

4. On the Model tab, enter Model2 in the Model Name box. Specify Dynamic as the model type.

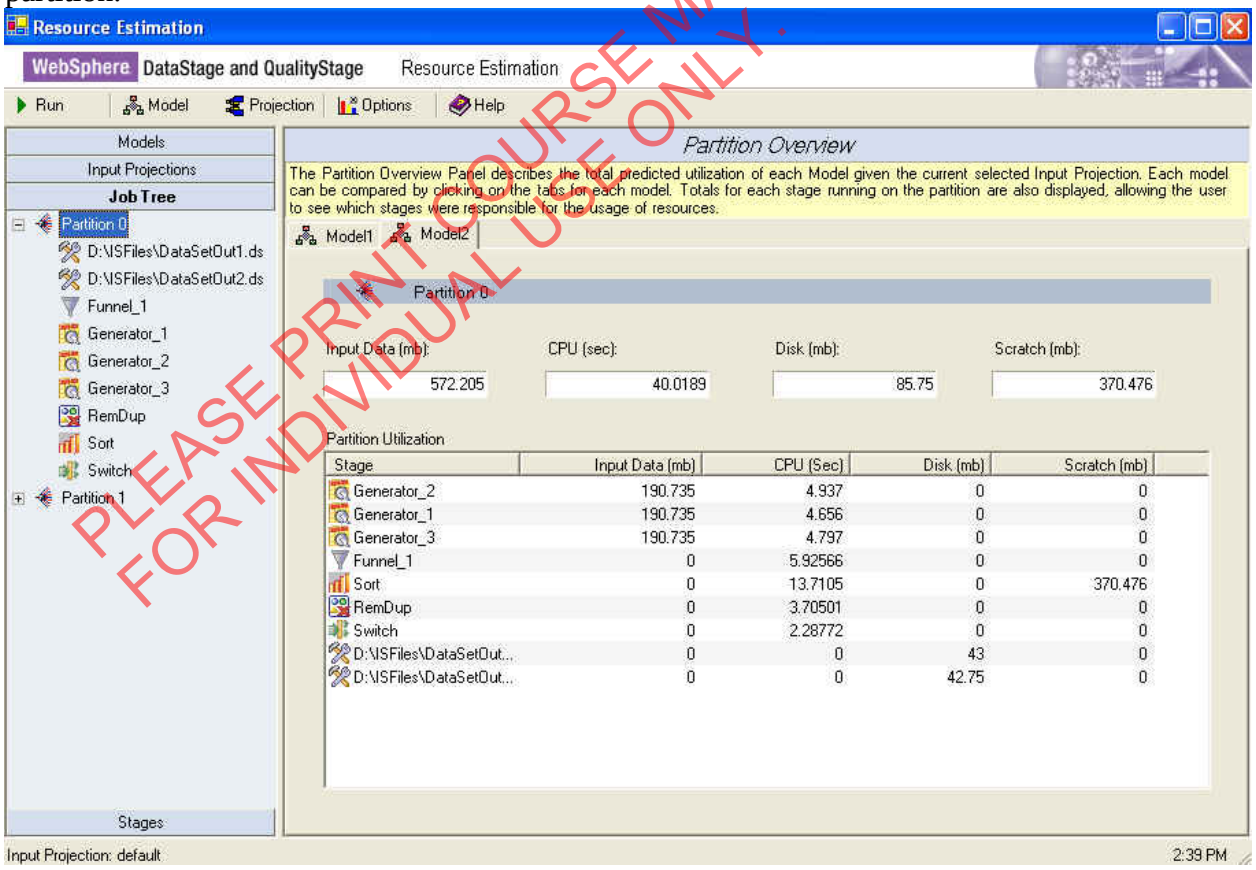


5. Click Generate.

6. On the Model Overview window, select each of your models in the Models folder and view the results.

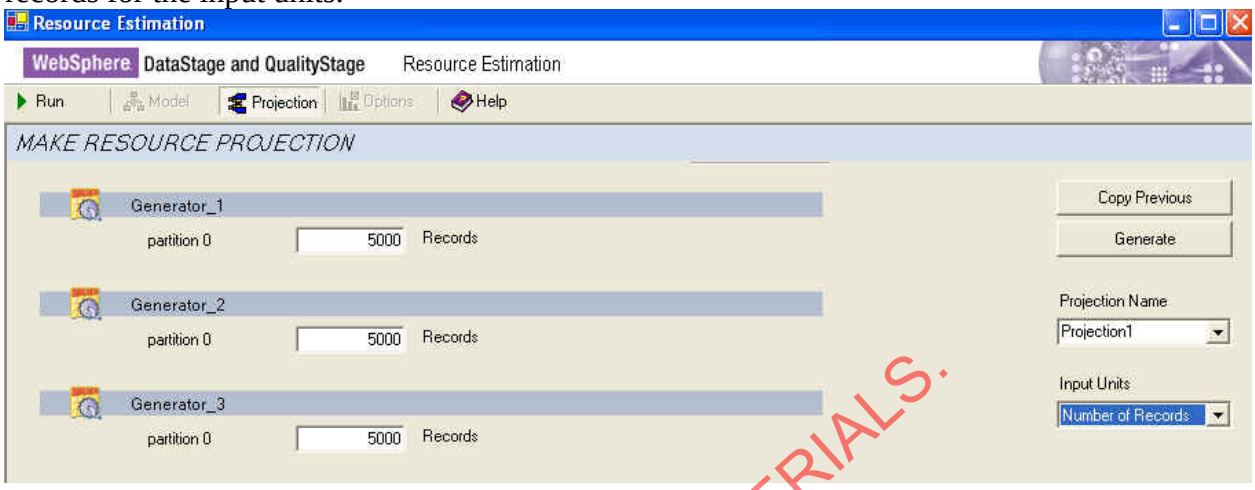


7. Open the Job Tree folder. View and compare the information for each model on each partition.



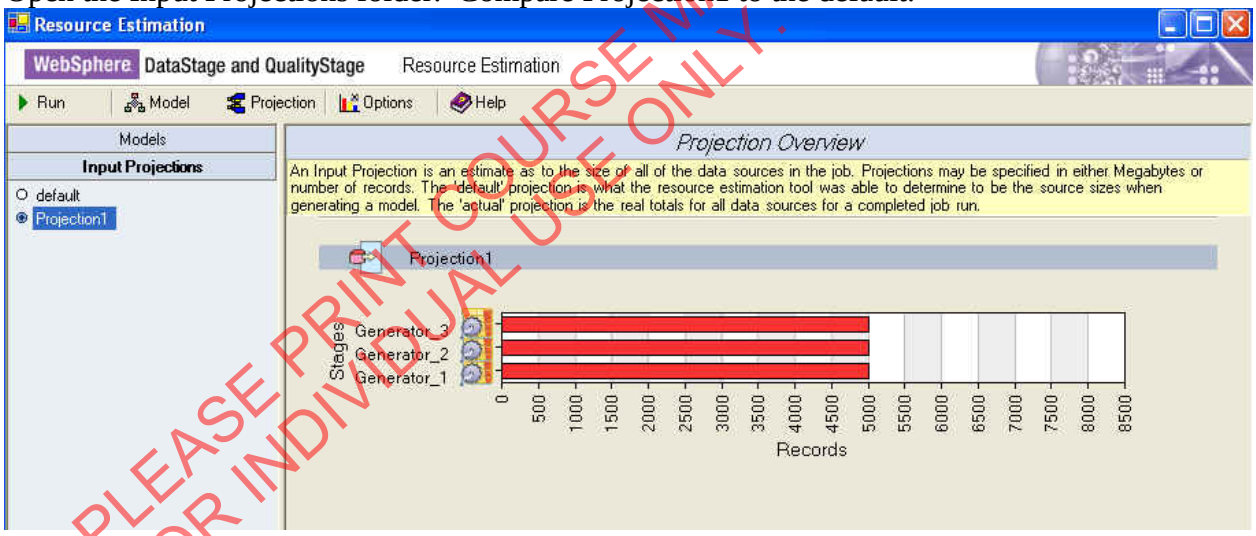
8. Select individual stages on the tree and view the results. Explore!

9. Click the Projection button at the top of the window. Enter 5000 MB for each input Generator stage. Enter Projection1 in the Projection Name box. Specify number of records for the input units.



10. Click Generate.

11. Open the Input Projections folder. Compare Projection1 to the default.



Special Topic 5: Teradata

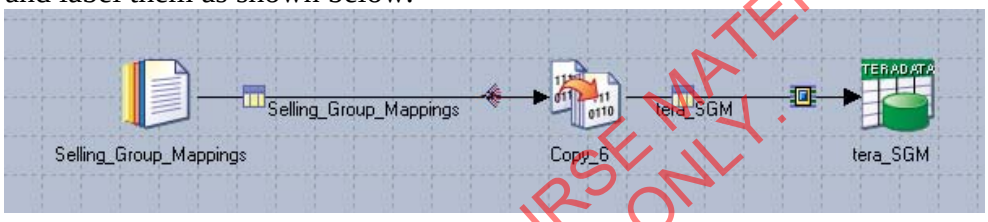
Assumptions

- The directory on the server contains a file named Selling_Group_Mapping.txt.
- Teradata is set up and configured.

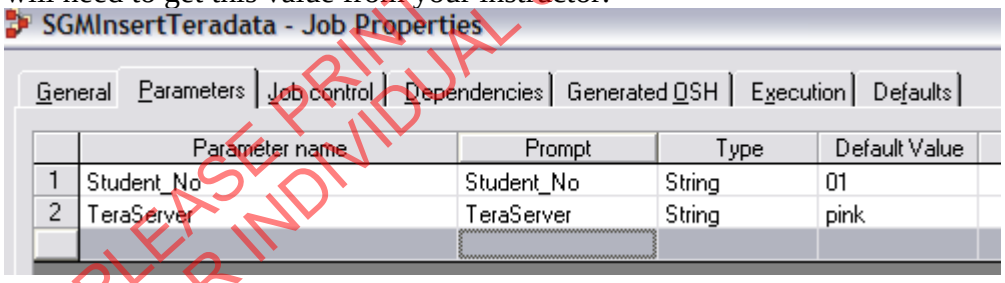
Task: Teradata EE Stage as a Target

In this task you create a job that loads records from Selling_Group_Mapping.txt into a Teradata table. You will use the Teradata Enterprise stage to do the inserts. The Teradata Enterprise stage also allows you to create a table in the database.

1. Open a new Parallel job and save it under the name SGMInsertTeradata.
2. Add a sequential stage for source, a Copy stage, and a target Teradata Enterprise stage and label them as shown below.

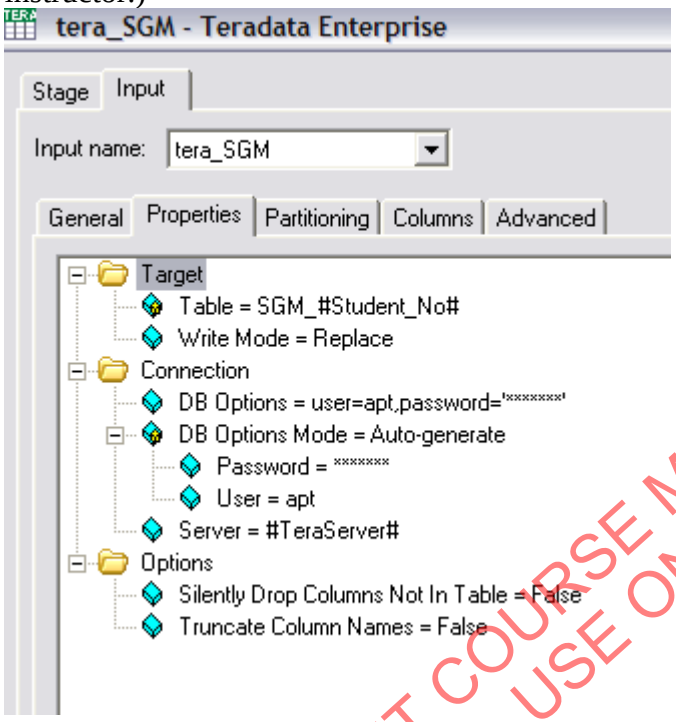


3. Create two job parameters **Student_No** and **Tera_Server**. Set the Student_No to your student number. Set the Tera_Server default to the name of the Teradata system. You will need to get this value from your instructor.



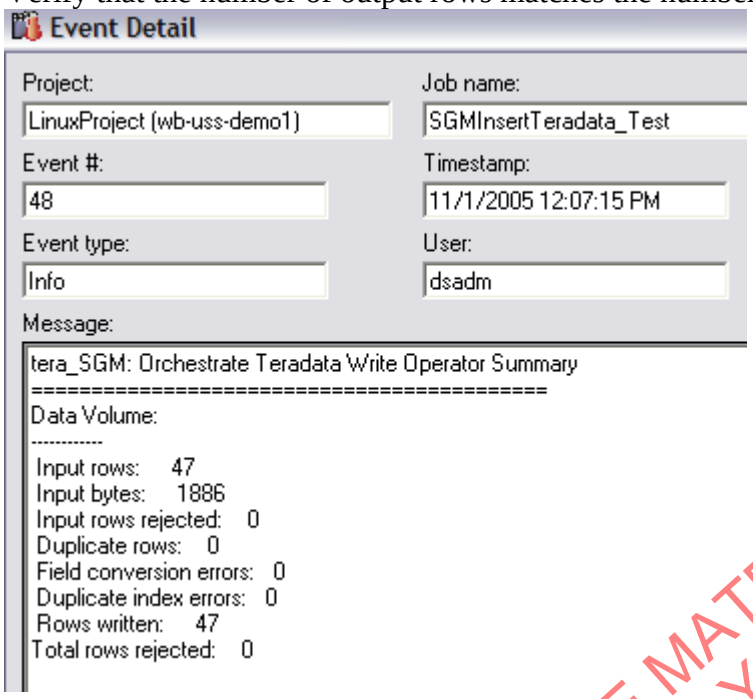
4. Edit the sequential stage labeled **Selling_Group_Mappings** to read from Selling_Group_Mapping.txt file. If you don't have an existing Table Definition for it, import the metadata from the Selling_Group_Mapping.schema file.
5. View the data to verify that the stage can read the file properly.
6. In the Copy stage, map all the columns from source to target. Open up the Teradata Enterprise stage.
7. Write to the SGM_#Student_No# table, where "#Student_No#" is the job parameter you defined in earlier.

8. Specify the Write Mode as “Replace”. This will create a new table if none exists or replace an existing table.
9. Specify the Server using your job parameter.
10. Specify the User and Password to connect to Teradata. (Get this information from your instructor.)



11. Compile and run the job. Check the job log. There shouldn’t be any errors or warnings.
12. In the Director log, open up the message labeled. “Orchestrate Teradata Write Operator Summary.” (Note: There are two. Open up the one sub-labeled “Data Volume”.)

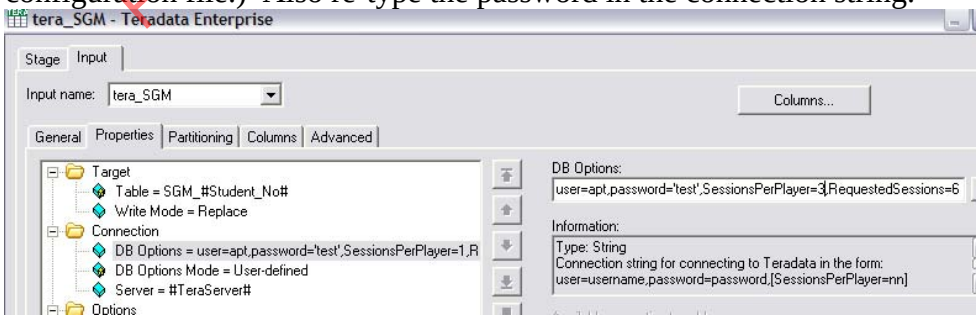
Verify that the number of output rows matches the number of input rows.



Optional Task: Specify the SessionsPerPlayer and RequestedSessions options.

Perform this task only if directed to by your instructor.

1. Open up your target Teradata Enterprise stage.
2. Change the DB Options Mode setting to “User-defined”.
3. Add the SessionsPerPlay and RequestedSessions options to the DB Options string. Your instructor will provide the setting for RequestedSessions (in the example 6). Calculate the value for SessionsPerPlayer by dividing the RequestedSessions value by the number on nodes defined in your configuration file. (The example assumes an 2-node configuration file.) Also re-type the password in the connection string.



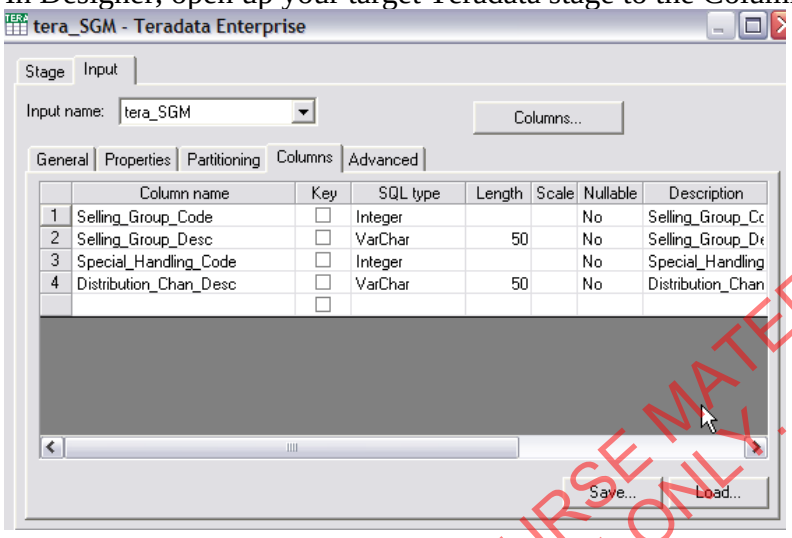
4. Recompile and run your job.

5. Examine the log.

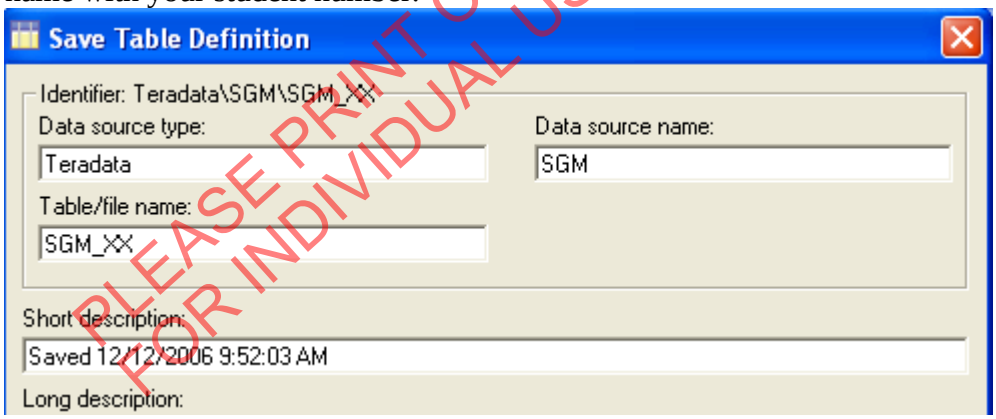
Task: Save a Table Definition for a Teradata table

In this task you will save a Table Definition for the SGM_XX table.

1. In Designer, open up your target Teradata stage to the Columns tab.



2. Click the Save button and specify the information as shown. Replace "XX" in the Table name with your student number.



3. Verify in the Table Definitions folder in the Repository that your table definition has been saved.

Teradata EE Stage as a Source

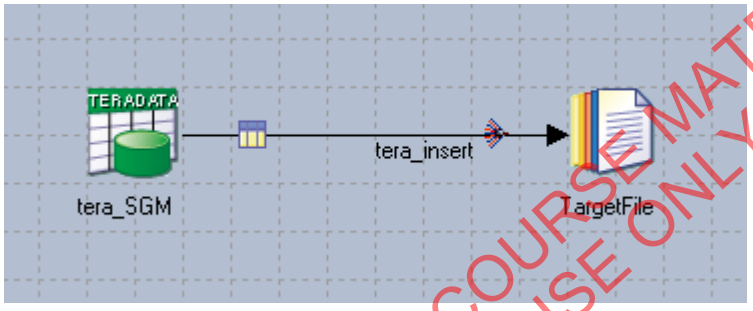
Assumptions

- The job **SGMInsertTeradata** has been successfully run and the table SGM_XX has been created.

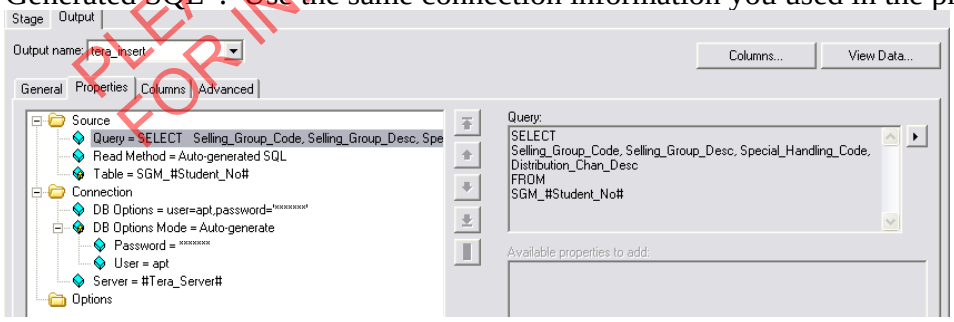
Task: Teradata EE Stage as a Source

In this task you design a job that reads records from SGM_XX DB2 table that you created in the previous lab.

1. Save your SGMInsertTeradata job as SGMSelectTeradata.
2. Move the stages around as shown. Here, you will be using the Teradata stage to read rather than write.

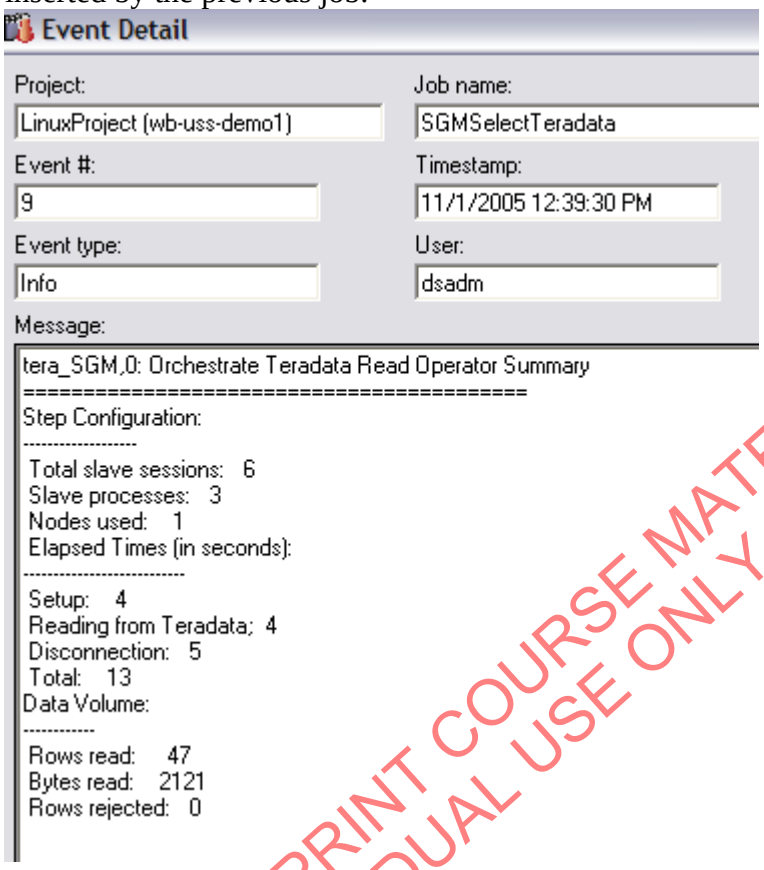


3. Verify that you have the two job parameters Student_No and TeraServer as in the previous lab.
4. Open the source Teradata stage. On the Columns tab, load the Table Definition for the SGM_XX.
5. Select data from SGM_XX table that was created in the previous exercise using "Auto Generated SQL". Use the same connection information you used in the previous lab.



6. Click the View Data button to verify that you can read the table using the specified connection information.
7. In the Sequential stage, write to a file named TargetFile.txt.

8. Compile and run the job. Verify the results by examining the job log and the target data.
9. In the job log, open the message labeled “Orchestrate Teradata Read Operator Summary” and examine its contents. Verify that the number of rows read (47) matches the number inserted by the previous job.



Teradata Multiload Stage

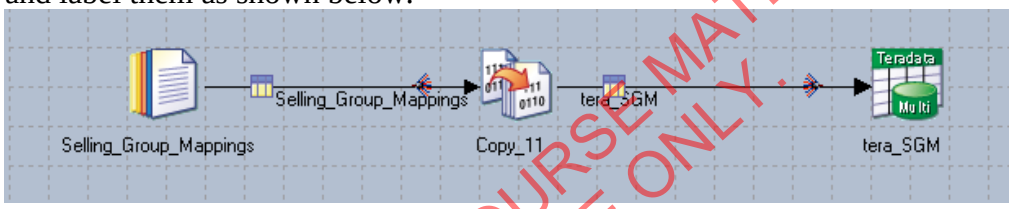
Assumptions

- The directory on the server contains file named Selling_Group_Mapping.txt.
- You successfully ran the SGMInsertTeradata job that creates a table named SGM_#Student_No#. For this exercise to work, the target table must already exist.

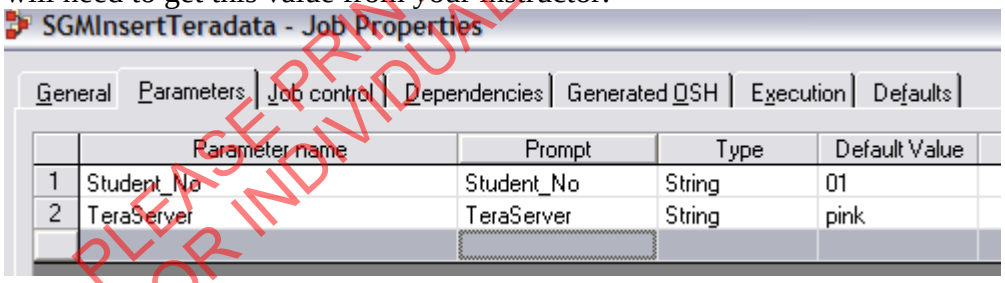
Task: Teradata MultiLoad Stage as a Target

In this task you create a job that loads records from Selling_Group_Mapping.txt into a Teradata table. You will use the Teradata Multiload stage to do the inserts.

1. Open a new Parallel job and save it under the name SGMInsertTeradata_MultiLoad.
2. Add a sequential stage for source, a Copy stage, and a target Teradata Multiload stage and label them as shown below.

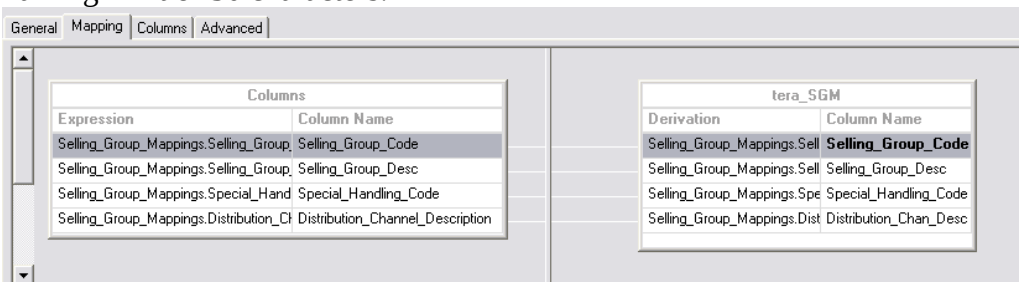


3. Create two job parameters **Student_No** and **Tera_Server**. Set the Student_No to your student number. Set the Tera_Server default to the name of the Teradata system. You will need to get this value from your instructor.

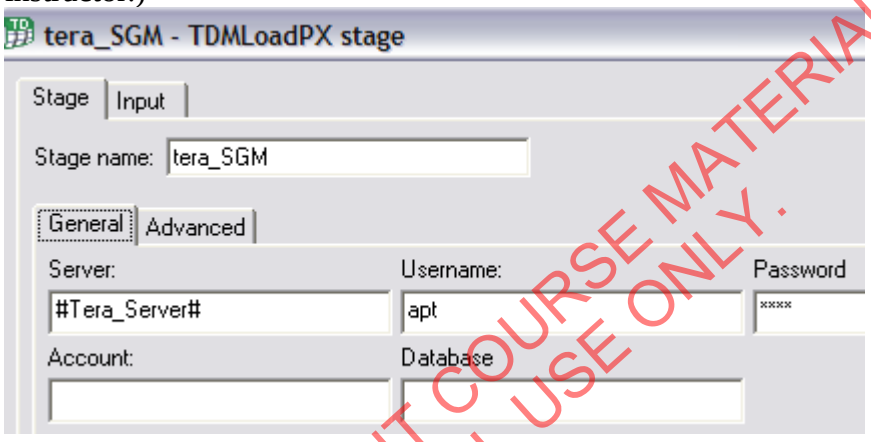


4. Edit the sequential stage labeled **Selling_Group_Mappings** to read from Selling_Group_Mappings.txt file. If you don't have an existing Table Definition for it, import the metadata from the Selling_Group_Mappings.schema file.
5. View the data to verify that the stage can read the file properly.
6. In the Copy stage, map all the columns from source to target. Important: Change the name of the last target column as shown. This is necessary to satisfy Teradata's column

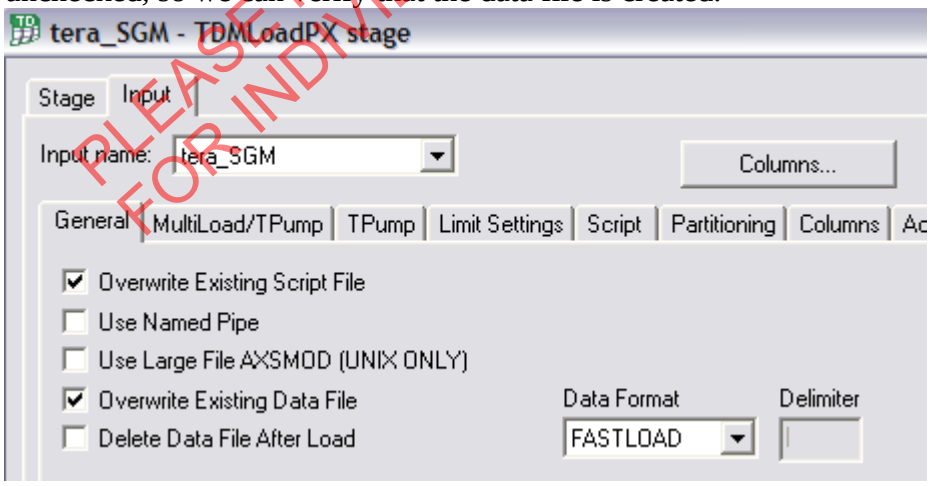
naming limit of 30 characters.



7. Open up the Teradata Multiload stage to the General tab.
8. Specify the Server using your job parameter.
9. Specify the User and Password to connect to Teradata. (Get this information from your instructor.)

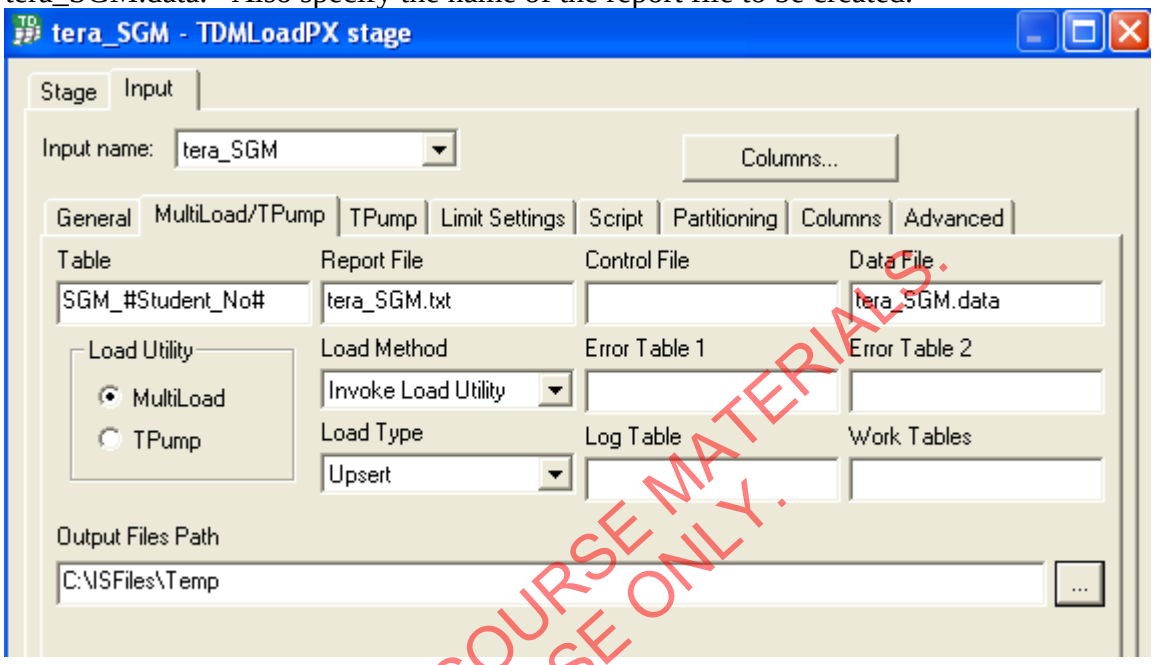


10. Click on the Input>General tab. Specify the options as shown. These options write to a data file rather than using a named pipe. The "Delete Data File After Load" is unchecked, so we can verify that the data file is created.



11. Click on the MultiLoad/Tpump tab. Select the radio button to use the MultiLoad load utility.

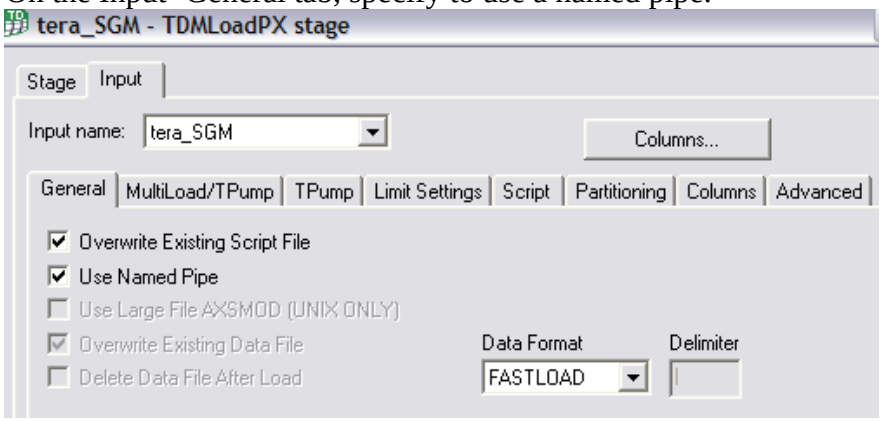
12. Write to the SGM_#Student_No# table, where "#Student_No#" is the job parameter you defined in earlier.
13. Specify the load method as "Invoke Load Utility" and specify the load type as "Upsert".
14. Specify your files directory as the Output Files Path and specify a data file, e.g., tera_SGM.data. Also specify the name of the report file to be created.



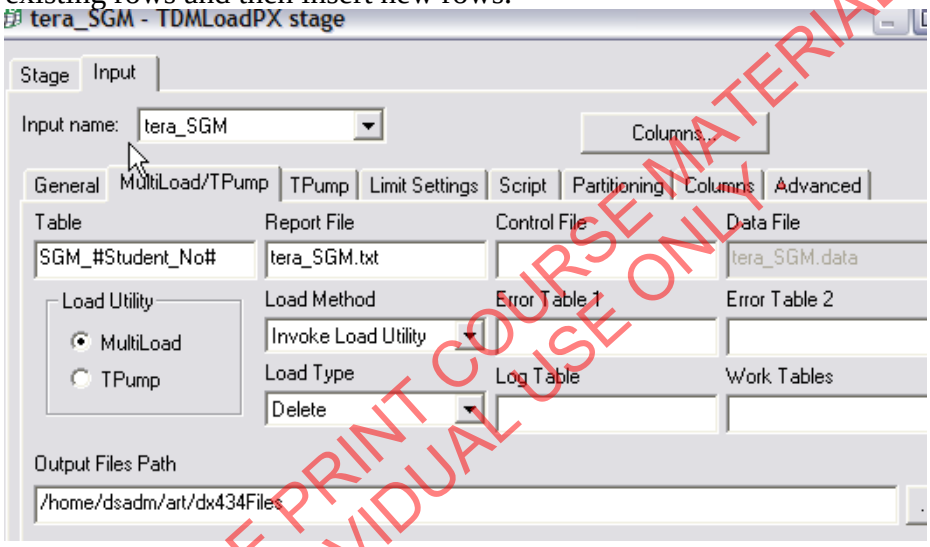
15. Compile and run the job. Check the job log. There shouldn't be any errors or warnings.

16. Log onto the directory where the report and data files were written. Open up the report file. Verify that all input records were successfully inserted or updated (depending on whether the table was empty or not).
17. Verify that the data file exists in the directory.
18. Try running the job with the following options:

19. On the Input>General tab, specify to use a named pipe.



20. On the MultLoad tab, try specifying different load types. For example, first delete the existing rows and then insert new rows.

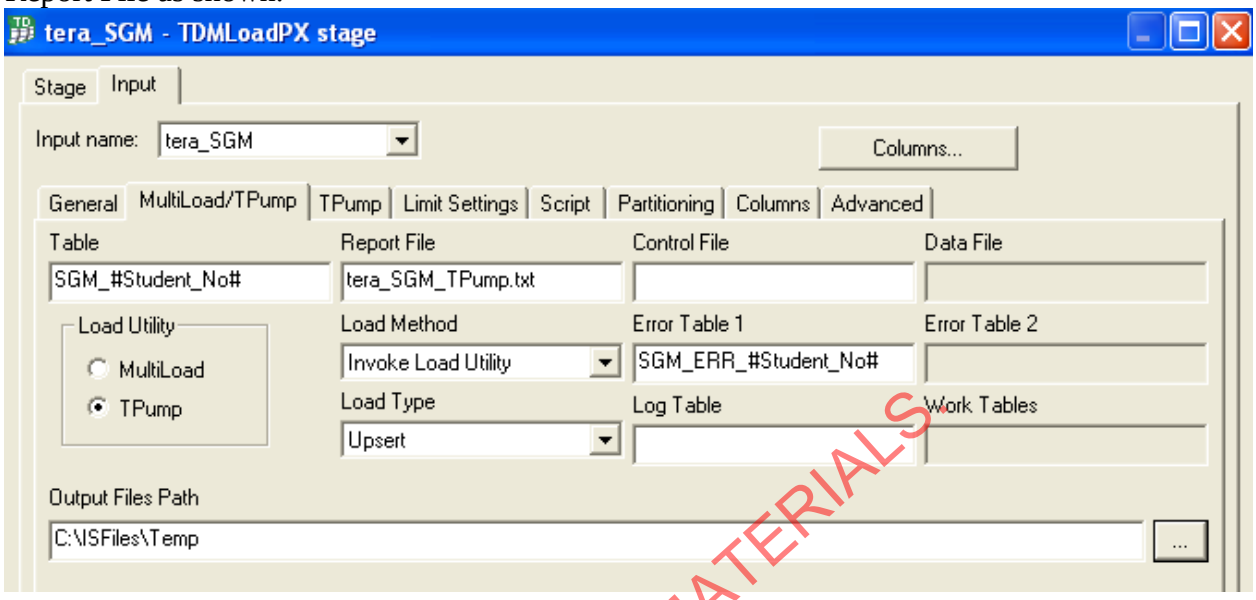


Task: Use Teradata MultiLoad stage TPump utility

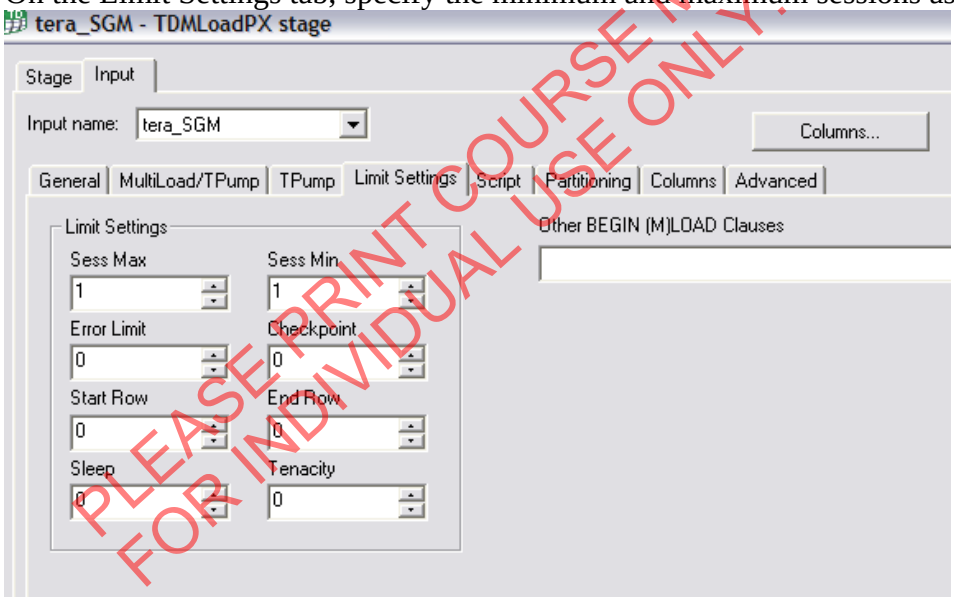
In this task you create a job that loads records from Selling_Group_Mapping.txt into a Teradata table. You will use the TPump utility to do the inserts.

1. Save your job as SGMInsertTeradata_TPump.

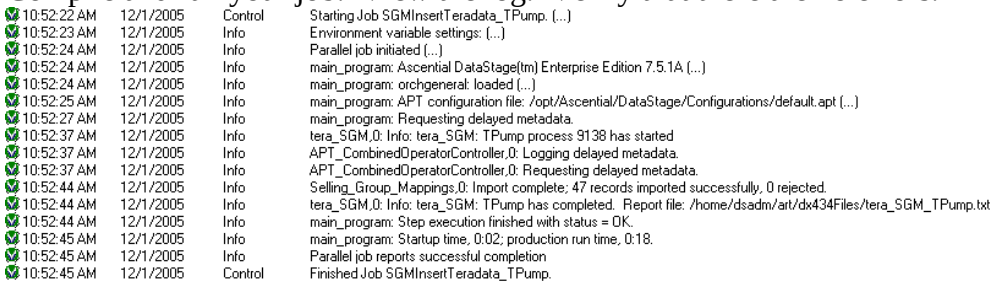
2. On the MultiLoad/TPump tab, select the TPump utility. And specify the Error Table and Report File as shown.



3. On the Limit Settings tab, specify the minimum and maximum sessions as 1.



4. Compile and run your job. View the log. Verify that there are no errors.



10:52:22 AM	12/1/2005	Control	Starting Job SGMInsertTeradata_TPump. (...)
10:52:23 AM	12/1/2005	Info	Environment variable settings: (...)
10:52:24 AM	12/1/2005	Info	Parallel job initiated (...)
10:52:24 AM	12/1/2005	Info	main_program: Ascential DataStage(tm) Enterprise Edition 7.5.1A (...)
10:52:24 AM	12/1/2005	Info	main_program: orchgeneral: loaded (...)
10:52:25 AM	12/1/2005	Info	main_program: APT configuration file: /opt/Ascential/DataStage/Configurations/default.apt (...)
10:52:27 AM	12/1/2005	Info	main_program: Requesting delayed metadata.
10:52:37 AM	12/1/2005	Info	tera_SGM,0: Info: tera_SGM: TPump process 9138 has started
10:52:37 AM	12/1/2005	Info	APT_CombinedOperatorController,0: Logging delayed metadata.
10:52:37 AM	12/1/2005	Info	APT_CombinedOperatorController,0: Requesting delayed metadata.
10:52:44 AM	12/1/2005	Info	Selling_Group_Mappings,0: Import complete; 47 records imported successfully, 0 rejected.
10:52:44 AM	12/1/2005	Info	tera_SGM,0: Info: tera_SGM: TPump has completed. Report file: /home/dsadm/art/dx434Files/tera_SGM_TPump.txt
10:52:44 AM	12/1/2005	Info	main_program: Step execution finished with status = OK.
10:52:45 AM	12/1/2005	Info	main_program: Startup time, 0:02; production run time, 0:18.
10:52:45 AM	12/1/2005	Info	Parallel job reports successful completion
10:52:45 AM	12/1/2005	Control	Finished Job SGMInsertTeradata_TPump.

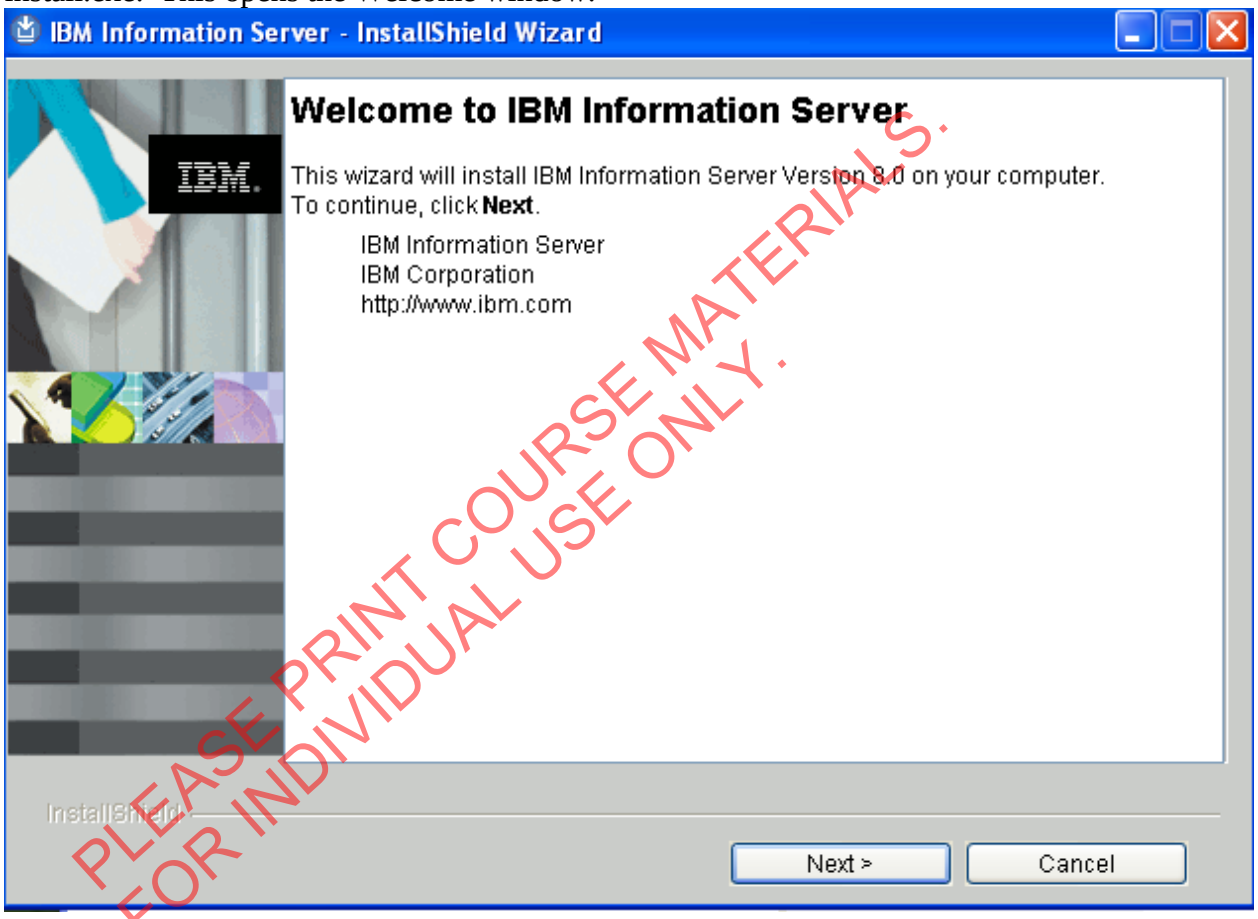
5. View and examine the report file.

Special Topic 6: Installation and Startup

Task: Install Information Server

The steps below describe the process for installing Information Server on Windows. Everything will be installed on one machine. The process for installing on Linux is similar.

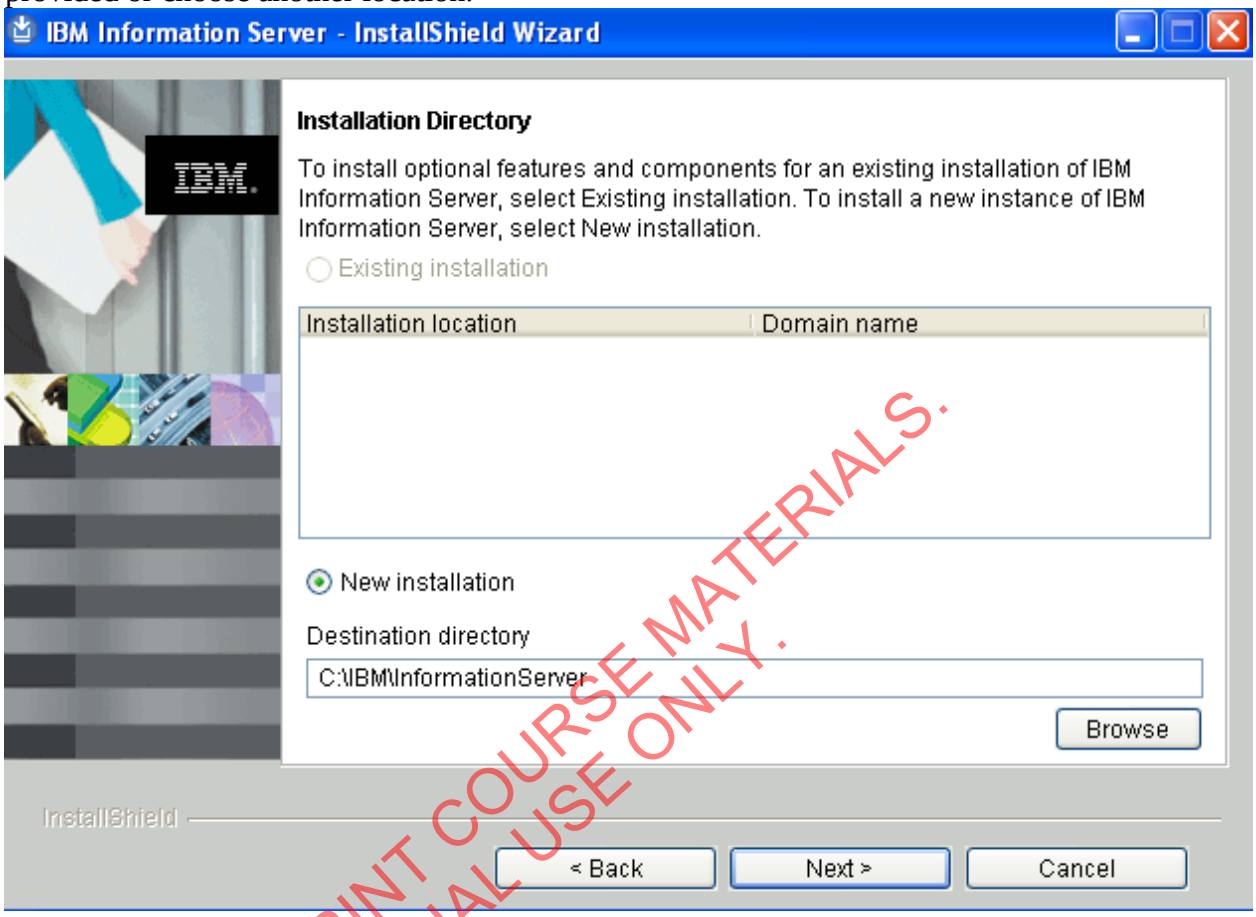
1. On Windows, move to the directory containing the installation image. Double-click install.exe. This opens the Welcome window.



2. Click Next to open the License Agreement window. Accept the license agreement terms.

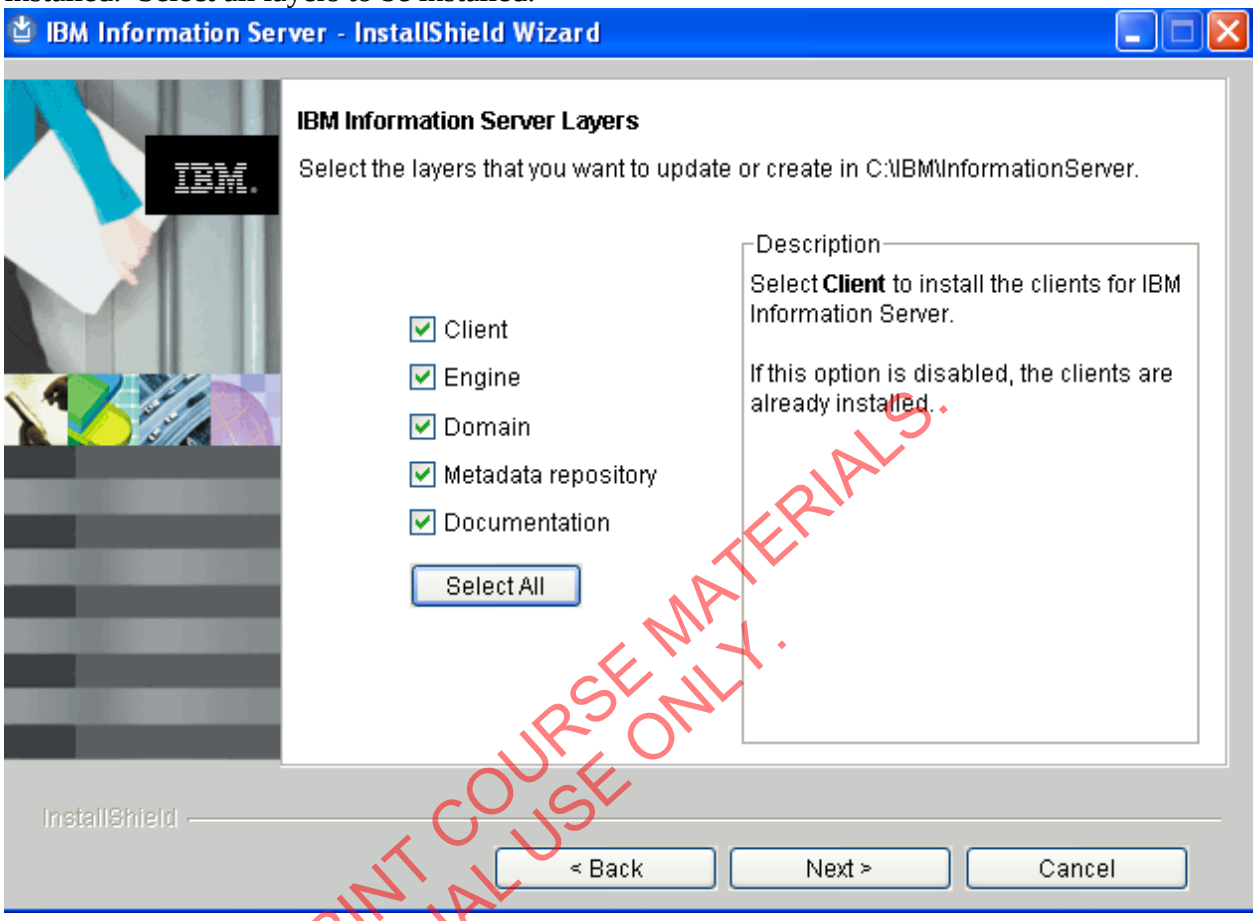
3. Click Next to specify whether a response file is to be created during the installation.

4. Click Next to specify the installation directory location. Accept the default location provided or choose another location.

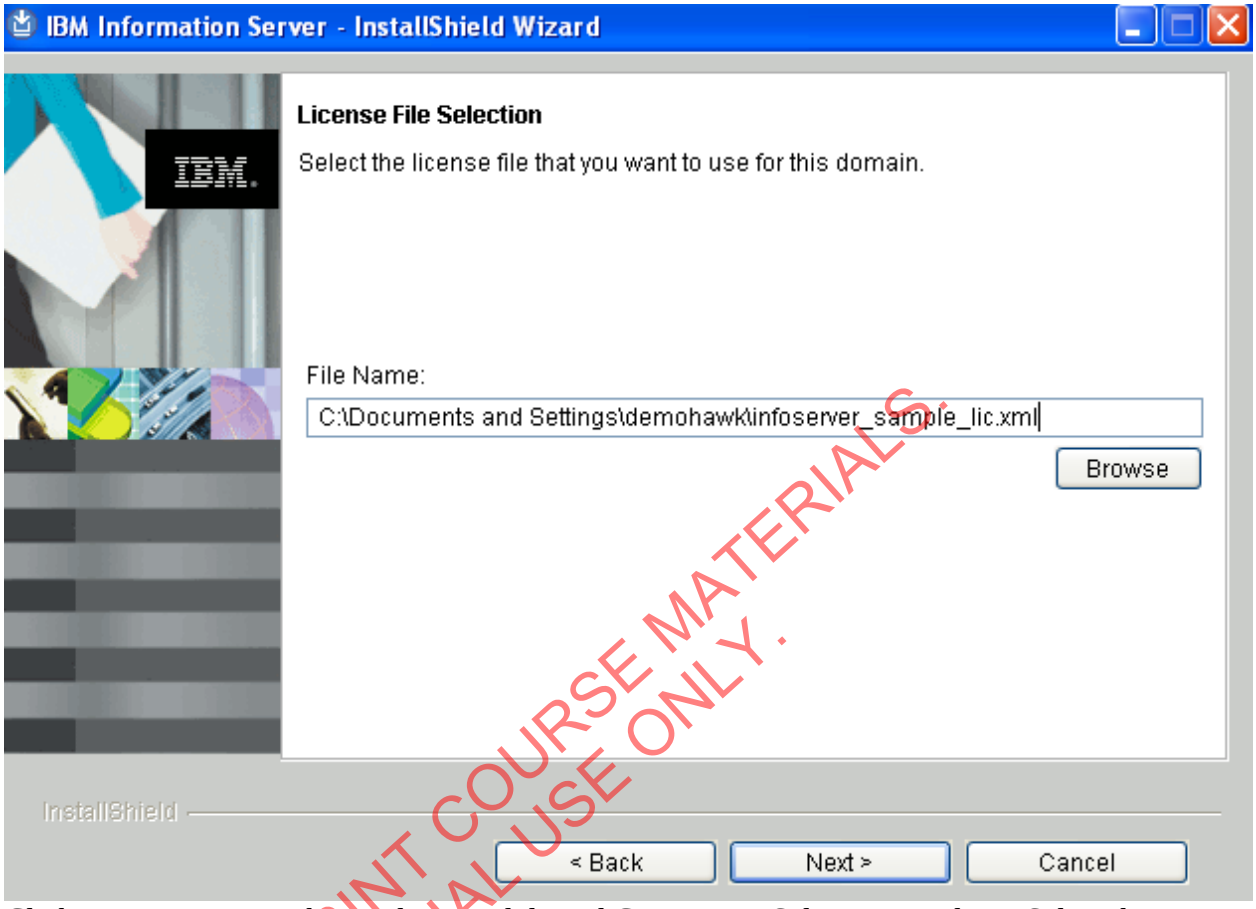


5. Click Next to select the component layers to install. Select all types. This will install all Information Server components and applications on a single machine.

6. Click Next to specify which layers of the Information Server product should be installed. Select all layers to be installed.

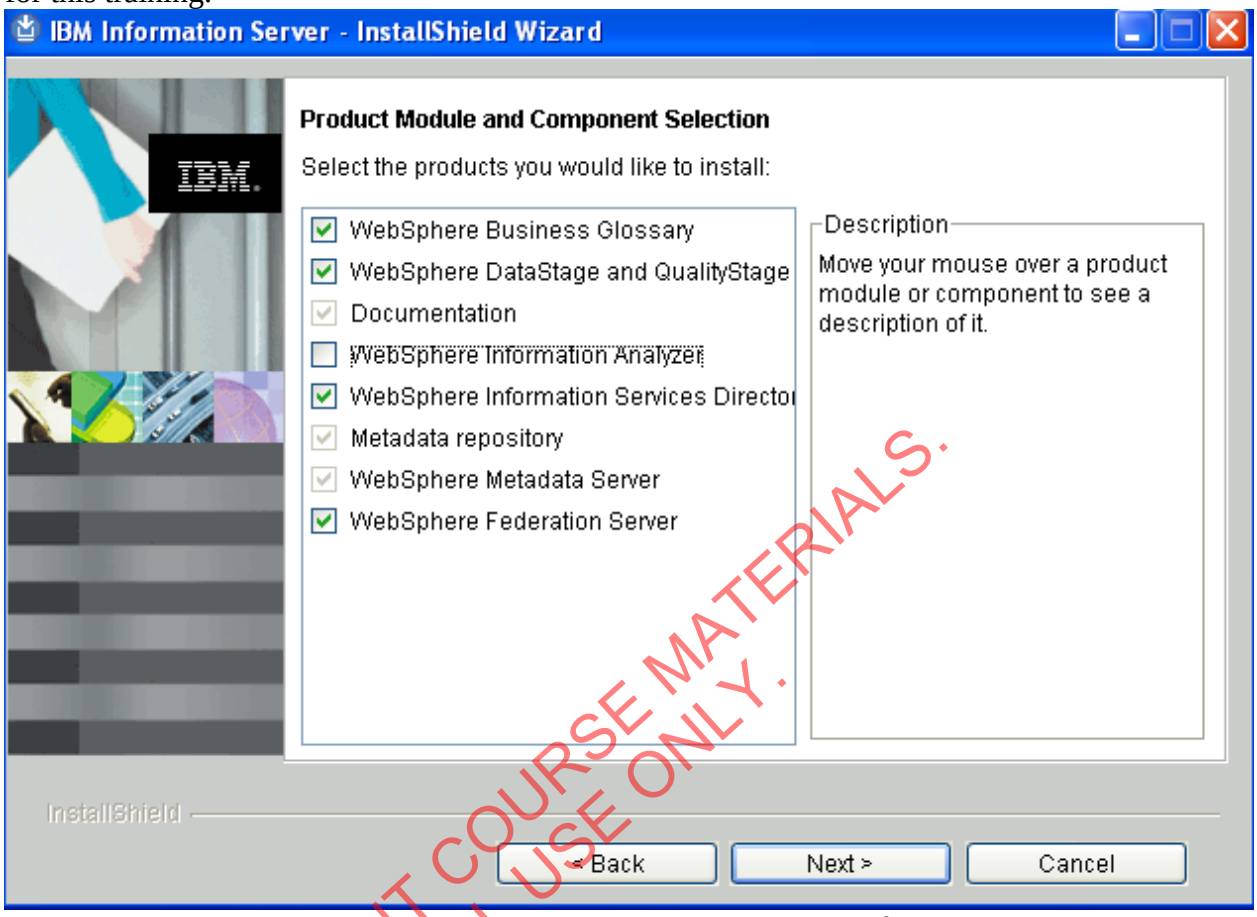


7. Click Next to move to the License File Selection window. Select the license file you want to use for the domain.



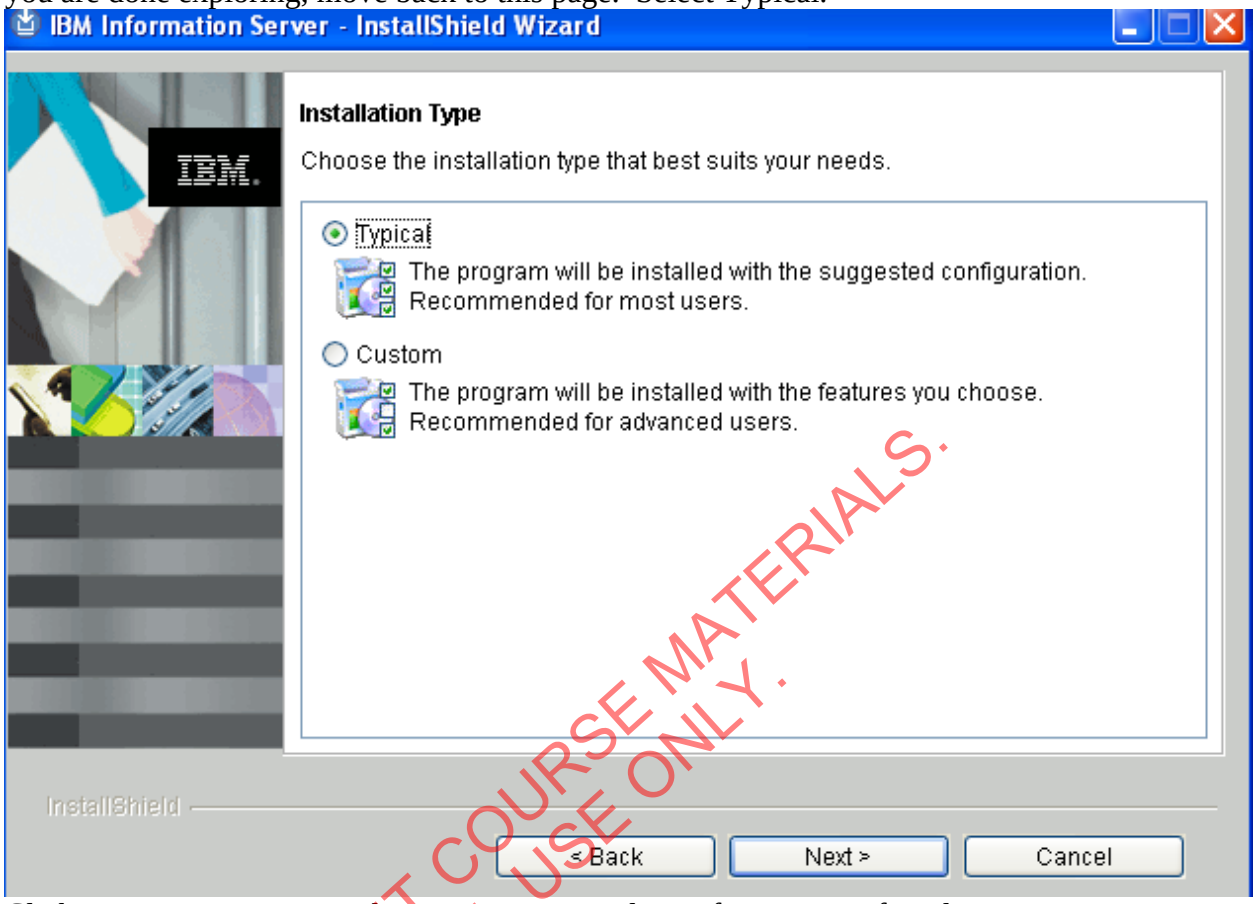
8. Click Next to move to the Product Module and Component Selection window. Select the products you would like to install. Be sure to specify at least DataStage and QualityStage

for this training.



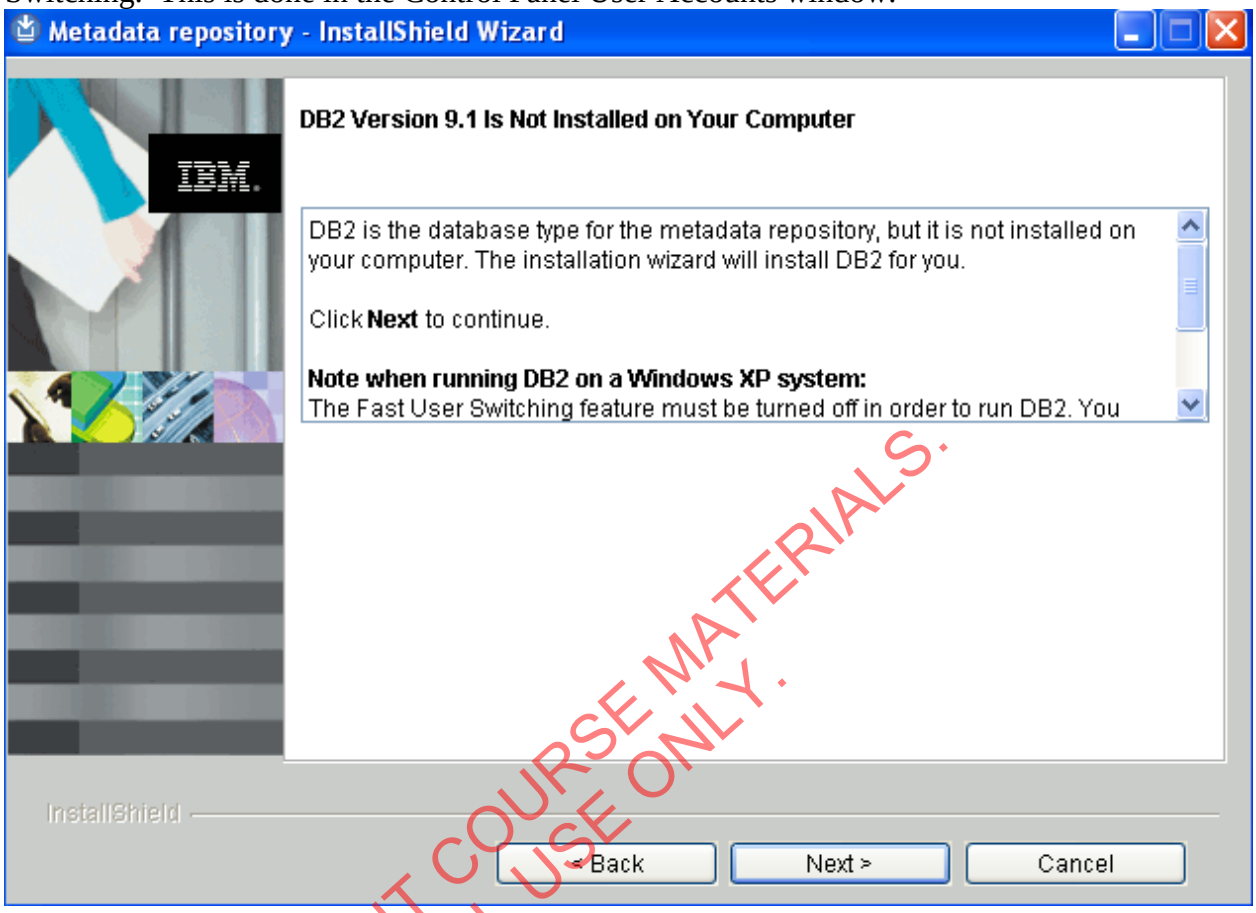
9. Click Next to move to the Installation Type window. For the sake of exploration, select the Custom installation type and then click Next. Examine the options you have. When

you are done exploring, move back to this page. Select Typical.

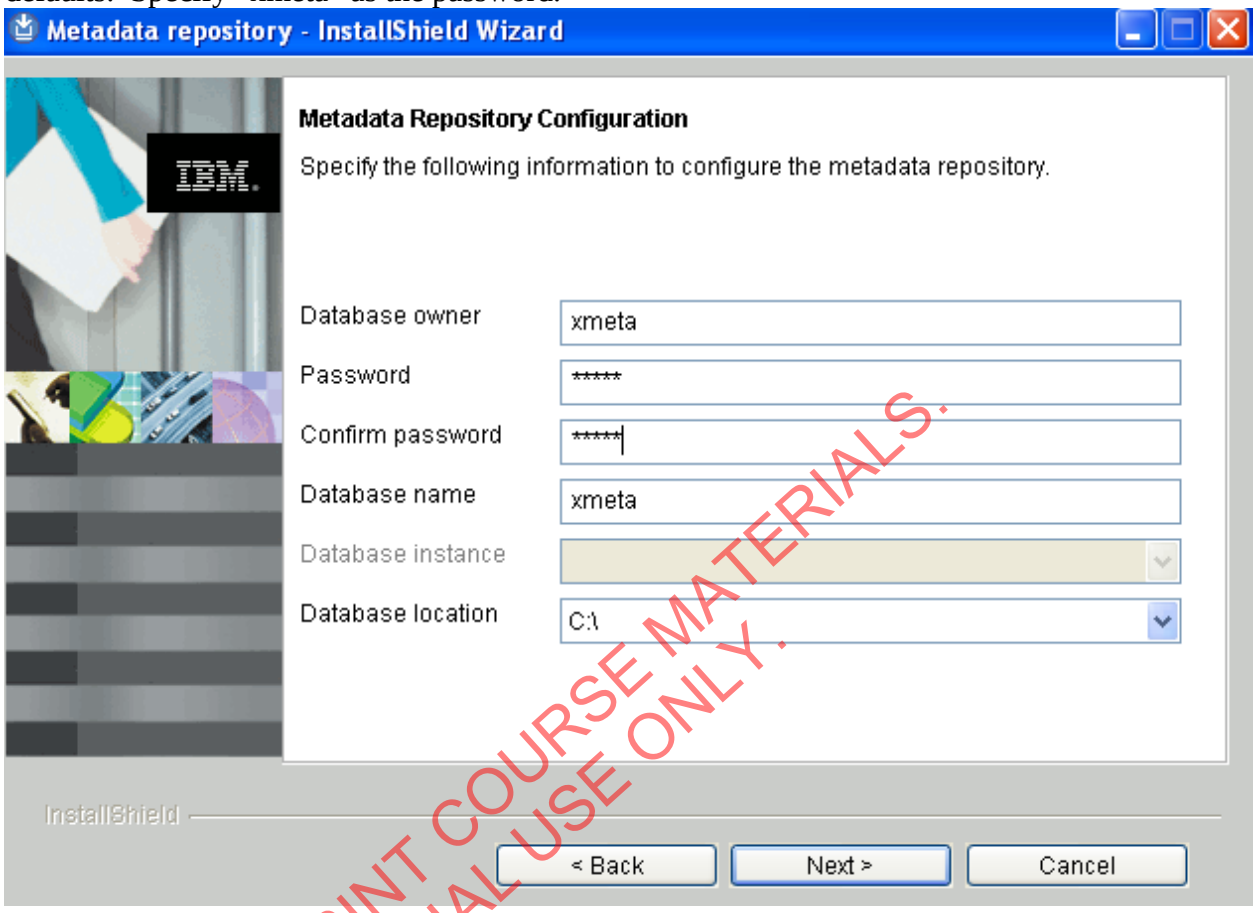


10. Click Next to move to Metadata Repository window. If DB2 is not found on your system, it will be installed. If DB2 already exists on your system, you will be given options to configure it. Be sure to read the note about truning off the Fast User

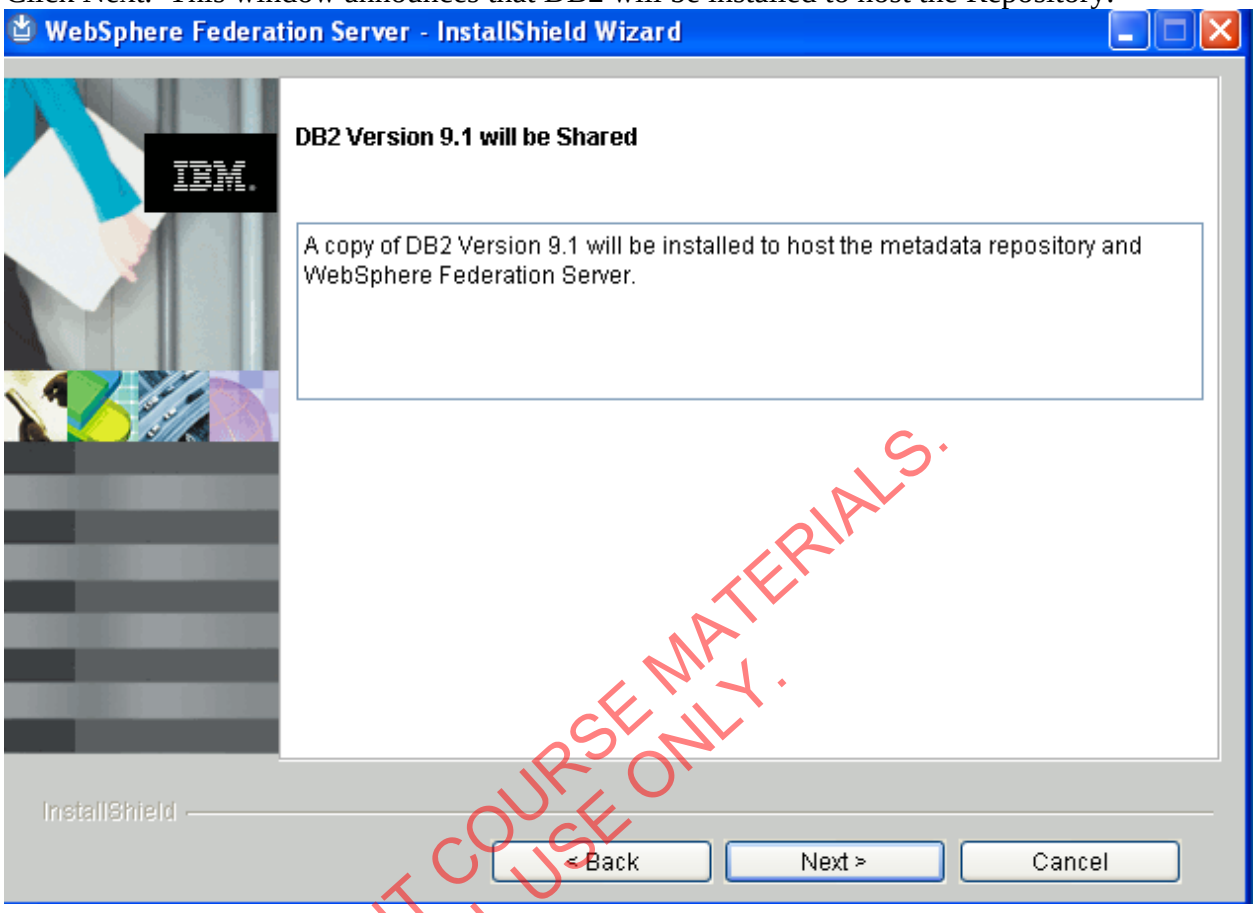
Switching. This is done in the Control Panel User Accounts window.



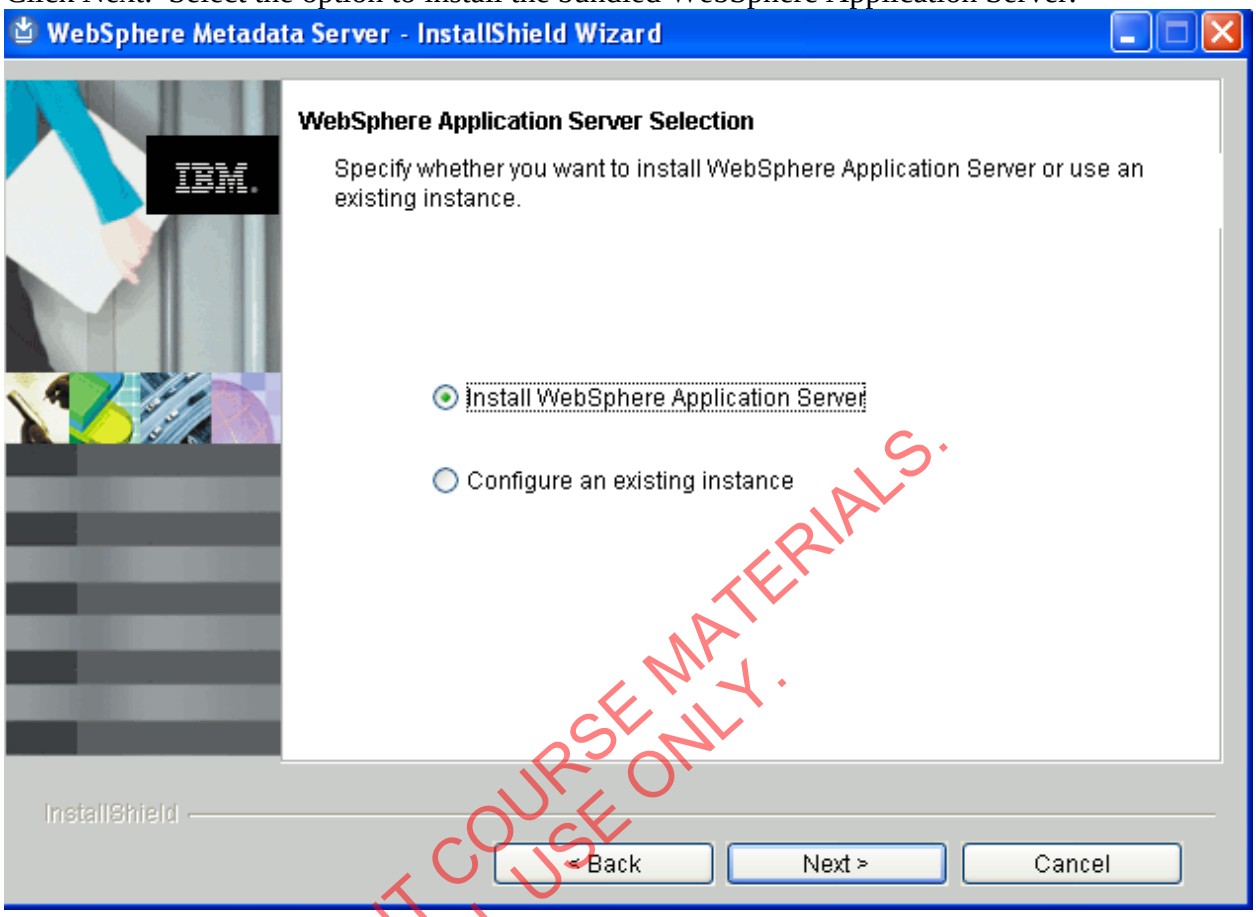
11. Click Next. This displays the Metadata Repository Configuration window. Accept the defaults. Specify “xmeta” as the password.



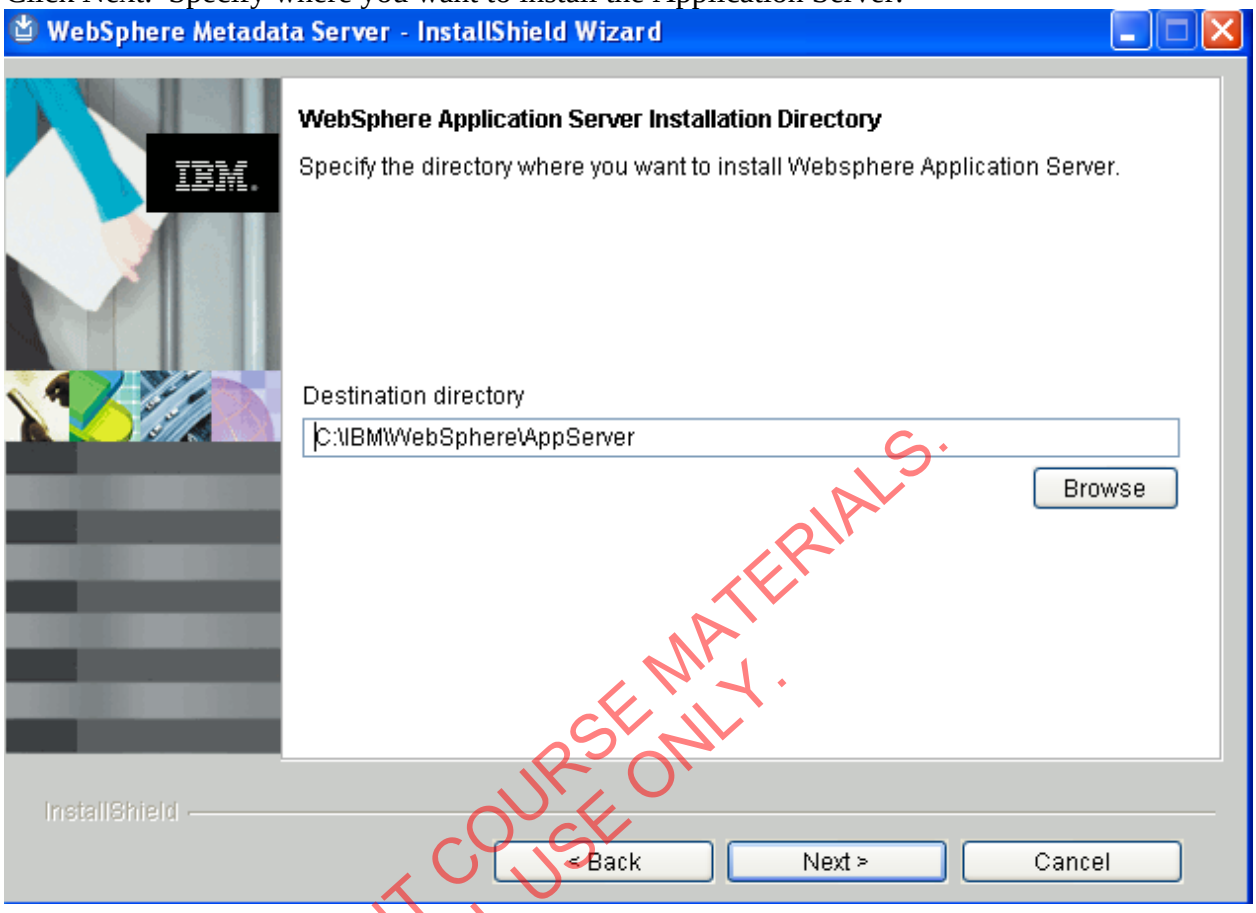
12. Click Next. This window announces that DB2 will be installed to host the Repository.



13. Click Next. Select the option to install the bundled WebSphere Application Server.

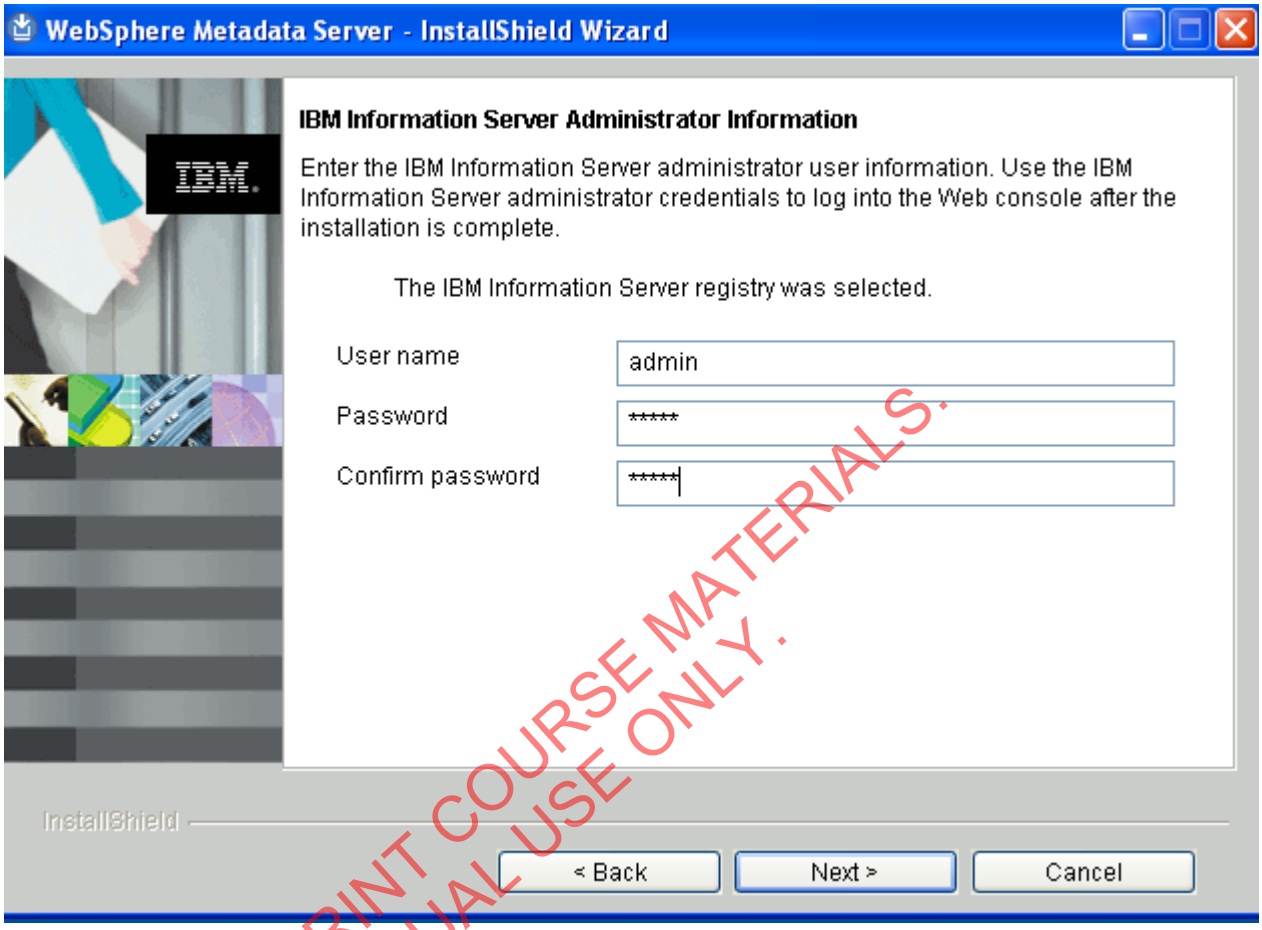


14. Click Next. Specify where you want to install the Application Server.

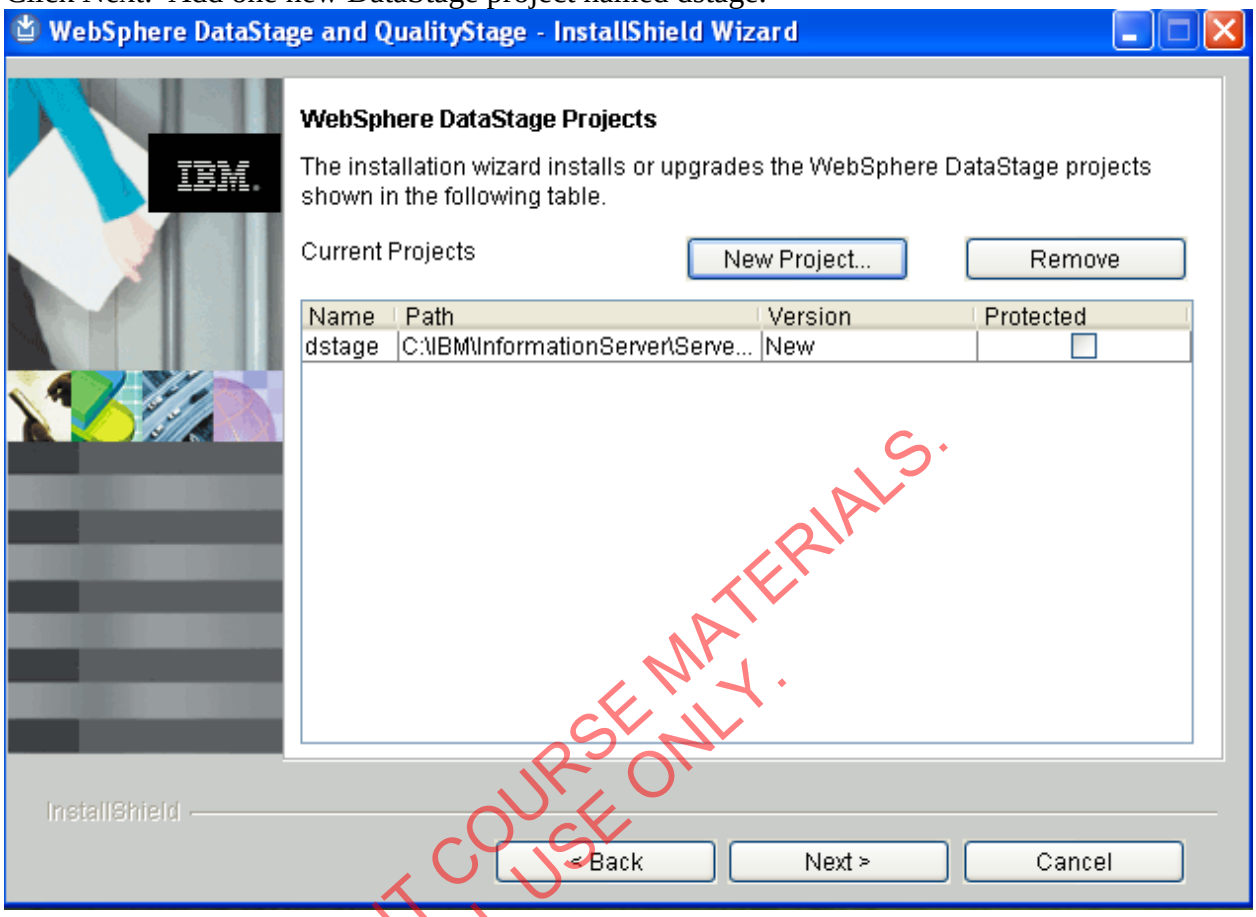


15. Click Next. Specify the Application Server credentials. User appserv for both user ID and password. Accept the default style of user directory.

16. Click Next. Specify admin as the Information Server administrator. Password is "admin".

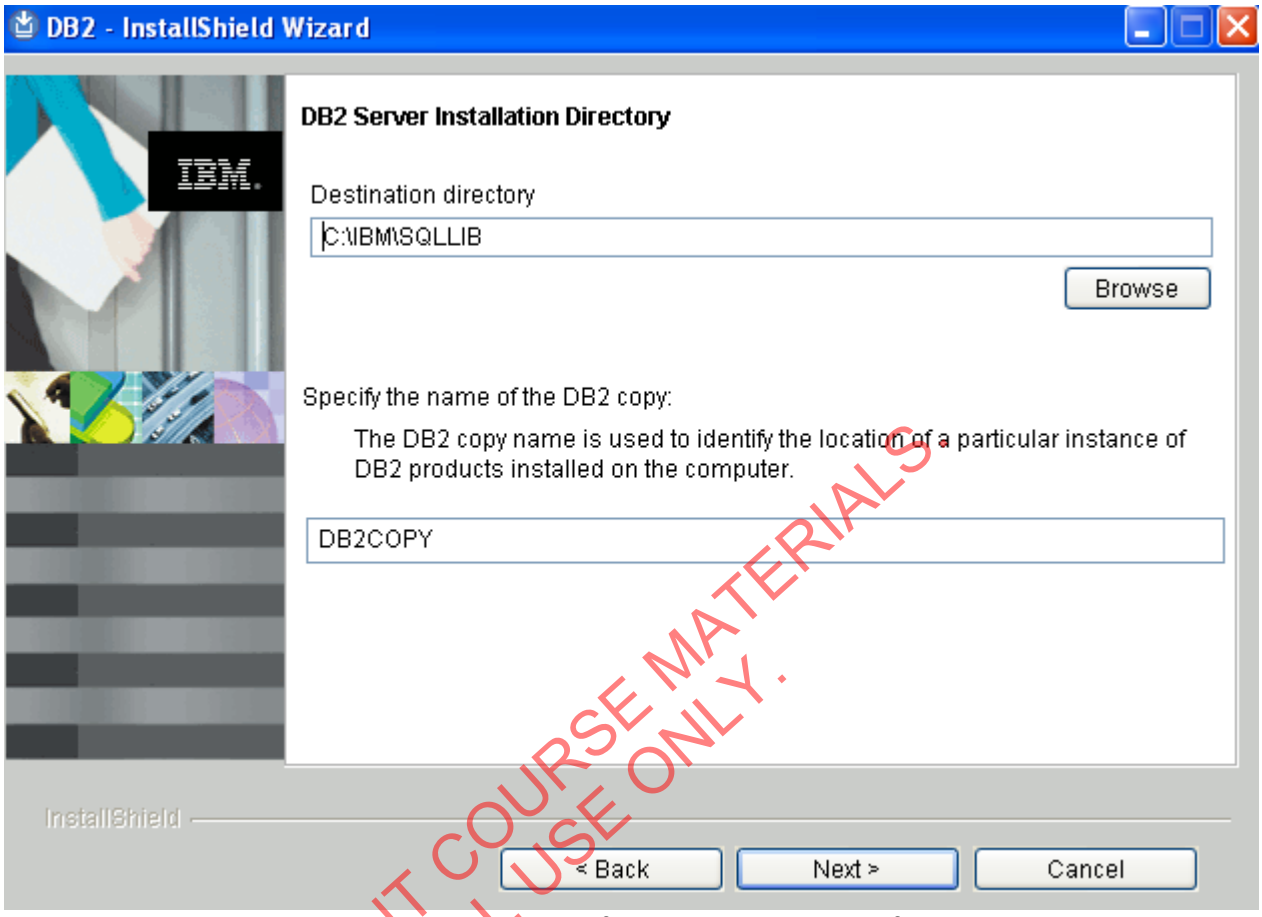


17. Click Next. Add one new DataStage project named dstage.



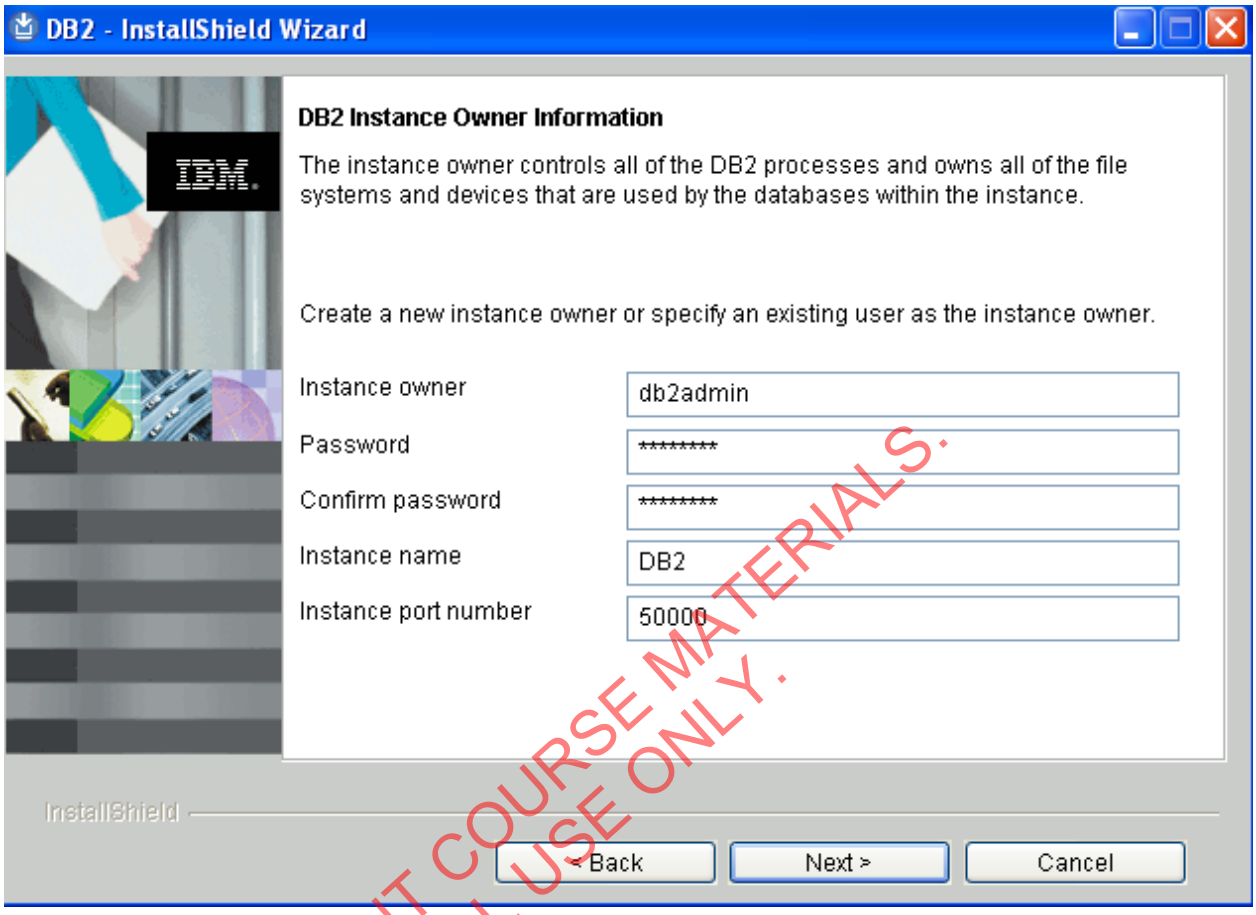
18. Click Next to move to the Languages window.

19. Click Next. Enter the location of the database server.

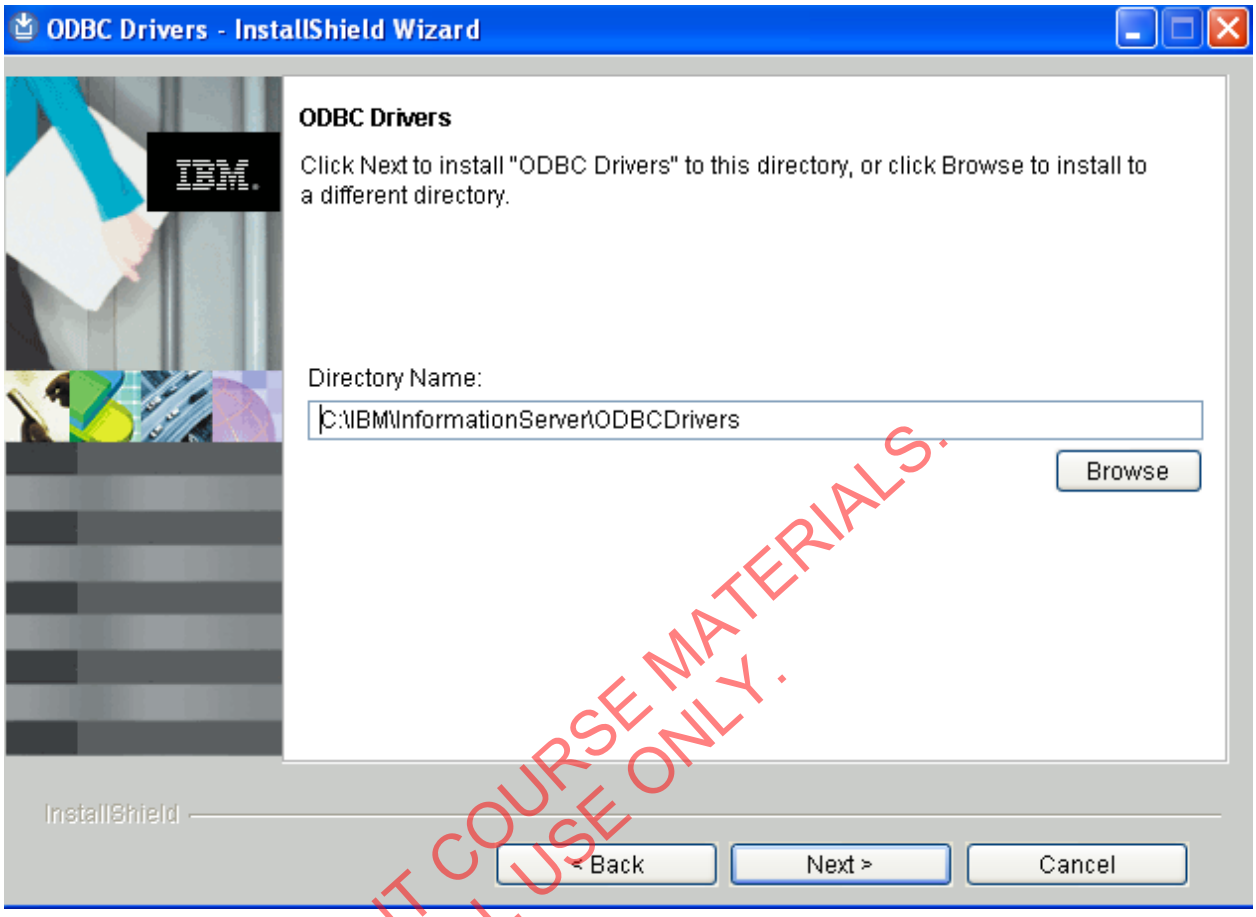


20. Click Next. Enter the DB2 Instance Owner Information. Enter the information as shown. Password is the same as the user

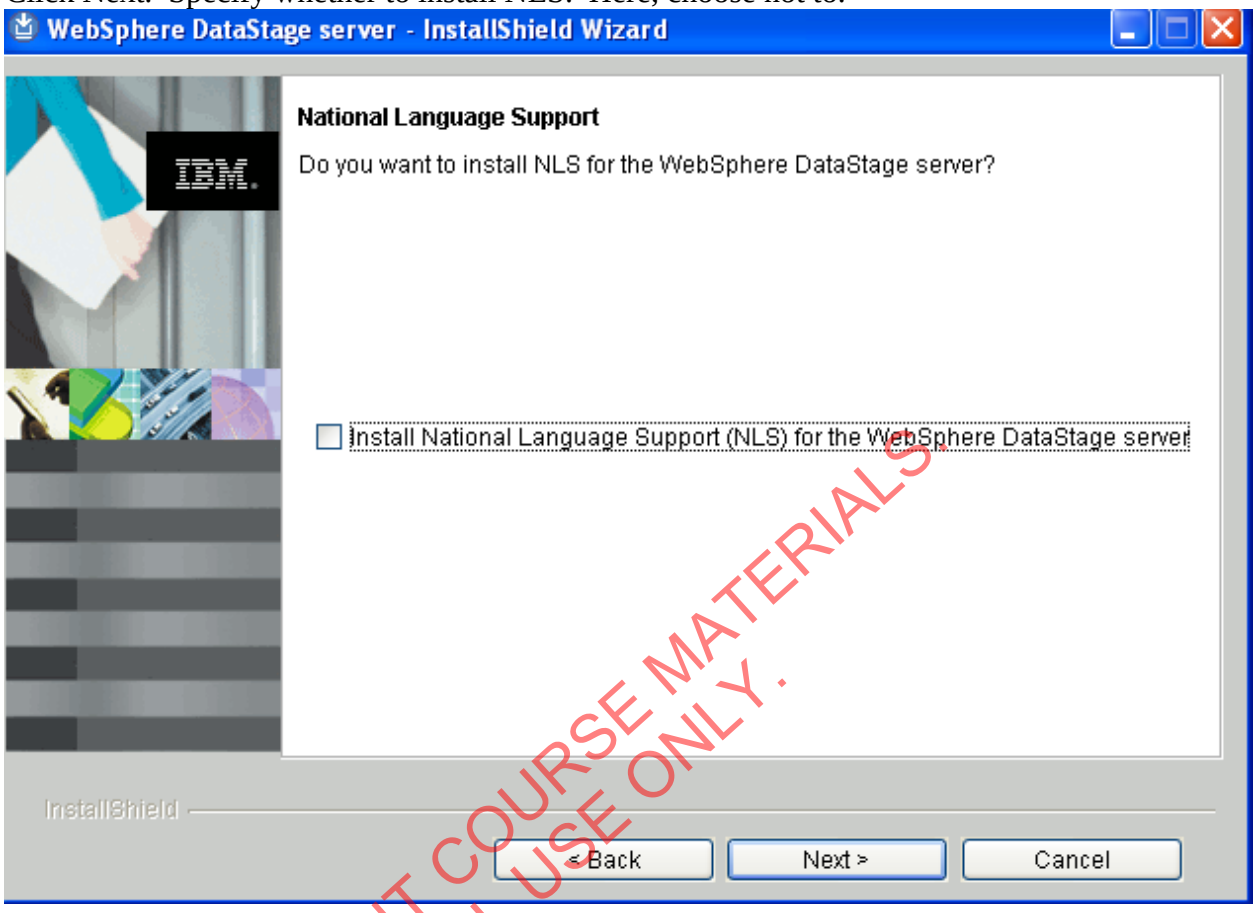
name.



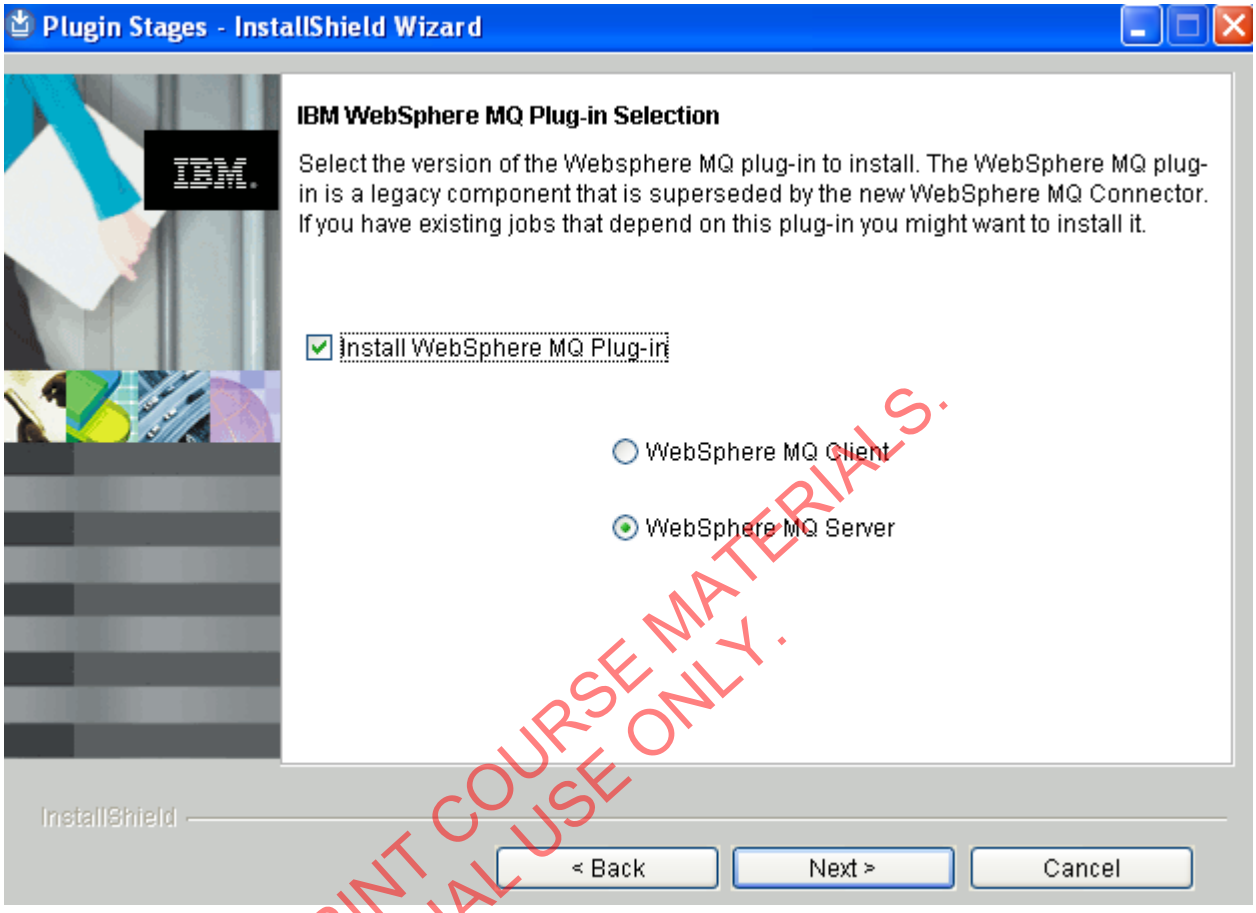
21. Click Next. Enter the location for the ODBC drivers.



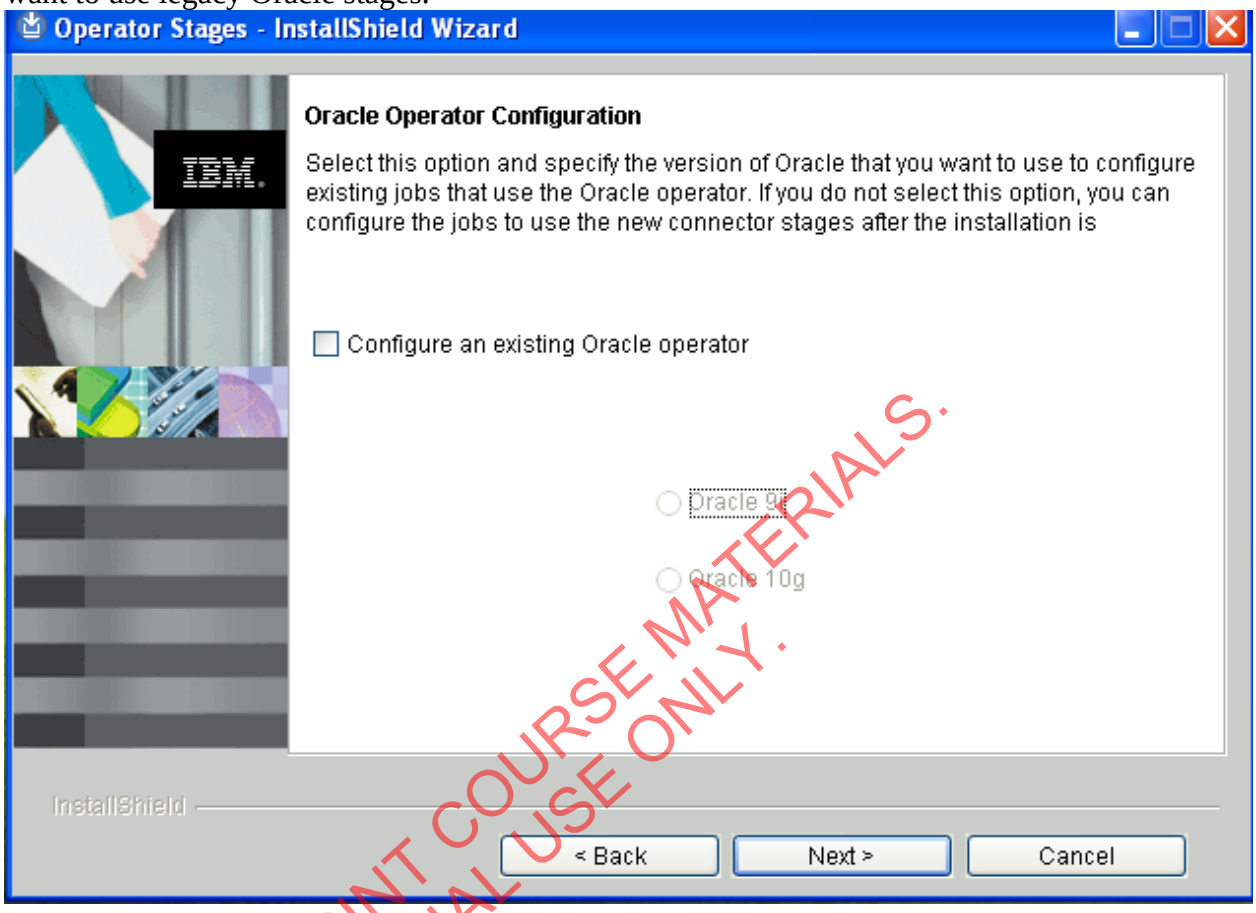
22. Click Next. Specify whether to install NLS. Here, choose not to.



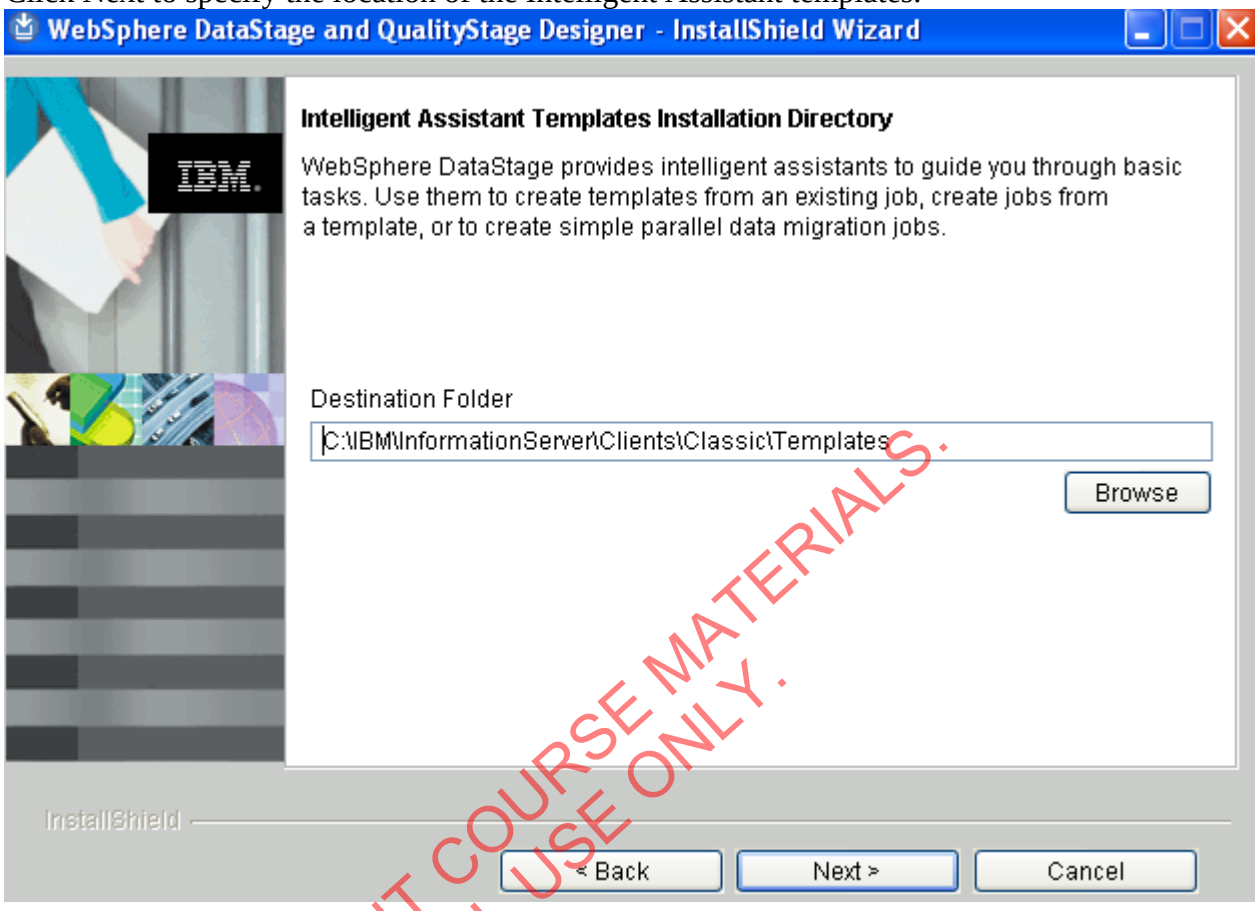
23. Click Next. Specify whether to install the WebSphere MQ Plugin. Go ahead and select it.



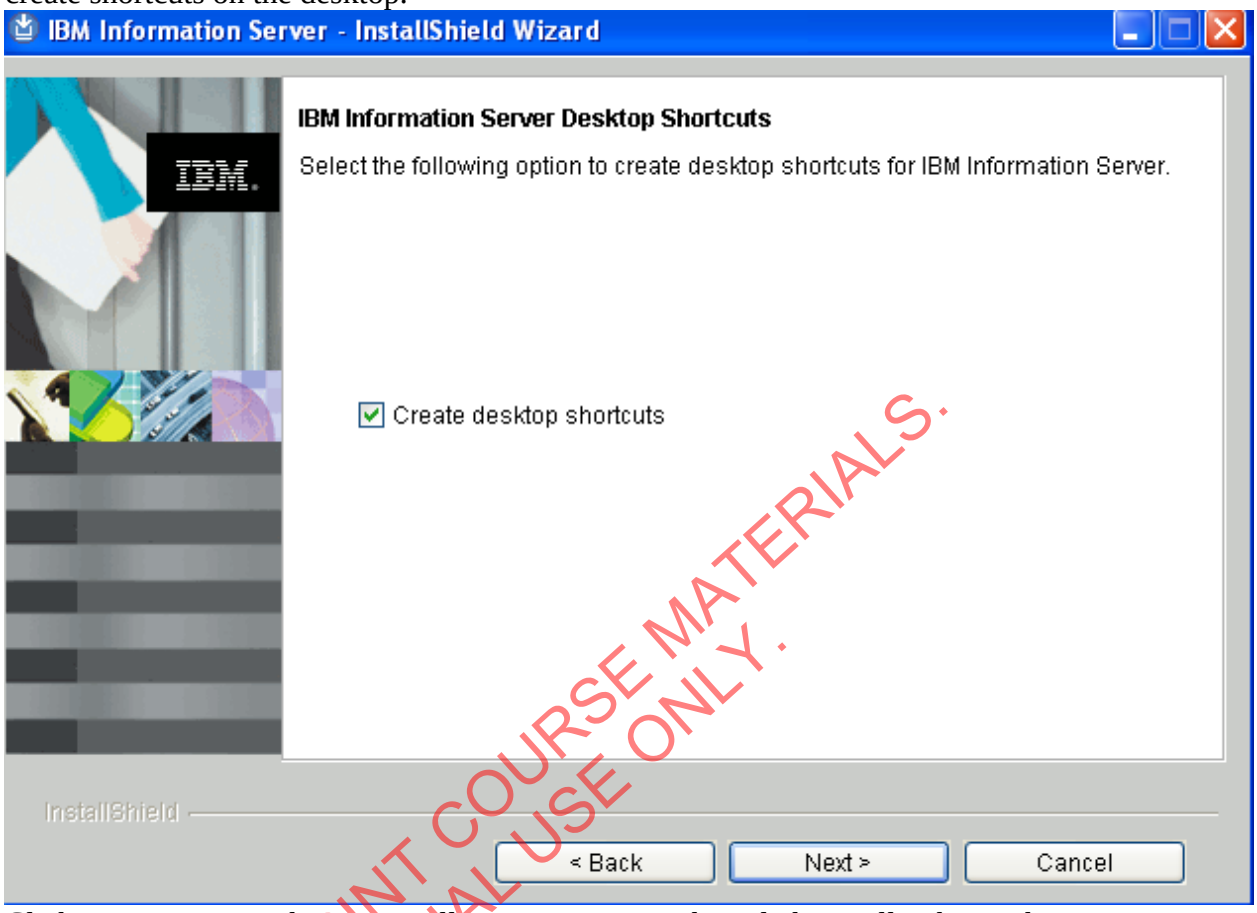
24. Click Next to move to the Oracle Operator Configuration window. Select the box if you want to use legacy Oracle stages.



25. Click Next to specify the location of the Intelligent Assistant templates.



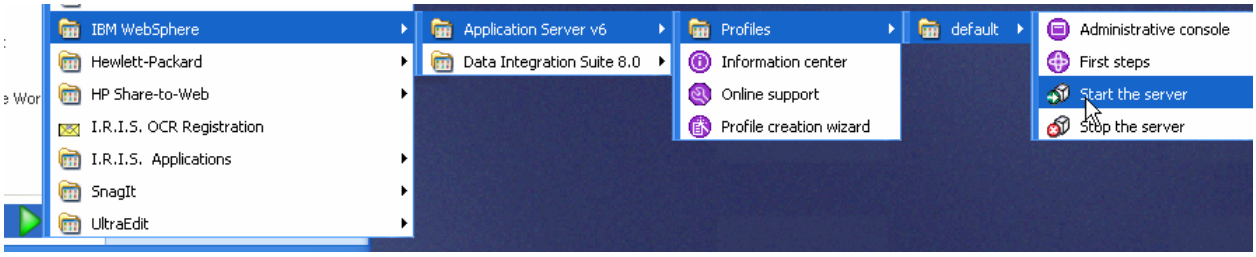
26. Click Next to move to the Information Server Desktop Shortcuts window. Choose to create shortcuts on the desktop.



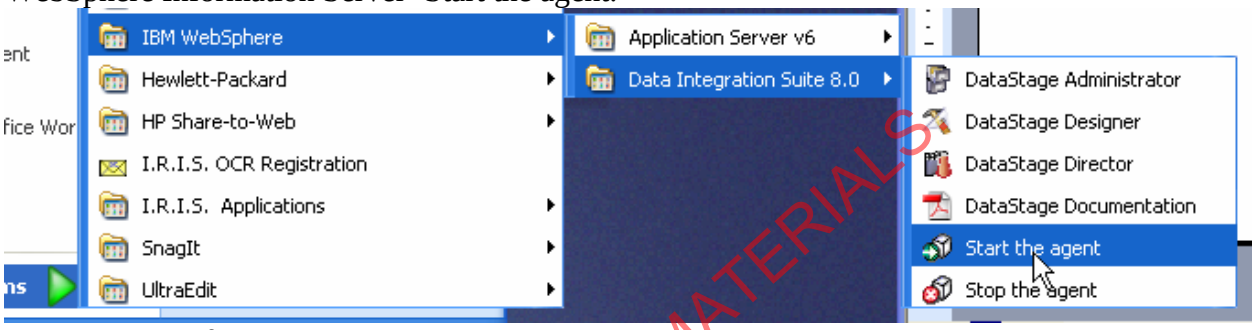
27. Click Next to review the pre-installation summary. Then click Install to begin the installation process.
28. After installation, restart your computer.

Task: Test the installation

3. After installation, by default, the following should start automatically. You can verify this by examining your services, available in Windows Control Panel>Administrative services.
   - IBM WebSphere Application Server
   - ASB agent
   - DB2
4. If not already started, start the Application Server. To accomplish this on Windows, click IBM WebSphere>Application Server>Profiles>default>Start the server.



5. If not already started, start the ASB agent. To accomplish this on Windows, click IBM WebSphere Information Server>Start the agent.



6. Log onto the Information Server Administration console. From the WebSphere Information Server menu select WebSphere Information Server Web Console. Then enter the login information. The default user ID and password after installation is admin / admin.

Special Topic 7: Solution Development Example

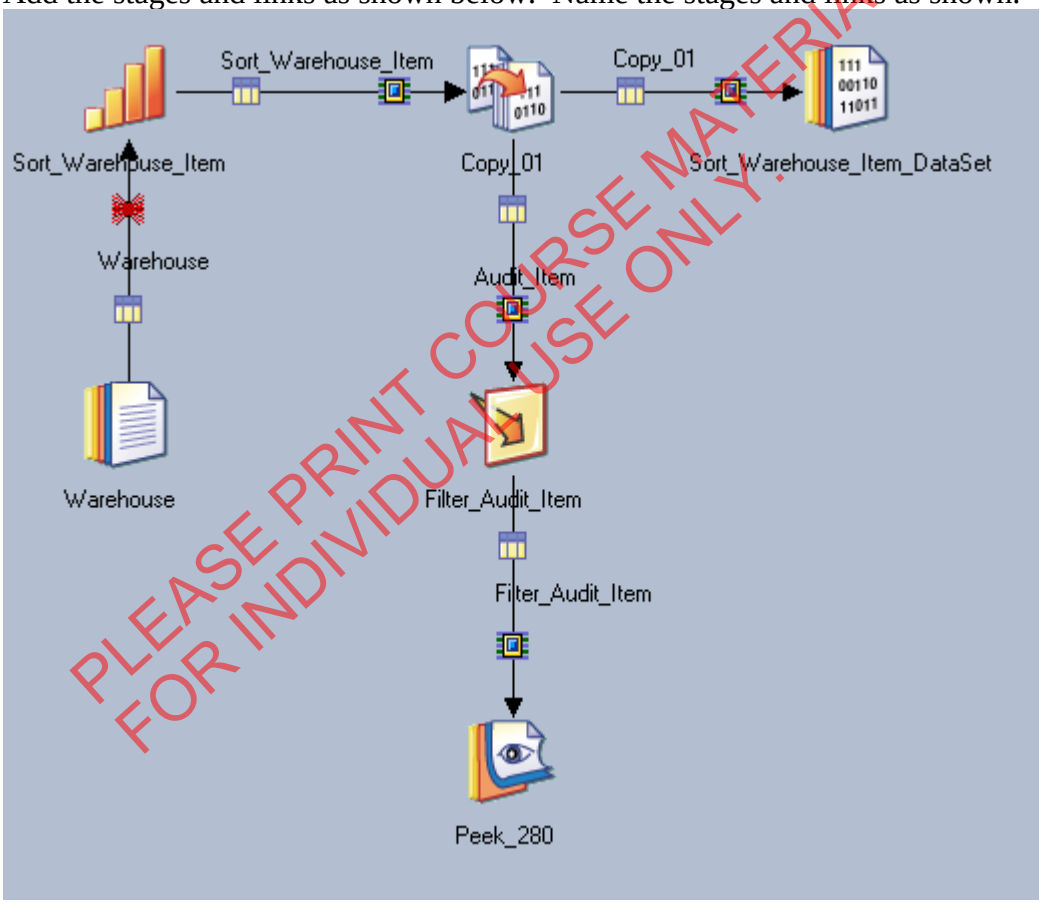
Build Warehouse_01 Job

Assumptions

- The dx444Files directory on the DataStage server machine contains the following files: warehouse_031005_01.txt, warehouse_031005_02.txt, warehouse_031005_03.txt, warehouse_031105_01.txt, and warehouse_031105_02.txt.

Task: Create the job design

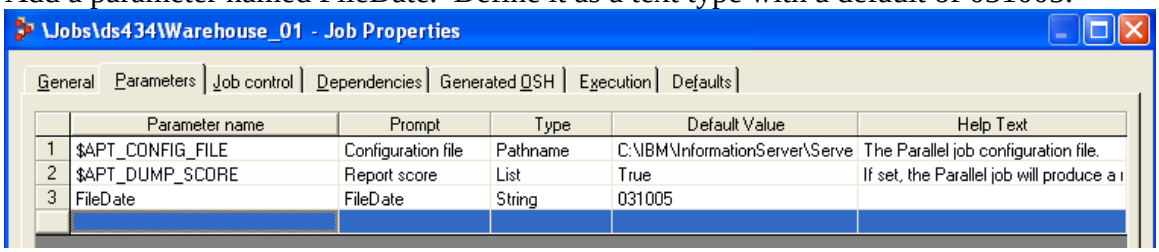
1. Open a new Parallel job and save it under the name Warehouse_01. Store it in your _Training>Jobs folder.
2. Add the stages and links as shown below. Name the stages and links as shown.



Task: Add job parameters

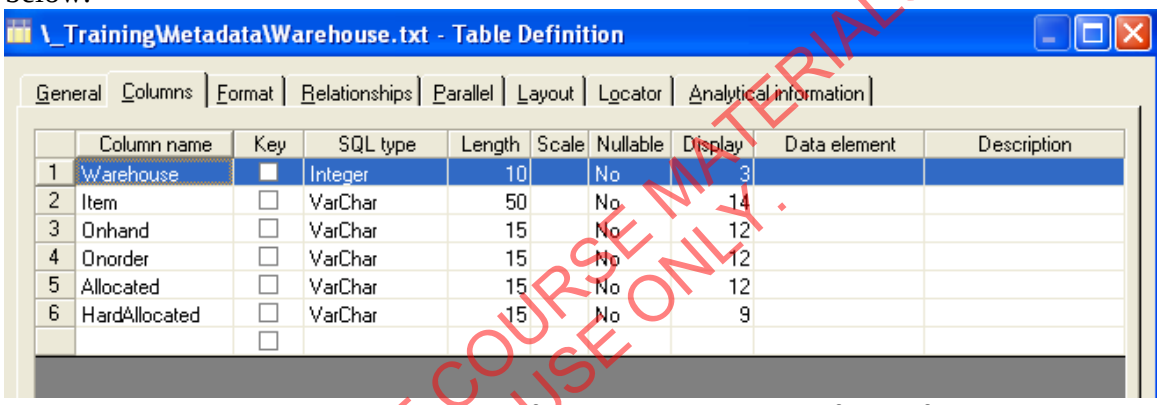
1. Open up the Job Properties window and click on the Parameters tab.
2. Click on the Add Environment Variable button and add the APT_CONFIG_FILE and APT_DUMP_SCORE variables. Accept the defaults that are added.

3. Add a parameter named FileDate. Define it as a text type with a default of 031005.



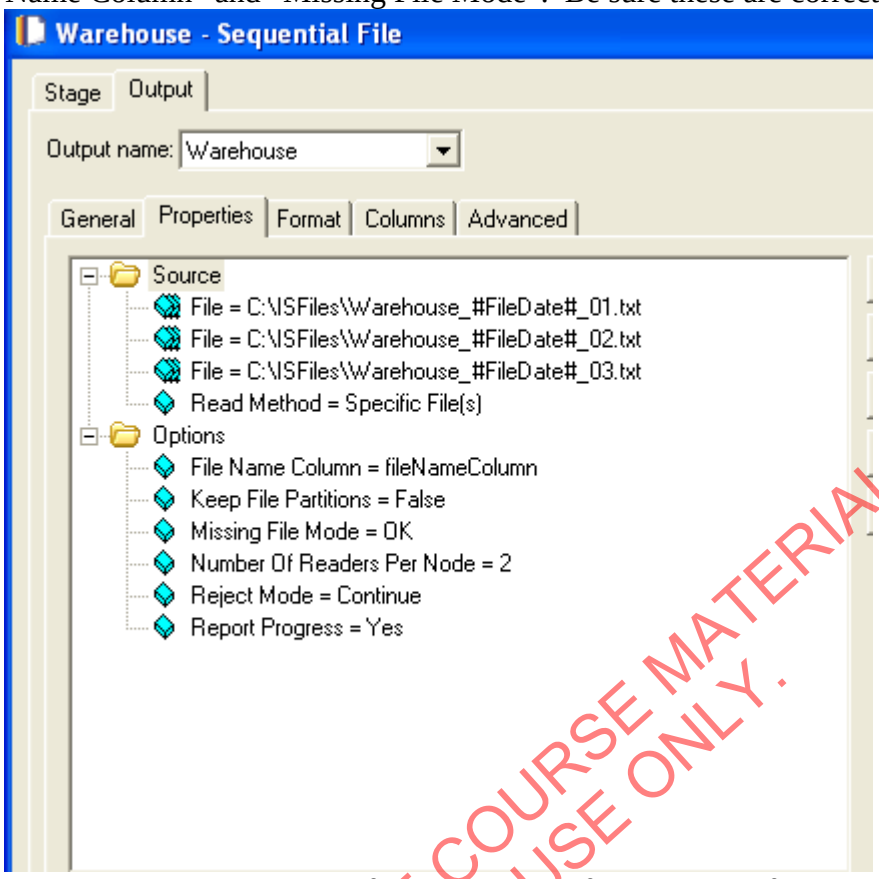
Task: Edit the source Sequential stage

1. Import the Table Definition for the Warehouse.txt sequential file. Use the types shown below.

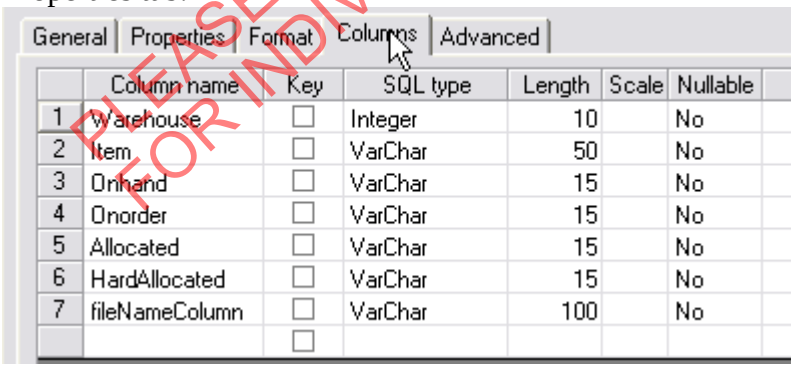


2. Examine the Warehouse_031005_01.txt file to see an example of one of the source data files your job will be reading.
3. Open up your Sequential source stage. On the Properties tab, specify that all the Warehouse_<File Date>_#File Number#.txt files for the date specified by the FileDate job parameter are to be read. Assume that there is a maximum of three of these files numbered 01, 02, 03. This can be handled by specifically listing all three files and specifying OK for the "Missing File Mode" property.
4. Also on the Properties tab, specify that the name of the file is to be added as a column. And specify that the number of readers per node is 2. Note below the values for "File

Name Column" and "Missing File Mode". Be sure these are correct.

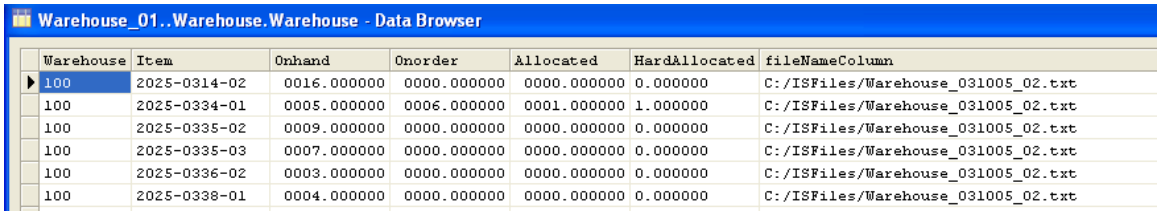


5. On the Format tab, load the format metadata for the source files.
6. On the Columns tab, load the column definitions for the source files. Add the fileNameColumn to the bottom of the columns list. Note that column names are case sensitive. This must match exactly the name of the "File Name Column" specified on the Properties tab.



	Column name	Key	SQL type	Length	Scale	Nullable
1	Warehouse	☐	Integer	10		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	fileNameColumn	☐	VarChar	100		No
		☐				

7. View the data to verify that you can read it using the stage. Verify that the file name is shown.

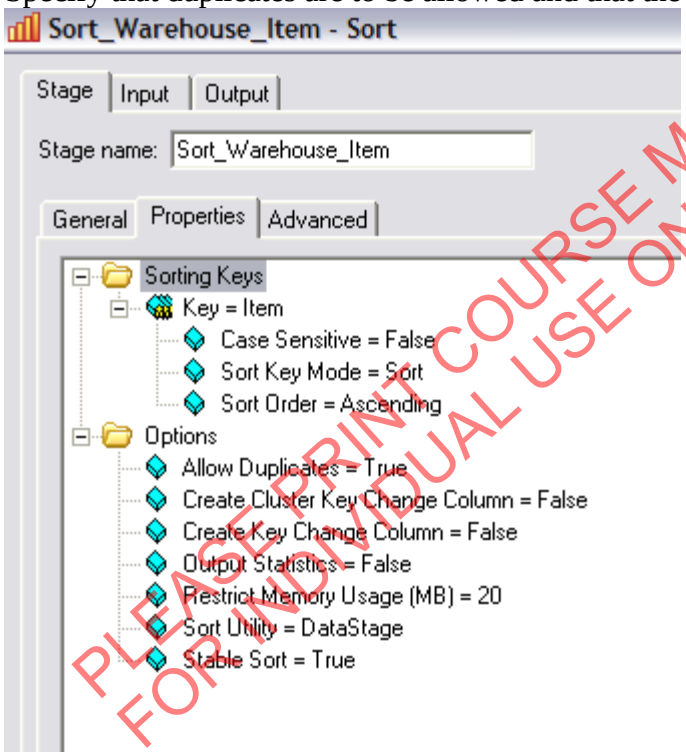


Warehouse	Item	Onhand	Onorder	Allocated	HardAllocated	fileNameColumn
100	2025-0314-02	0016.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt
100	2025-0334-01	0005.000000	0006.000000	0001.000000	1.000000	C:/ISFiles/Warehouse_031005_02.txt
100	2025-0335-02	0009.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt
100	2025-0335-03	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt
100	2025-0336-02	0003.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt
100	2025-0338-01	0004.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt

Task: Edit the Sort stage

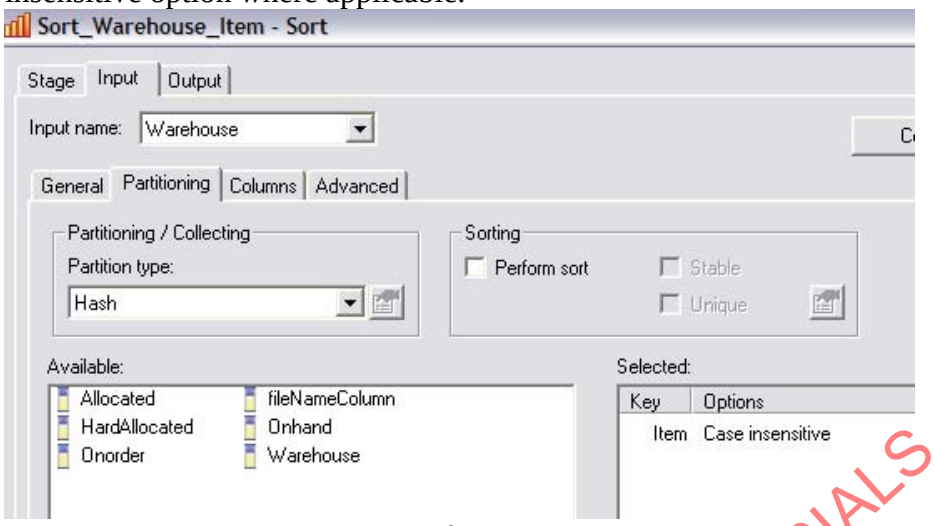
The Sort stage is used to sort the records by the Item field.

1. Open up the Sort stage. On the Properties tab, specify that the data is to be sorted by the Item field in ascending order. Make the sort case insensitive.
2. Specify that duplicates are to be allowed and that the sort is to be Stable.



3. On the Input, Partitioning tab, specify that records are to be partitioned using a Hash algorithm. The partitioning key is the Item column, as shown below. Select the Case

insensitive option where applicable.



4. On the Output>Mapping tab specify that all columns move through the stage.

Task: Edit the Copy stage

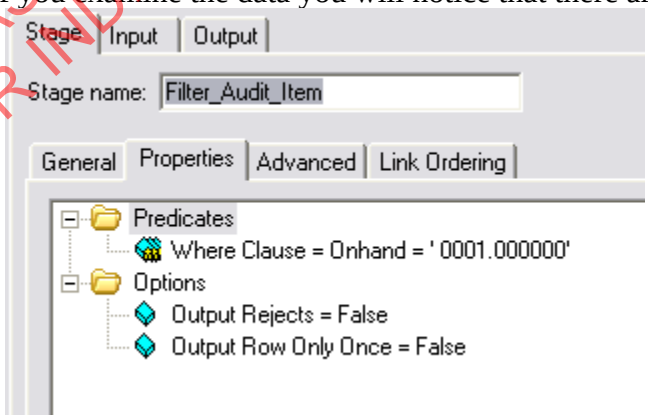
The Copy stage is used direct the records down two output links. The Primary output link is to the target DataSet stage. The link to the Filter Peek stages is used to spot check a small number of the records.

1. Specify that all columns (fields) explicitly move through the stage to both output links.

Task: Edit the Filter and Peek stages

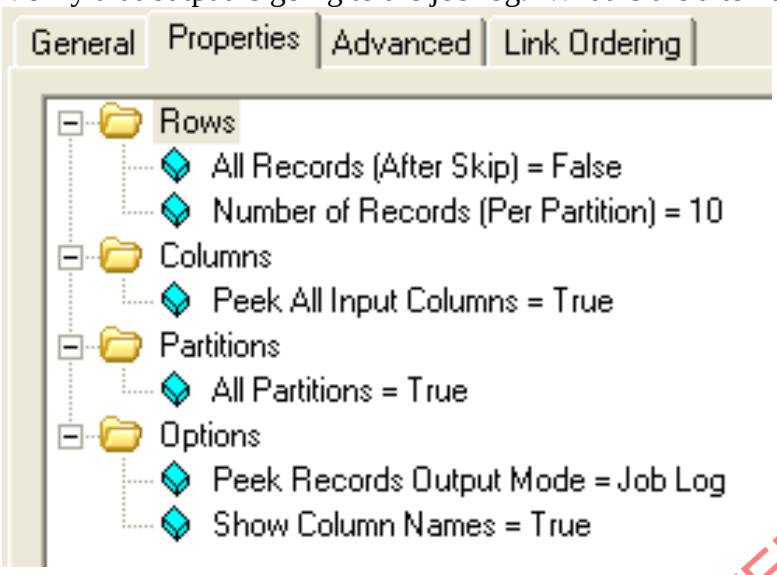
The Filter stage is used to select small subset of records, with Onhand = '0001.000000'.

1. On the Filter Properties tab specify the filtering condition Where clause, namely, audit Onhand = ' 0001.000000'. ***NOTE*:** It is important to have a single space in front of the numerals. If you examine the data you will notice that there are spaces before the numerals.



2. Although nothing needs to done on the Peek stage, open up the Properties tab and examine the default properties.
3. Verify that the all input columns are being peeked at. What is the alternative?

4. Verify that output is going to the job log. What is the alternative?

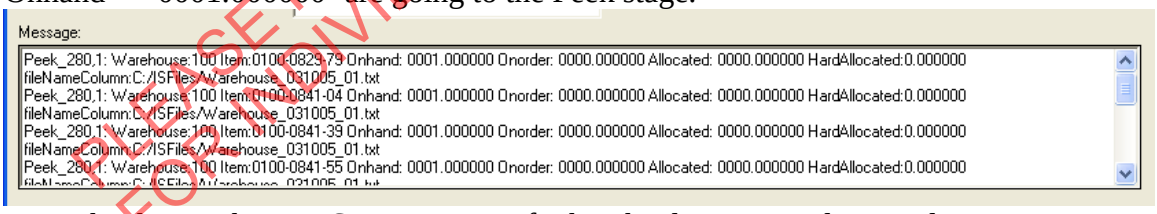


Task: Edit the DataSet stage

1. Specify that the data is to be directed to a dataset file named Sort_Warehouse_Item_DataSet.ds in your ISFiles directory.

Task: Compile and run

1. Compile your job.
2. Run your job.
3. Open up Director and look at the runtime log.
4. In the job log verify from the message written by the Peek stage that only records with Onhand = ‘ 0001.000000’ are going to the Peek stage.



5. View the data in the DataSet stage. Verify that the data is sorted correctly.

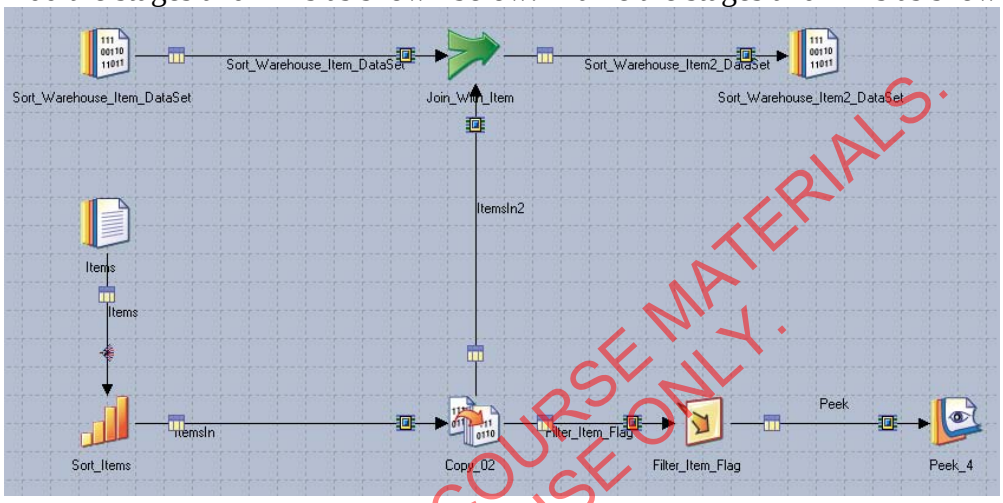
Build Warehouse_02 Job

Assumptions

- You have built and successfully executed the Warehouse_01 job.

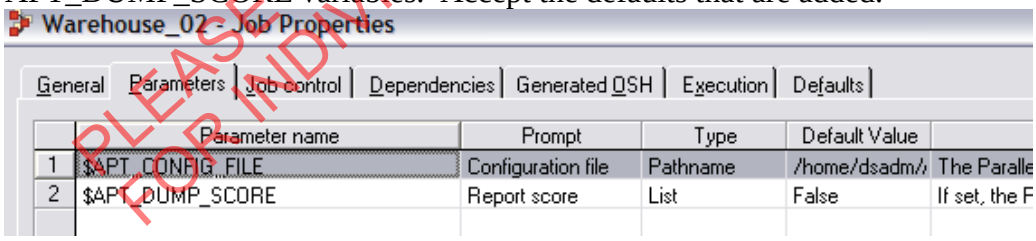
Task: Create the job design

1. Open a new Parallel job and save it under the name Warehouse_02. Store it in your _Training>Jobs folder.
2. Add the stages and links as shown below. Name the stages and links as shown.



Task: Add job parameters

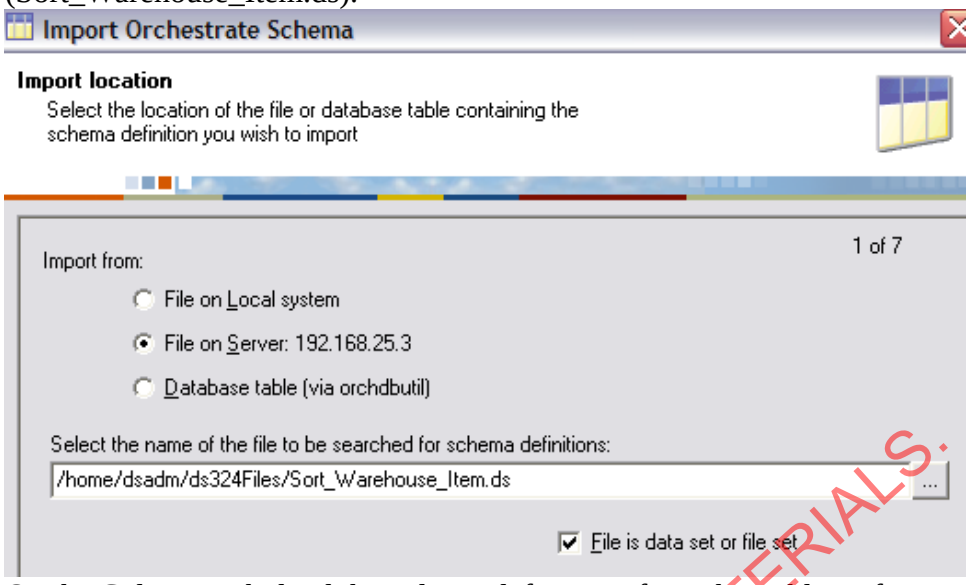
1. Open up the Job Properties window and click on the Parameters tab.
2. Click on the Add Environment Variable button and add the APT_CONFIG_FILE and APT_DUMP_SCORE variables. Accept the defaults that are added.



Task: Edit the source data set

1. Extract data from the dataset target file (Sort_Warehouse_Item.ds) of your Warehouse_01 job.
2. Import the Table Definition for this dataset. To do this, click Import>Orchestrate Schema Definitions, select the "File is data set" check box, and then select the dataset

(Sort_Warehouse_Item.ds).



3. On the Columns tab, load the column definitions from the Table Definition you imported. It will be placed in the Orchestrate>Schema Imports sub-folder.
4. Click View data to verify that your job can read from the dataset.

Task: Edit the source Sequential stage

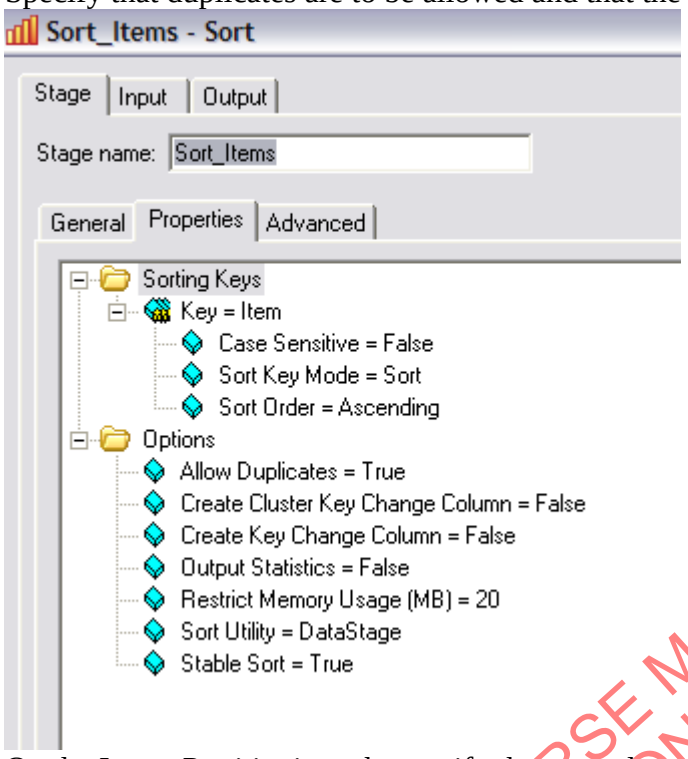
1. On the Properties tab, specify the path to the Items.txt file.
2. On the Format and Columns tabs, load the metadata. If necessary, import a Table Definition for the file to load into the stage.
3. Click View data to verify that your job can read from the file.

Task: Edit the Sort stage

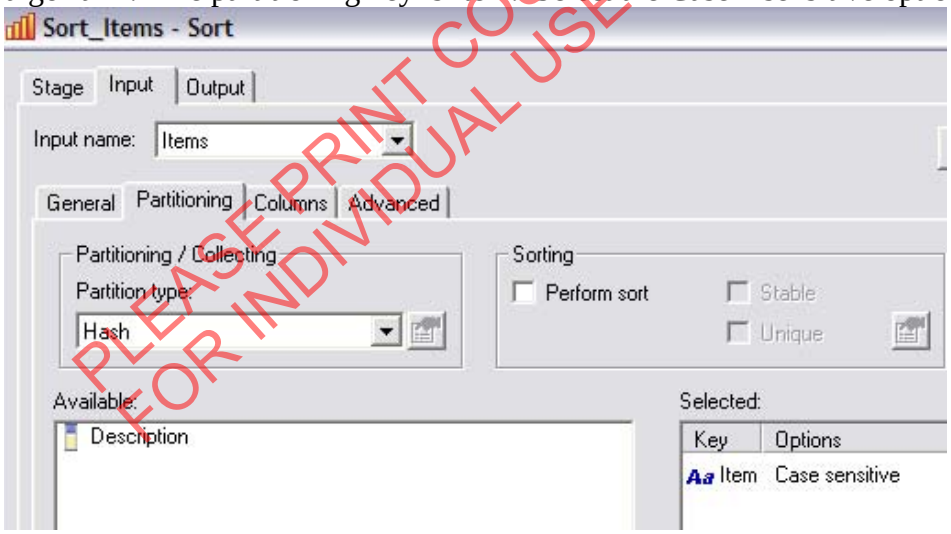
The Sort stage is used to sort the records by the Item field.

1. Open up the Sort stage. On the Properties tab, specify that the data is to be sorted by the Item field, in ascending order. The sort should be case insensitive.

2. Specify that duplicates are to be allowed and that the sort is to be Stable.



3. On the Input, Partitioning tab, specify that records are to be partitioned using a Hash algorithm. The partitioning key is Item. Select the Case insensitive option.



4. Specify that all columns (fields) explicitly move through the stage.

Task: Edit the Copy stage

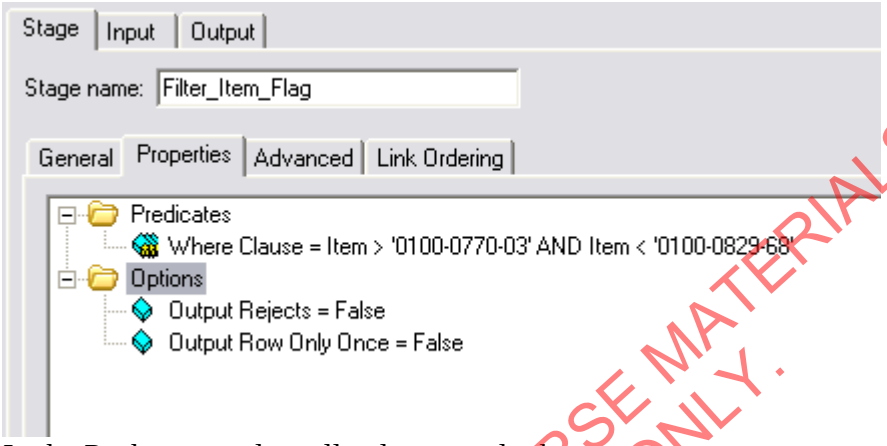
The Copy stage is used direct the records down two output links. The main stream output link is to the Join stage. The link to the Filter Peek stages is used to spot check a small number of the records.

1. Specify that all columns move through the stage to the Join stage.
2. Specify that all columns move through to the Filter Peek stages.

Task: Edit the Filter and Peek stages

The Filter stage is used to select a small subset of records, namely those with Item greater than '0100-0770-03' and less than '0100-0829-68'.

1. On the Properties tab specify the filtering condition Where clause, namely, Item greater than '0100-0770-03' and less than '0100-0829-68'.

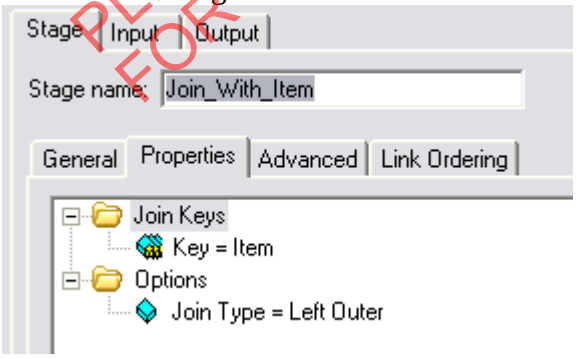


2. In the Peek stage select all columns to look at.

Task: Edit the Join stage

The join stage is used to merge the data from the output of the previous job with the data from the Items.txt file. Both these data streams have been sorted by the Item field.

1. Open up the Join stage.
2. On the Properties page, specify that the join is a Left Outer join and that the Join key is the Item field. Verify on the Link Ordering tab that the link from the data set is the left link. This means that all rows from this link will be sent out the Join stage, even those with no matching Item values.

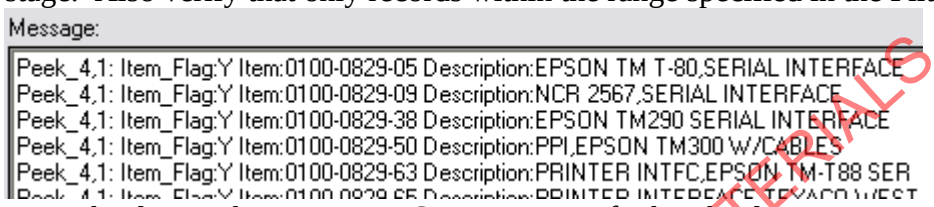


Task: Edit the DataSet stage

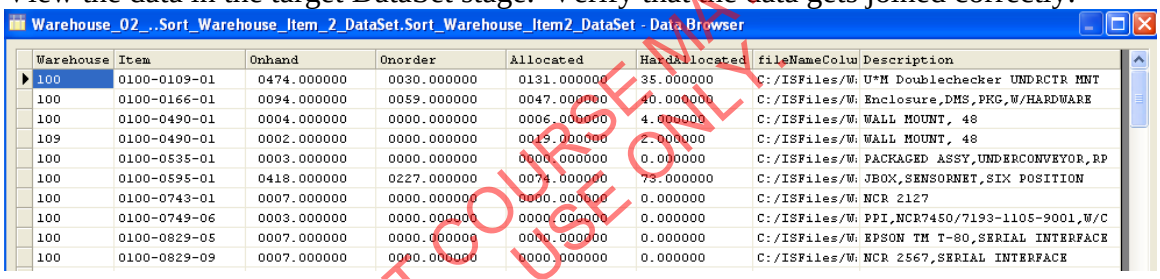
1. Specify that the data is to be directed to a dataset file named Sort_Warehouse_Item_2.ds.

Task: Compile and run and test

1. Compile your job.
2. Run your job.
3. Open up Director and look at the runtime log.
4. In the job log verify from the messages written by the Peek stage that only records with items greater than '0100-0770-03' and less than '0100-0829-68' are going to the Peek stage. Also verify that only records within the range specified in the Filter are selected.



5. View the data in the target DataSet stage. Verify that the data gets joined correctly.



Warehouse	Item	Onhand	Onorder	Allocated	HardAllocated	fileNameColu	Description
100	0100-0109-01	0474.000000	0030.000000	0131.000000	35.000000	C:/ISFiles/W	U*M Doublechecker UNDRCTR MNT
100	0100-0166-01	0094.000000	0059.000000	0047.000000	40.000000	C:/ISFiles/W	Enclosure,DMS,PKG,W/HARDWARE
100	0100-0490-01	0004.000000	0000.000000	0006.000000	4.000000	C:/ISFiles/W	WALL MOUNT, 48
109	0100-0490-01	0002.000000	0000.000000	0019.000000	2.000000	C:/ISFiles/W	WALL MOUNT, 48
100	0100-0535-01	0003.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/W	PACKAGED ASSY,UNDERCONVEYOR,RP
100	0100-0595-01	0418.000000	0227.000000	0074.000000	73.000000	C:/ISFiles/W	JBOX,SENSORNET,SIX POSITION
100	0100-0743-01	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/W	NCR 2127
100	0100-0749-06	0003.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/W	PPI,NCR7450/7193-1105-9001,W/C
100	0100-0829-05	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/W	EPSON TM T-80,SERIAL INTERFACE
100	0100-0829-09	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/W	NCR 2567,SERIAL INTERFACE

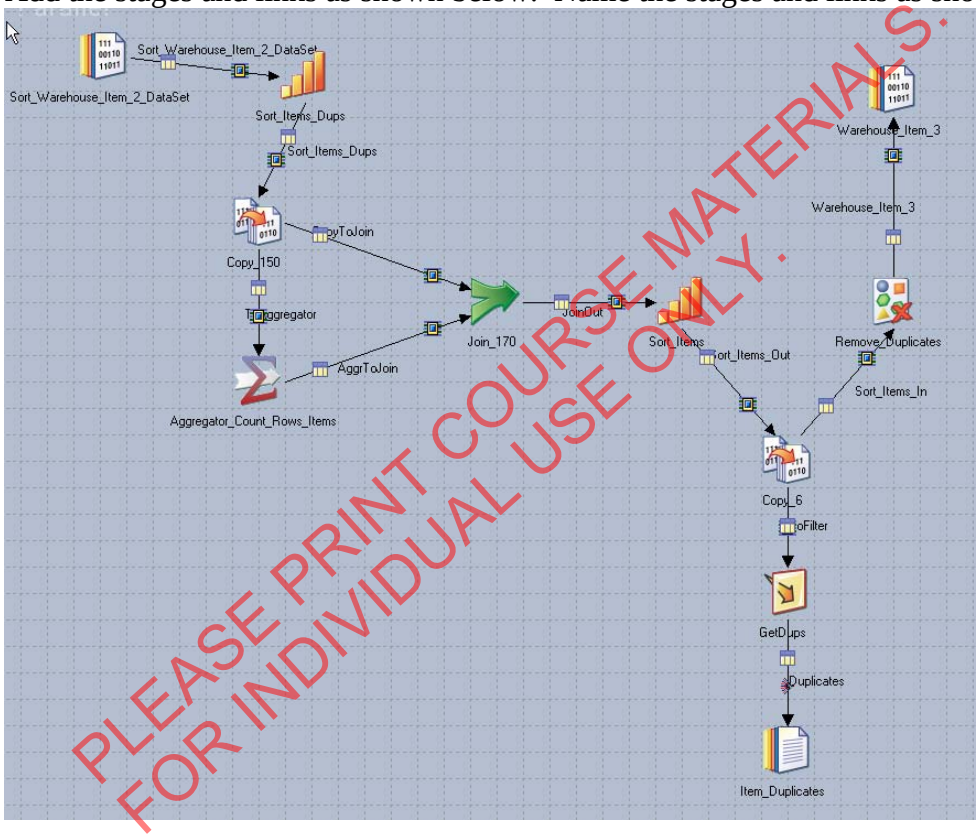
Build Warehouse_03 Job

Assumptions

- You have built and successfully executed the Warehouse_02 job. If you haven't, import the Warehouse_02.dsx file and, if necessary, the Warehouse_01.dsx file from your Lab>Solutions folder on the Student CD.

Task: Create the job design

1. Open a new parallel job and save it under the name Warehouse_03. Store it your _Training>Jobs folder.
2. Add the stages and links as shown below. Name the stages and links as shown.



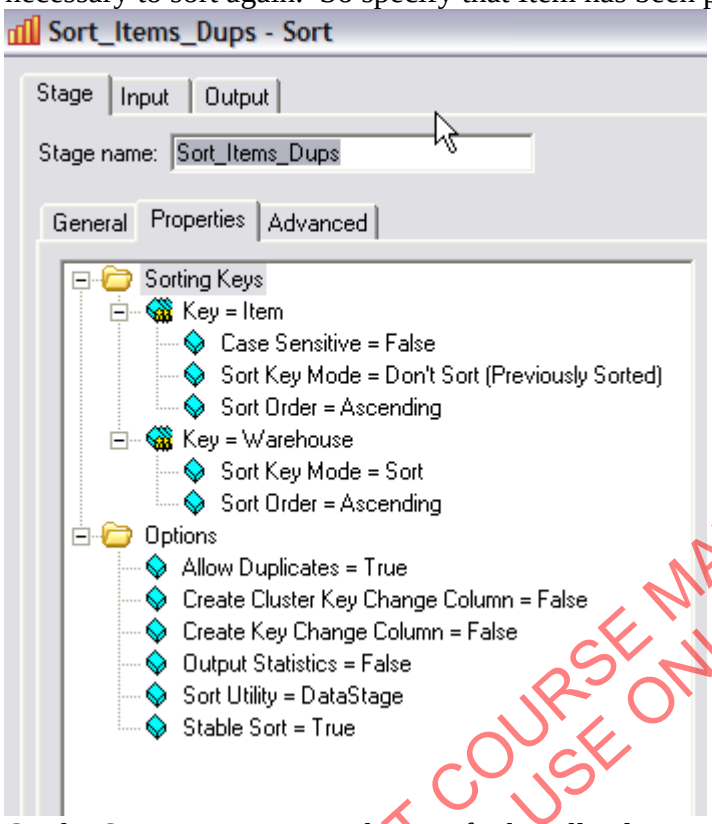
Task: Edit the source DataSet stage

1. Extract data from the DataSet that was the target of the previous job.

Task: Edit the Sort_Items_Dups stage

The Sort stage is used to sort the records in the proper order to build the groups of duplicates. A duplicate is a record with the same Item and Warehouse. The number of duplicates will be calculated in the Aggregator stage and added to each record.

1. Open up the Sort stage. On the Properties tab, specify that the records are to be sorted by Item and Warehouse keys. Since the records were previously sorted by Item, it is not necessary to sort again. So specify that Item has been previously sorted.



2. On the Outputs>Mapping tab, specify that all columns explicitly move through the stage.

Task: Edit the Copy stage

The Copy stage is used direct the records down two output links. One output link is directly to the Join stage. The second output is to the Aggregator stage.

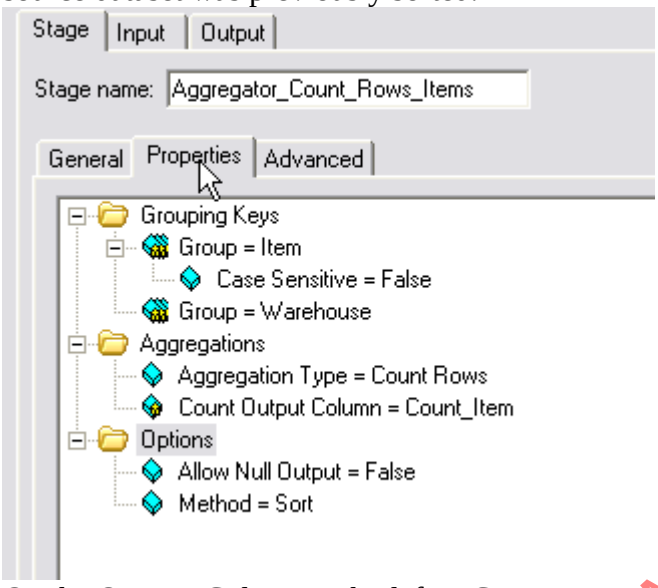
1. Specify that all columns (fields) explicitly move through the stage to the output link going to the Join stage.
2. Specify that only the Item and Warehouse columns are to go to the Aggregator stage.

Task: Edit the Aggregator stage

The Aggregator stage is used to count the number of records in each Item Warehouse group.

1. On the Input>Columns tab, verify that only the Item and Warehouse columns are input to the Aggregator stage.
2. On the Properties tab, specify that records are to be grouped by Item and Warehouse. The Item group is not case sensitive.
3. Specify that the type of aggregation is to count the rows.
4. Specify that the aggregation amount is to go in the Count_Item output column.

5. Under the Options folder, select Sort as the aggregation method. This assumes that the records are sorted by grouping keys, which is the case, since the data going into the source dataset was previously sorted.



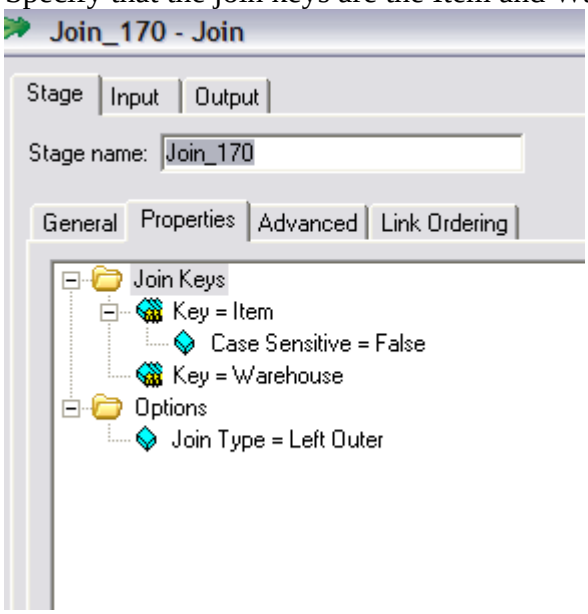
6. On the Output>Columns tab, define Count_Item as an integer.
7. On the Output>Mapping tab, move all three columns across the stage..

Task: Edit the Join stage

The Join stage is used to merge the records from the Sort_Warehouse_Item_2_DataSet stage, coming out of the Copy stage, with the records coming out of the Aggregator stage. This is done to add the Count_Item field to each record.

1. Open up the Join stage.
2. On the Properties tab, specify that the join is a left outer join. That is, the driver (outer) link is the records from the Sort_Warehouse_Item_2_DataSet stage. Note: Check the link ordering to verify that CopyToJoin is the left link. If it isn't change the order.

3. Specify that the join keys are the Item and Warehouse fields.



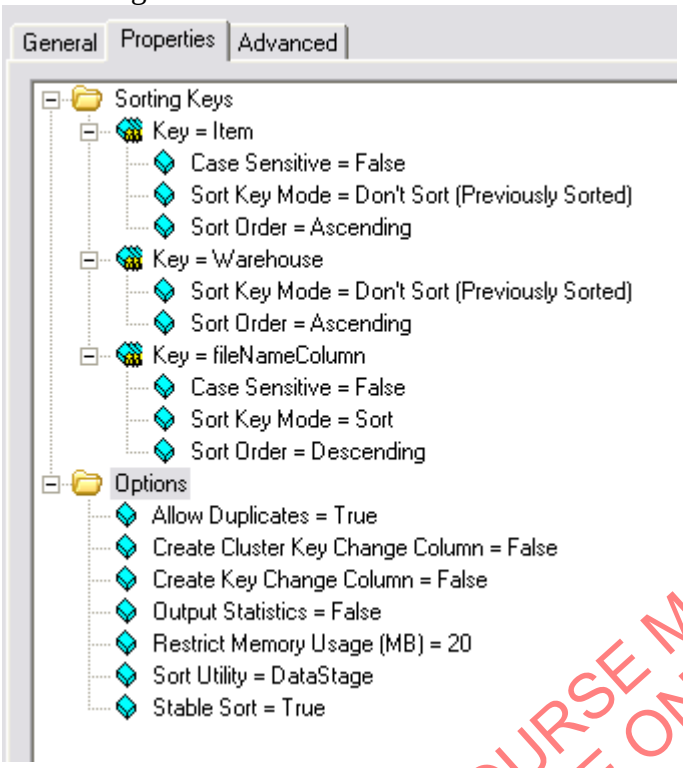
4. On the Outputs>Mapping tab, be sure that the Count_Item field is added to the output from the Join stage.

Task: Edit the Sort_Items stage

This Sort stage is used to sort the records in the proper order before duplicates in each group are removed. A duplicate is a record with the same Item number and Warehouse. The proper order is to have the duplicate from the latest file at the top. Recall that the name of the file from which the record was read is in the fileNameColumn.

1. Open up the Sort stage. On the Properties tab, specify that the Item and Warehouse keys have been previously sorted in ascending order and don't need to be resorted.

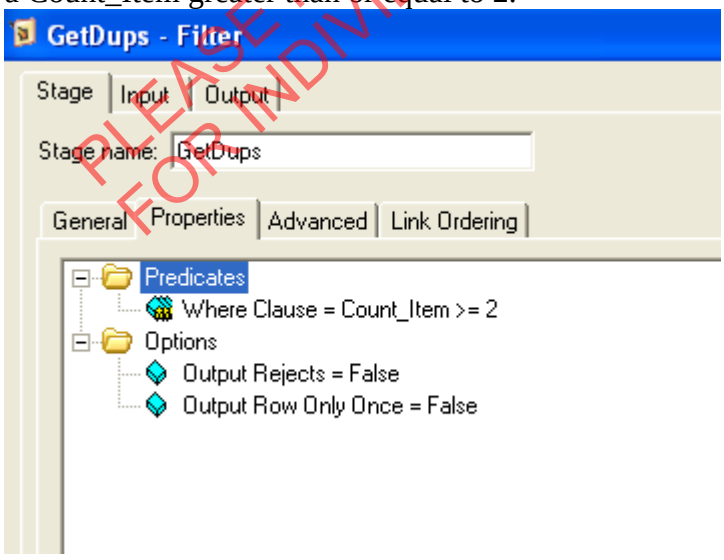
2. Specify that within each group the rows are to be sorted by the fileNameColumn in descending order.



3. On the Outputs>Mapping tab, specify that all columns explicitly move through the stage.

Task: Edit the Copy, Filter, and Duplicates Sequential stages

1. In the Copy stage after the Sort, direct all columns down both output links.
2. In the Filter stage, select the rows that are members of duplicate groups, that is, rows with a Count_Item greater than or equal to 2.

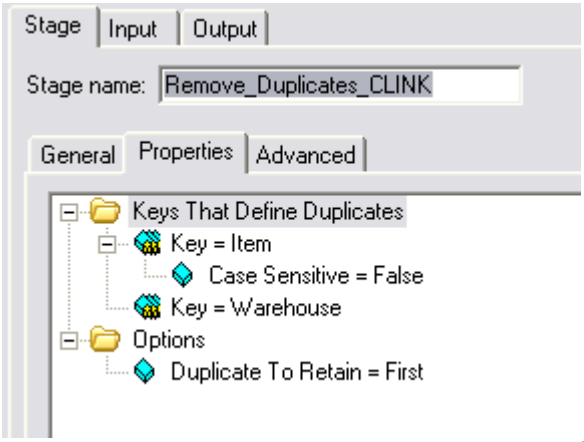


3. Send all duplicates to a sequential file named Item_Duplicates.txt.

Task: Edit the RemoveDuplicates stage

The RemoveDuplicates stage is used to remove duplicate Item Warehouse records. The first is to be retained.

1. Open the RemoveDuplicates stage and specify that duplicate Item Warehouse records are to be removed. The first is to be retained.



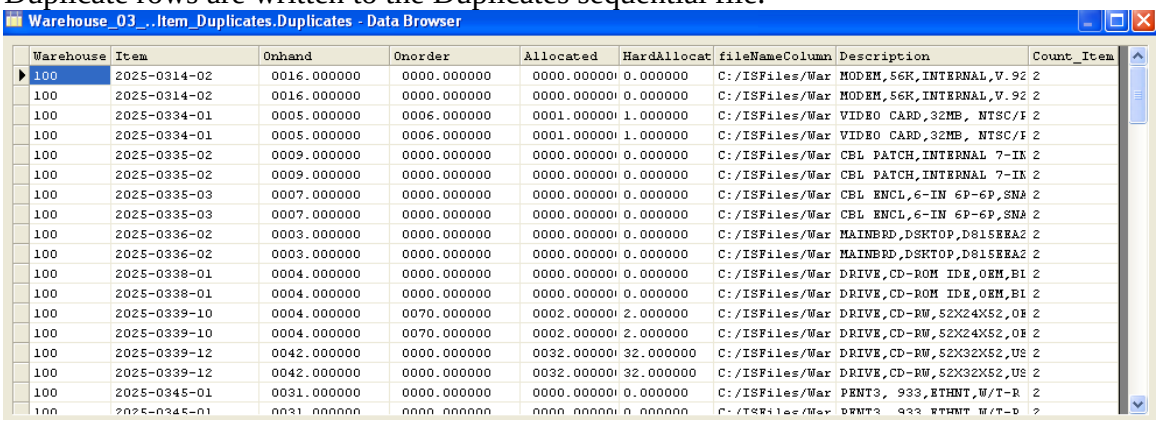
Task: Edit the DataSet stage

1. Specify that the data is to be directed to a dataset file named Sort_Warehouse_Item_3.ds.

Task: Compile and run

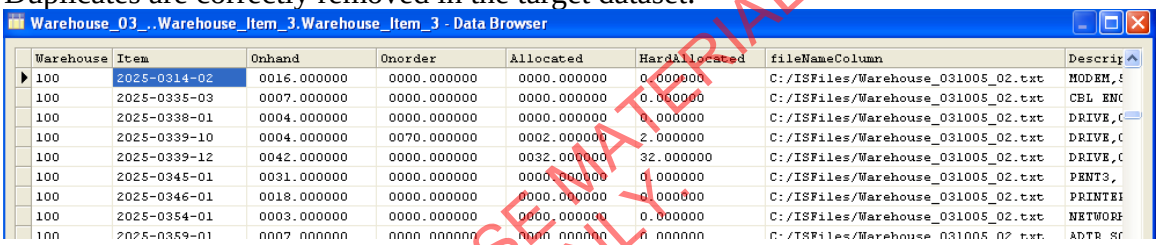
1. Compile and run your job.
2. Open up Director and look at the runtime log. Examine any warnings or errors. You may see a couple of warning messages related to the Aggregator stage. The Aggregator calculates its result in an internal, nullable field and maps this to the specified output column. If the output column is not nullable and has a different type than the internal column's type, you will see warnings. These can be ignored in most cases.
3. Test and verify that your job is performing correctly.

- Duplicate rows are written to the Duplicates sequential file.


Warehouse_03_..Item_Duplicates.Duplicates - Data Browser

Warehouse	Item	Onhand	Onorder	Allocated	HardAllocat	fileNameColumn	Description	Count_Item
100	2025-0314-02	0016.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	MODEM,56K,INTERNAL,V.92	2
100	2025-0314-02	0016.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	MODEM,56K,INTERNAL,V.92	2
100	2025-0334-01	0005.000000	0006.000000	0001.000000	1.000000	C:/ISFiles/War	VIDEO CARD,32MB, NTSC/P	2
100	2025-0334-01	0005.000000	0006.000000	0001.000000	1.000000	C:/ISFiles/War	VIDEO CARD,32MB, NTSC/P	2
100	2025-0335-02	0009.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	CBL PATCH,INTERNAL 7-IN	2
100	2025-0335-02	0009.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	CBL PATCH,INTERNAL 7-IN	2
100	2025-0335-03	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	CBL ENCL,6-IN 6P-6P,SNA	2
100	2025-0335-03	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	CBL ENCL,6-IN 6P-6P,SNA	2
100	2025-0336-02	0003.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	MAINBRD,DSKTOP,D815EEA2	2
100	2025-0336-02	0003.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	MAINBRD,DSKTOP,D815EEA2	2
100	2025-0338-01	0004.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	DRIVE,CD-ROM IDE,OEM,BI	2
100	2025-0338-01	0004.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	DRIVE,CD-ROM IDE,OEM,BI	2
100	2025-0339-10	0004.000000	0070.000000	0002.000000	2.000000	C:/ISFiles/War	DRIVE,CD-RW,52X24X52,OE	2
100	2025-0339-10	0004.000000	0070.000000	0002.000000	2.000000	C:/ISFiles/War	DRIVE,CD-RW,52X24X52,OE	2
100	2025-0339-12	0042.000000	0000.000000	0032.000000	32.000000	C:/ISFiles/War	DRIVE,CD-RW,52X32X52,US	2
100	2025-0339-12	0042.000000	0000.000000	0032.000000	32.000000	C:/ISFiles/War	DRIVE,CD-RW,52X32X52,US	2
100	2025-0345-01	0031.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	PENT3, 933,ETHNT,W/T-R	2
100	2025-0345-01	0031.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/War	PENT3, 933,ETHNT,W/T-R	2

- The Item_Count has been added.
- Duplicates are correctly removed in the target dataset.


Warehouse_03_..Warehouse_Item_3.Warehouse_Item_3 - Data Browser

Warehouse	Item	Onhand	Onorder	Allocated	HardAllocated	fileNameColumn	Descrip
100	2025-0314-02	0016.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	MODEM,5
100	2025-0335-03	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	CBL ENC
100	2025-0338-01	0004.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	DRIVE,C
100	2025-0339-10	0004.000000	0070.000000	0002.000000	2.000000	C:/ISFiles/Warehouse_031005_02.txt	DRIVE,C
100	2025-0339-12	0042.000000	0000.000000	0032.000000	32.000000	C:/ISFiles/Warehouse_031005_02.txt	DRIVE,C
100	2025-0345-01	0031.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	PENT3,
100	2025-0346-01	0018.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	PRINTER
100	2025-0354-01	0003.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	NETWORK
100	2025-0359-01	0007.000000	0000.000000	0000.000000	0.000000	C:/ISFiles/Warehouse_031005_02.txt	ADTR,80

Build Warehouse_04 Job

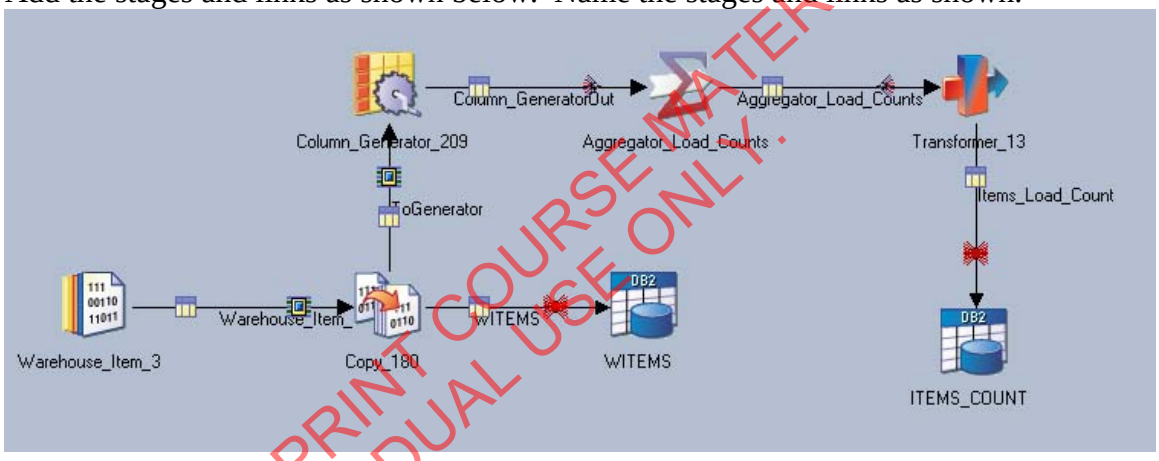
The de-duplicated output of the previous job must be loaded into the appropriate table. A trigger table will be loaded with a field named Item_Load_Count that will contain a count of the number of records that should have been written to the target table.

Assumptions

- You have built and successfully executed the Warehouse_03 job. If you haven't, import the Warehouse_03.dsx file and, if necessary, the Warehouse_02.dsx file and the Warehouse_01.dsx file from your Lab>Solutions folder on the Student CD.

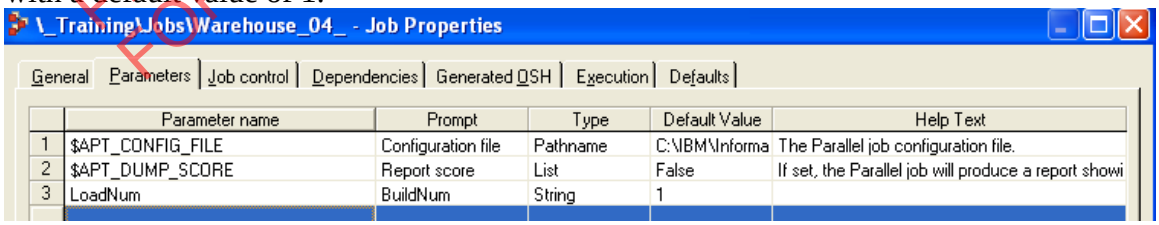
Task: Create the job design

1. Open a new Parallel job and save it under the name Warehouse_04 in your _Training>Jobs folder.
2. Add the stages and links as shown below. Name the stages and links as shown.



Task: Add job parameters

1. Open up the Job Properties window and click on the Parameters tab.
2. Click on the Add Environment Variable button and add a parameter named LoadNum with a default value of 1.



Task: Edit the source DataSet stage

1. Extract data from the DataSet (Sort_Warehouse_Item_3.ds) that was the target of the previous job.

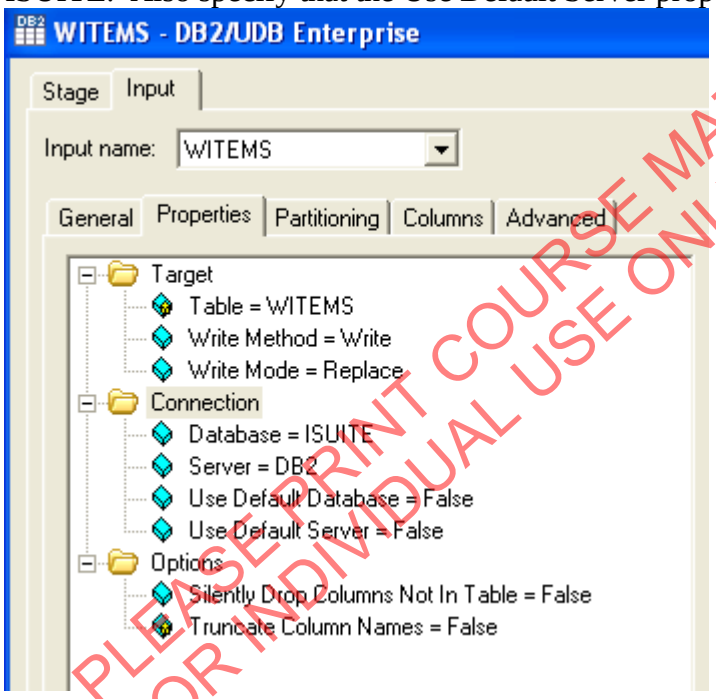
Task: Edit the Copy stage

The Copy stage is used direct the records down two output links. One output link is to the DB2 stage. The second output is to the ColumnGenerator stage.

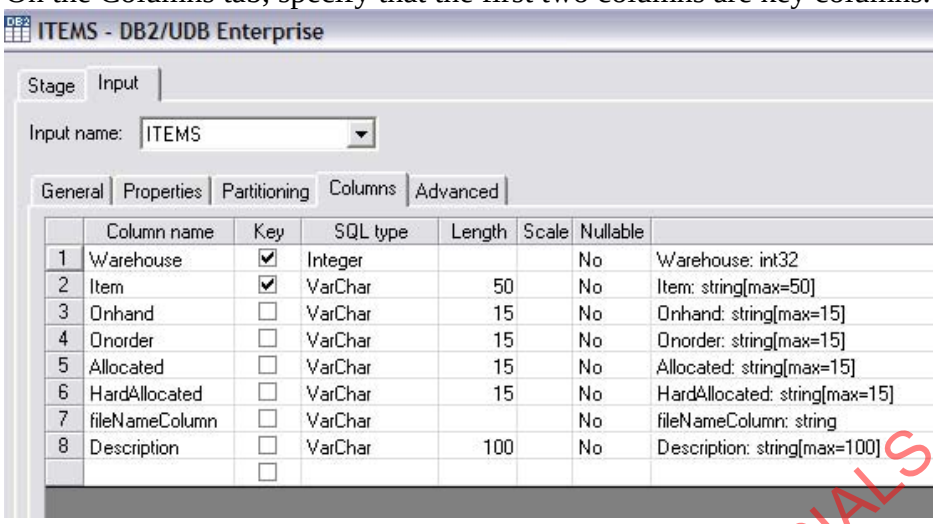
1. On the Output>Mapping tab, select the link going to the DB2 stage.
2. Pass all columns except Count_Item through.
3. Select the link going to the ColumnGenerator stage. Pass just the Item column through.

Task: Edit the ITEMS DB2 stage

1. On the Properties tab, specify the table to load: WITEMS.
2. Specify options to truncate and write to the table. The Write Method should be "Write" and the Write Mode should be "Replace".
3. In the Connection folder specify that "Use Default Database" is False. The database is ISUITE. Also specify that the Use Default Server property is False. The server is DB2.



4. On the Columns tab, specify that the first two columns are key columns.

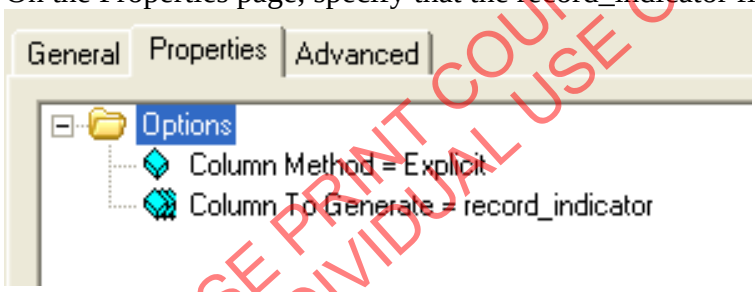


	Column name	Key	SQL type	Length	Scale	Nullable	
1	Warehouse	☑	Integer			No	Warehouse: int32
2	Item	☑	VarChar	50		No	Item: string[max=50]
3	Onhand	☐	VarChar	15		No	Onhand: string[max=15]
4	Onorder	☐	VarChar	15		No	Onorder: string[max=15]
5	Allocated	☐	VarChar	15		No	Allocated: string[max=15]
6	HardAllocated	☐	VarChar	15		No	HardAllocated: string[max=15]
7	fileNameColumn	☐	VarChar			No	fileNameColumn: string
8	Description	☐	VarChar	100		No	Description: string[max=100]

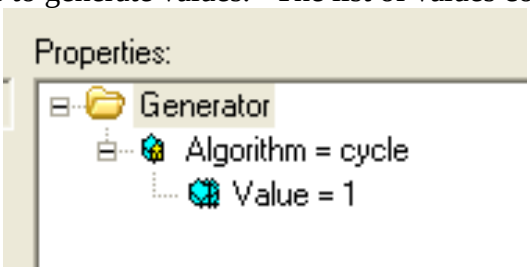
Task: Edit the ColumnGenerator stage

This stage is used to generate a new column named record_indicator.

1. Open up the ColumnGenerator stage.
2. On the Properties page, specify that the record_indicator field is to be generated.



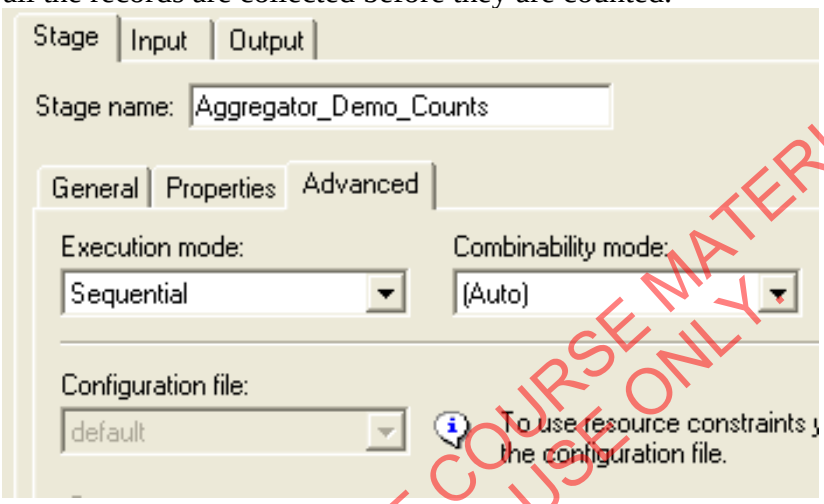
3. On the Output>Columns tab, click the right mouse button over the record_indicator field and click Edit row. This opens the Edit Column Meta Data window.
4. Select cycle as the algorithm to use to generate values. The list of values contains just one value, 1.



Task: Edit the Aggregator stage

The Aggregator stage is used to count the number of rows and output the result to the Item_Load_Count output column. Because all the rows need to be counted, not just the rows in each partition, it is necessary to execute the Aggregator stage in Sequential mode.

1. Open up the Aggregator stage.
2. On the Properties tab, group the records by record_indicator.
3. Specify the aggregation type as Count Rows and select the Item_Load_Count field as the Count Output Column.
4. On the Stage Advanced tab, select Sequential as the Execution mode. This insures that all the records are collected before they are counted.

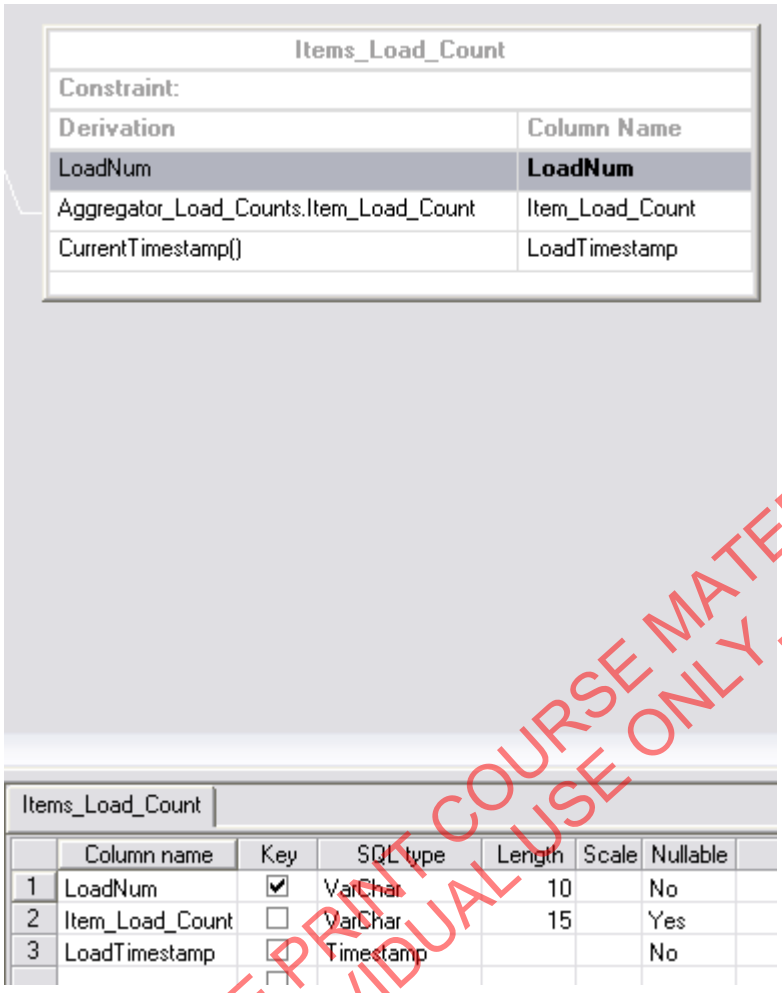


5. Only one column goes out of the Aggregator stage, namely, Item_Load_Count.

Task: Edit the Transformer stage

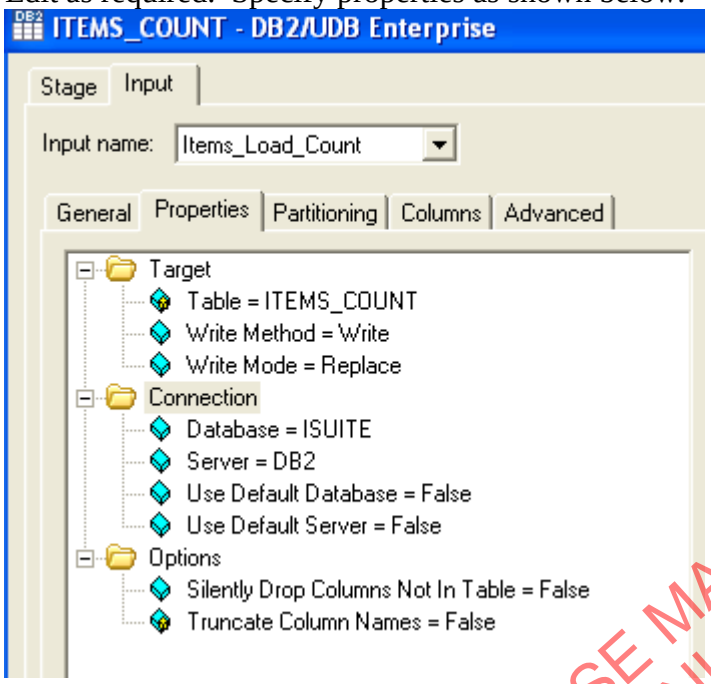
1. Open up the Transformer stage.
2. Create a new column named LoadNum. Make it a key column. It's derivation is a mapping from the LoadNum job parameter.
3. Map the Item_Load_Count column through the Transformer.

4. Create a new column named LoadTimestamp. Store the timestamp when the table is loaded.



Task: Edit the ITEM_COUNTS DB2 stage

1. Edit as required. Specify properties as shown below.



Task: Compile and run

1. Compile and run your job.
2. Open up Director and look at the runtime log.
3. Test and verify that your job is performing correctly.

Build Warehouse_04 Job (Teradata)

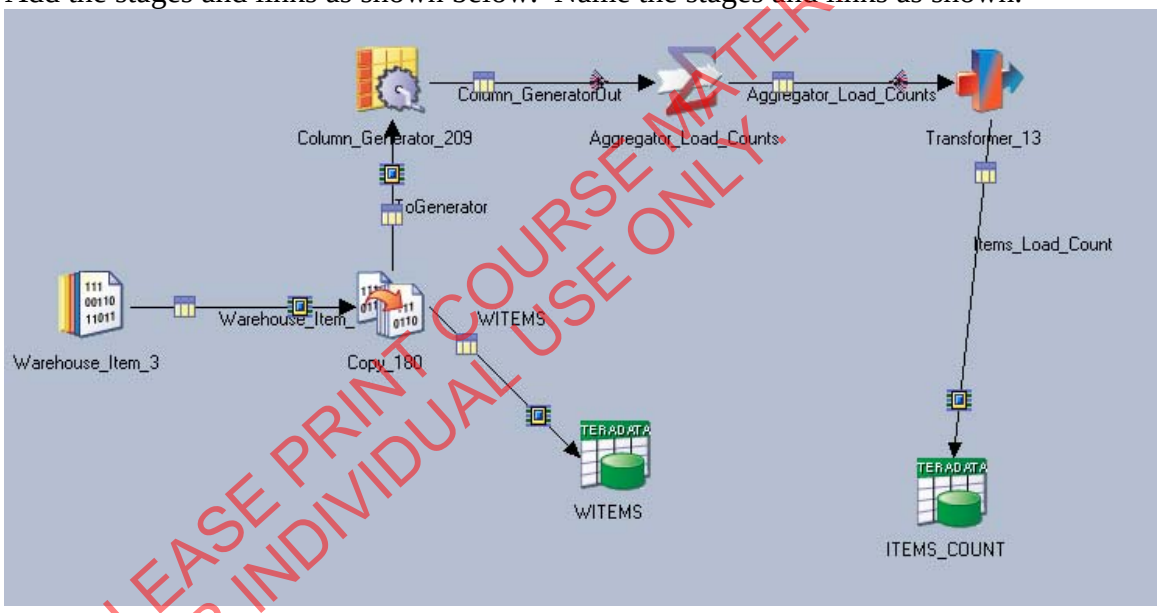
The de-duplicated output of the previous job must be loaded into the appropriate table. A trigger table will be loaded with a field named Item_Load_Count that will contain a count of the number of records that should have been written to the target table.

Assumptions

- You have built and successfully executed the Warehouse_03 job. If you haven't, import the Warehouse_03.dsx file and, if necessary, the Warehouse_02.dsx file and the Warehouse_01.dsx file from your Lab>Solutions folder on the Student CD.

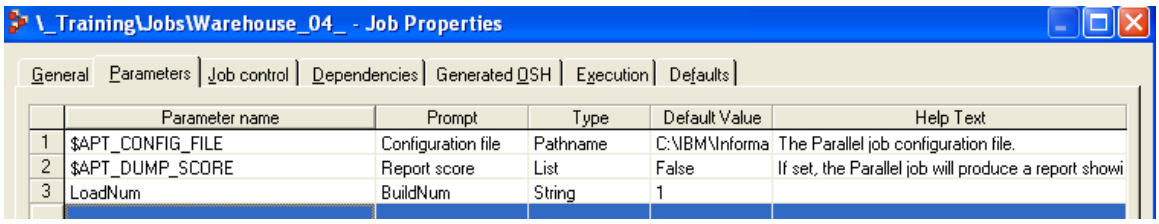
Task: Create the job design

1. Open a new Parallel job and save it under the name Warehouse_04_Teradata in your _Training>Jobs folder.
2. Add the stages and links as shown below. Name the stages and links as shown.



Task: Add job parameters

1. Open up the Job Properties window and click on the Parameters tab.
2. Click on the Add Environment Variable button and add a parameter named LoadNum with a default value of 1.



Task: Edit the source DataSet stage

1. Extract data from the DataSet (Sort_Warehouse_Item_3.ds) that was the target of the previous job.

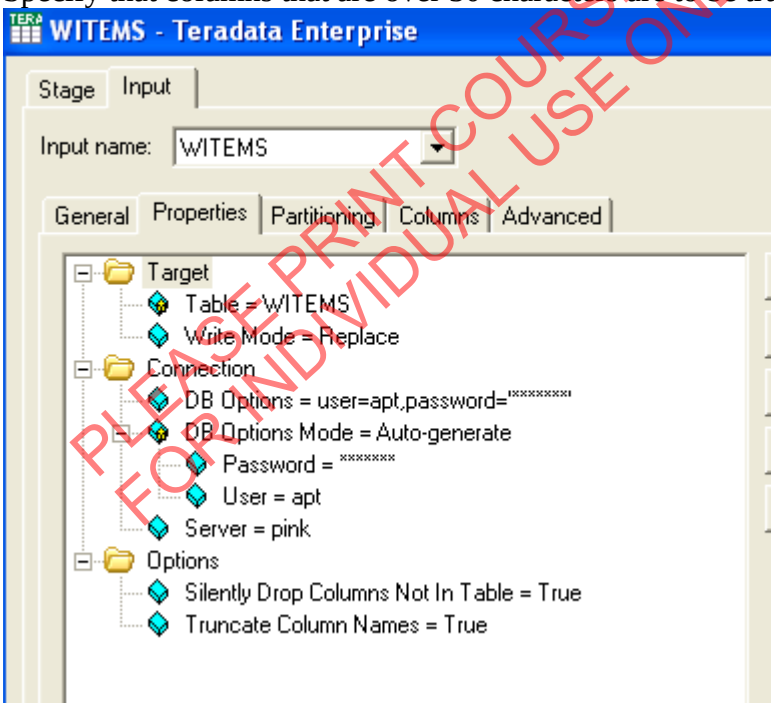
Task: Edit the Copy stage

The Copy stage is used direct the records down two output links. One output link is to the Teradata stage. The second output is to the ColumnGenerator stage.

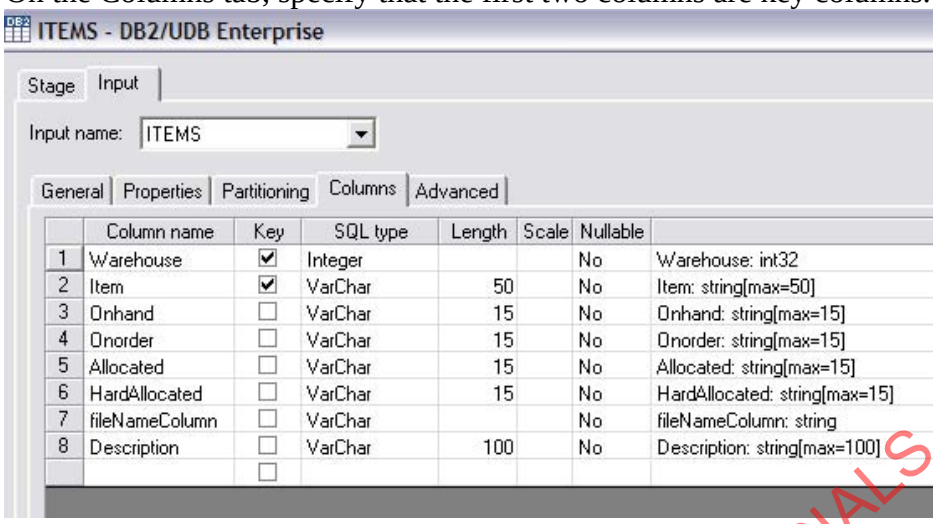
1. On the Output>Mapping tab, select the link going to the Teradata stage.
2. Pass all columns except Count_Item through.
3. Select the link going to the ColumnGenerator stage. Pass just the Item column through.

Task: Edit the WITEMS Teradata stage

1. On the Properties tab, specify the table to load: WITEMS.
2. Specify options to truncate and write to the table. The Write Mode should be "Replace".
3. In the Connection folder specify the DB Options Mode as Auto-generate. Then specify user name and password. (This information will be provided by your instructor.)
4. Specify the name of the Server. (This information will be provided by your instructor.)
5. Specify that columns in the record that are not in the table are to be silently dropped. Specify that columns that are over 30 characters are to be truncated.



6. On the Columns tab, specify that the first two columns are key columns.

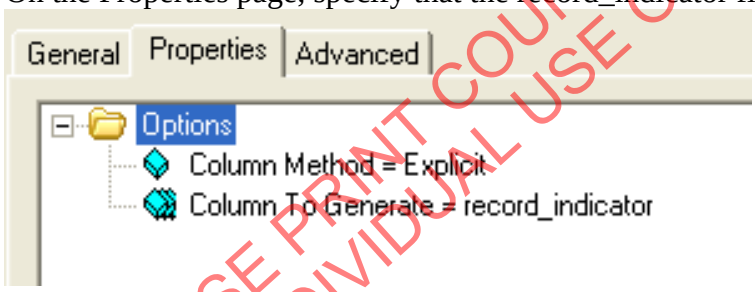


	Column name	Key	SQL type	Length	Scale	Nullable	
1	Warehouse	☑	Integer			No	Warehouse: int32
2	Item	☑	VarChar	50		No	Item: string[max=50]
3	Onhand	☐	VarChar	15		No	Onhand: string[max=15]
4	Onorder	☐	VarChar	15		No	Onorder: string[max=15]
5	Allocated	☐	VarChar	15		No	Allocated: string[max=15]
6	HardAllocated	☐	VarChar	15		No	HardAllocated: string[max=15]
7	fileNameColumn	☐	VarChar			No	fileNameColumn: string
8	Description	☐	VarChar	100		No	Description: string[max=100]

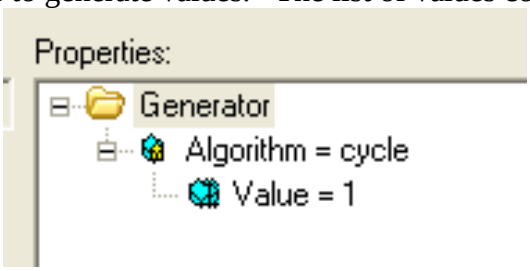
Task: Edit the ColumnGenerator stage

This stage is used to generate a new column named record_indicator.

1. Open up the ColumnGenerator stage.
2. On the Properties page, specify that the record_indicator field is to be generated.



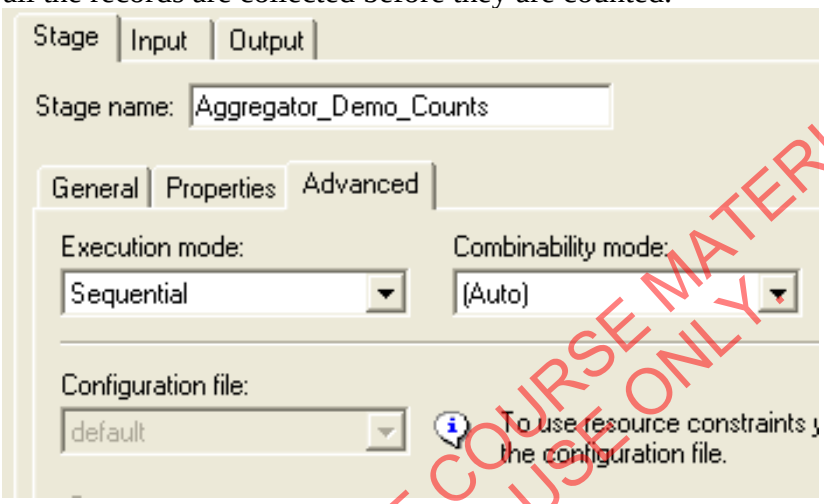
3. On the Output>Columns tab, click the right mouse button over the record_indicator field and click Edit row. This opens the Edit Column Meta Data window.
4. Select cycle as the algorithm to use to generate values. The list of values contains just one value, 1.



Task: Edit the Aggregator stage

The Aggregator stage is used to count the number of rows and output the result to the Item_Load_Count output column. Because all the rows need to be counted, not just the rows in each partition, it is necessary to execute the Aggregator stage in Sequential mode.

6. Open up the Aggregator stage.
7. On the Properties tab, group the records by record_indicator.
8. Specify the aggregation type as Count Rows and select the Item_Load_Count field as the Count Output Column.
9. On the Stage Advanced tab, select Sequential as the Execution mode. This insures that all the records are collected before they are counted.

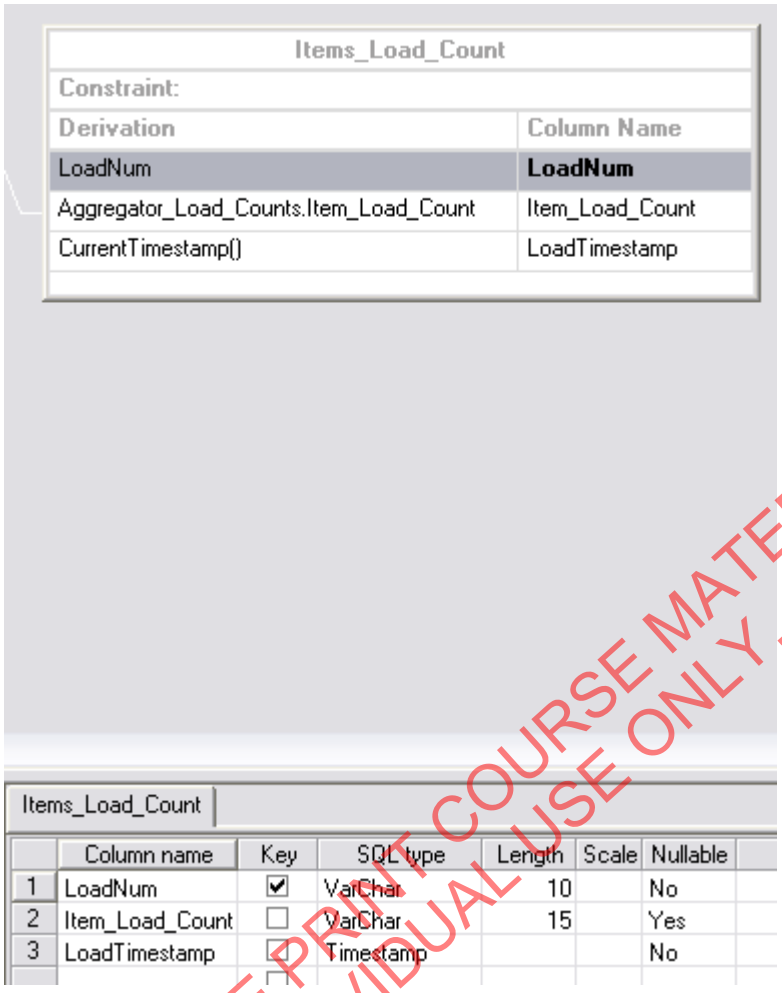


10. Only one column goes out of the Aggregator stage, namely, Item_Load_Count.

Task: Edit the Transformer stage

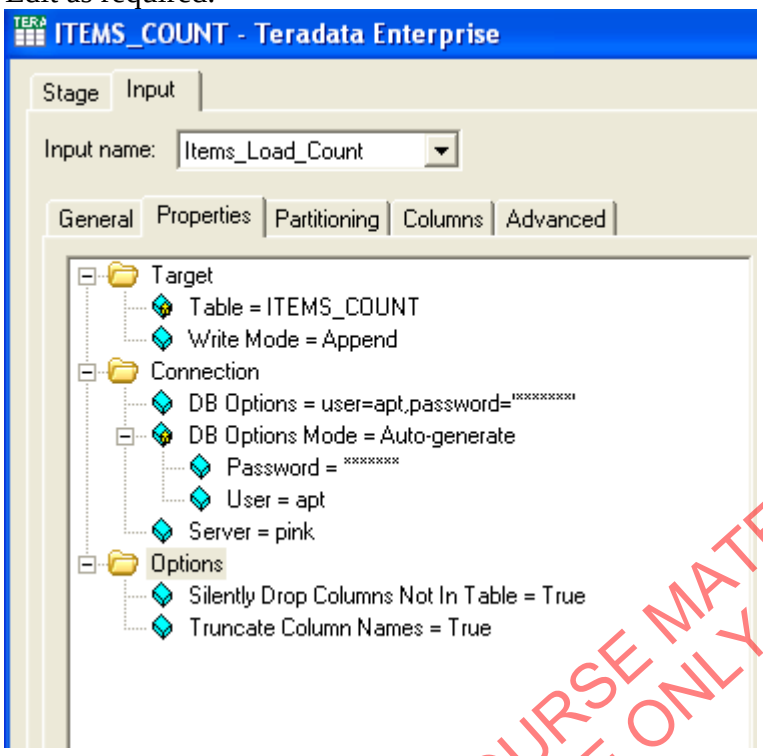
5. Open up the Transformer stage.
6. Create a new column named LoadNum. Make it a key column. It's derivation is a mapping from the LoadNum job parameter.
7. Map the Item_Load_Count column through the Transformer.

8. Create a new column named LoadTimestamp. Store the timestamp when the table is loaded.



Task: Edit the ITEM_COUNS Teradata stage

1. Edit as required.



Task: Compile and run

1. Compile and run your job.
2. Open up Director and look at the runtime log.
3. Test and verify that your job is performing correctly.

Build Warehouse_04 Job (Oracle)

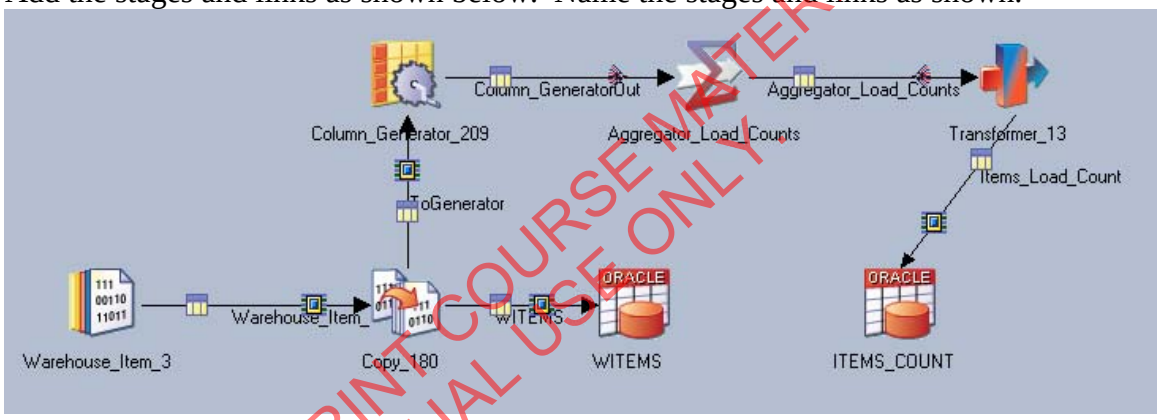
The de-duplicated output of the previous job must be loaded into the appropriate table. A trigger table will be loaded with a field named Item_Load_Count that will contain a count of the number of records that should have been written to the target table.

Assumptions

- You have built and successfully executed the Warehouse_03 job. If you haven't, import the Warehouse_03.dsx file and, if necessary, the Warehouse_02.dsx file and the Warehouse_01.dsx file from your Lab>Solutions folder on the Student CD.

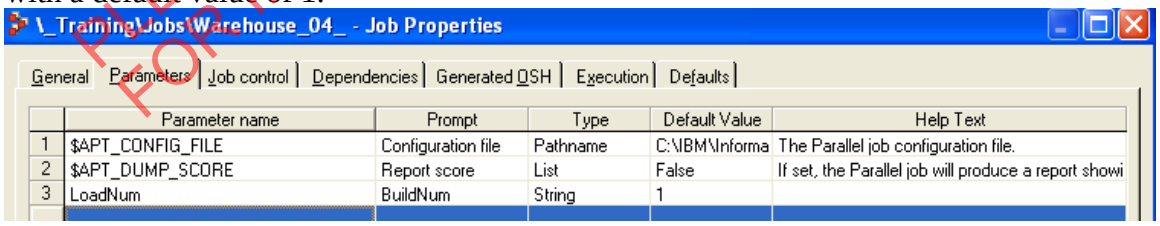
Task: Create the job design

1. Open a new Parallel job and save it under the name Warehouse_04_Oracle. Store it in a folder named ds434.
2. Add the stages and links as shown below. Name the stages and links as shown.



Task: Add job parameters

1. Open up the Job Properties window and click on the Parameters tab.
2. Click on the Add Environment Variable button and add a parameter named LoadNum with a default value of 1.



Task: Edit the source DataSet stage

1. Extract data from the DataSet (Sort_Warehouse_Item_3.ds) that was the target of the previous job.

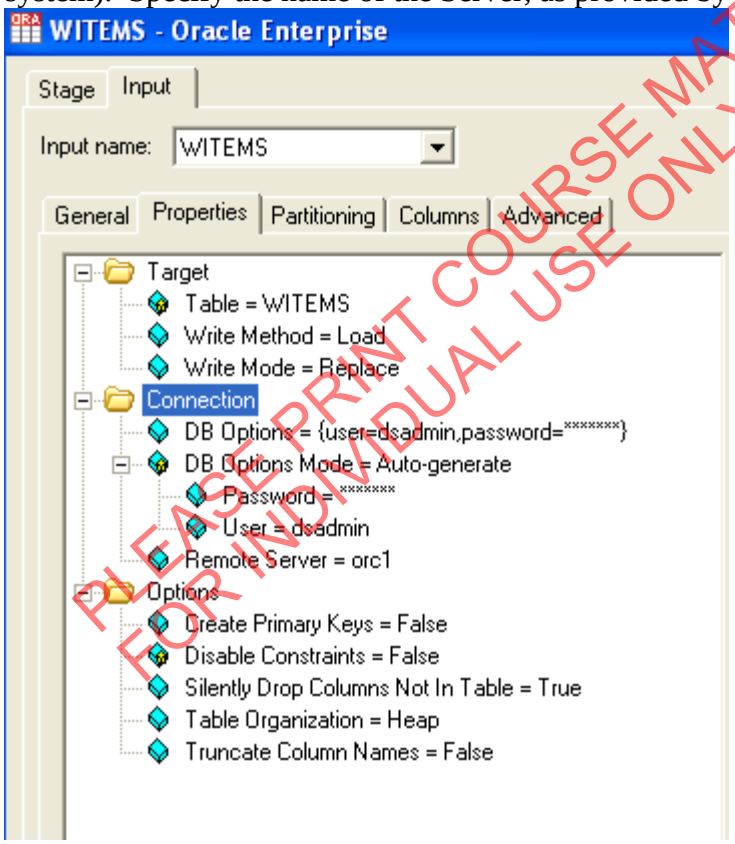
Task: Edit the Copy stage

The Copy stage is used direct the records down two output links. One output link is to the Oracle stage. The second output is to the ColumnGenerator stage.

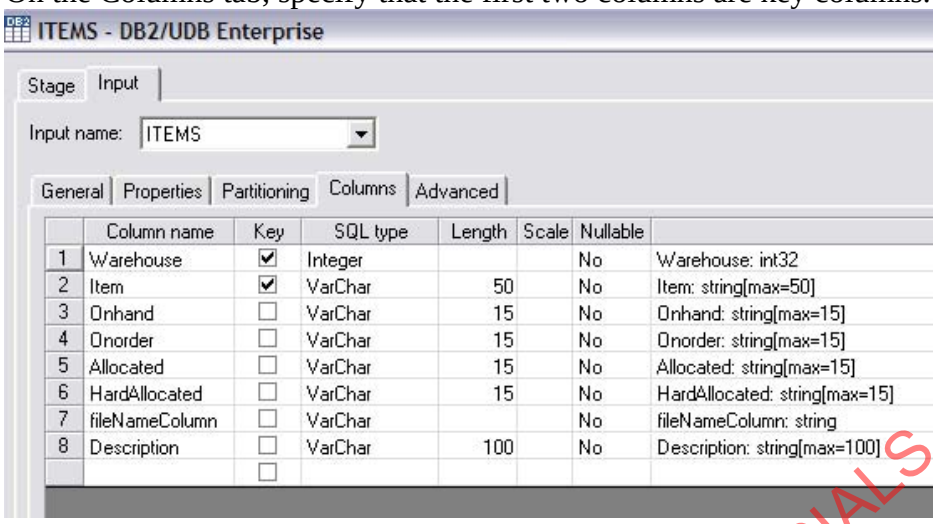
1. On the Output>Mapping tab, select the link going to the Oracle stage.
2. Pass all columns except Count_Item through.
3. Select the link going to the ColumnGenerator stage. Pass just the Item column through.

Task: Edit the ITEMS Oracle stage

1. On the Properties tab, specify the table to load: WITEMS.
2. Specify options to truncate and write to the table. The Write Method should be "Load" and the Write Mode should be "Replace".
3. In the Connection folder specify that DB Options Mode is Auto-generate. Specify the user name and password. (Your instructor will provide this.)
4. Select the Remote Server Connection option (assuming the Oracle Server is on a different system). Specify the name of the Server, as provided by your instructor.



5. On the Columns tab, specify that the first two columns are key columns.

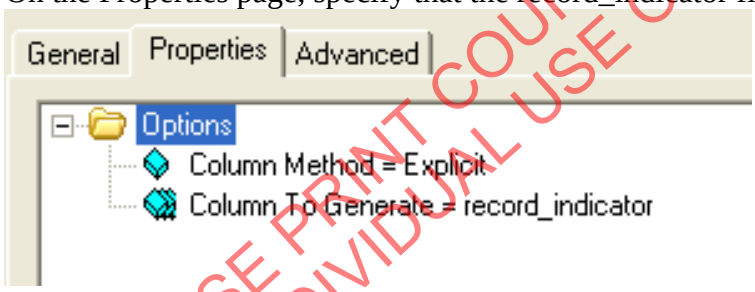


	Column name	Key	SQL type	Length	Scale	Nullable	
1	Warehouse	☑	Integer			No	Warehouse: int32
2	Item	☑	VarChar	50		No	Item: string[max=50]
3	Onhand	☐	VarChar	15		No	Onhand: string[max=15]
4	Onorder	☐	VarChar	15		No	Onorder: string[max=15]
5	Allocated	☐	VarChar	15		No	Allocated: string[max=15]
6	HardAllocated	☐	VarChar	15		No	HardAllocated: string[max=15]
7	fileNameColumn	☐	VarChar			No	fileNameColumn: string
8	Description	☐	VarChar	100		No	Description: string[max=100]

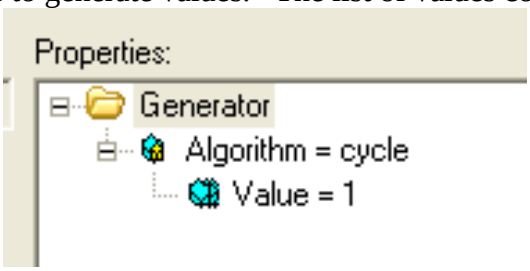
Task: Edit the ColumnGenerator stage

This stage is used to generate a new column named record_indicator.

1. Open up the ColumnGenerator stage.
2. On the Properties page, specify that the record_indicator field is to be generated.



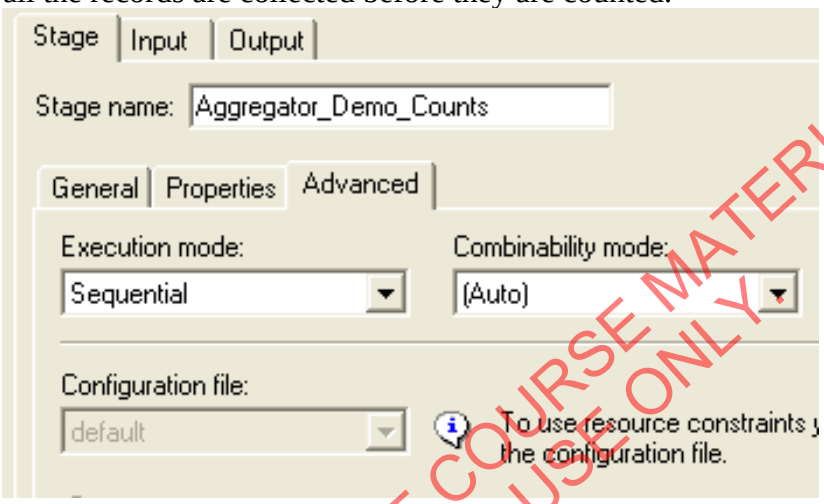
3. On the Output>Columns tab, click the right mouse button over the record_indicator field and click Edit row. This opens the Edit Column Meta Data window.
4. Select cycle as the algorithm to use to generate values. The list of values contains just one value, 1.



Task: Edit the Aggregator stage

The Aggregator stage is used to count the number of rows and output the result to the Item_Load_Count output column. Because all the rows need to be counted, not just the rows in each partition, it is necessary to execute the Aggregator stage in Sequential mode.

1. Open up the Aggregator stage.
2. On the Properties tab, group the records by record_indicator.
3. Specify the aggregation type as Count Rows and select the Item_Load_Count field as the Count Output Column.
4. On the Stage Advanced tab, select Sequential as the Execution mode. This insures that all the records are collected before they are counted.

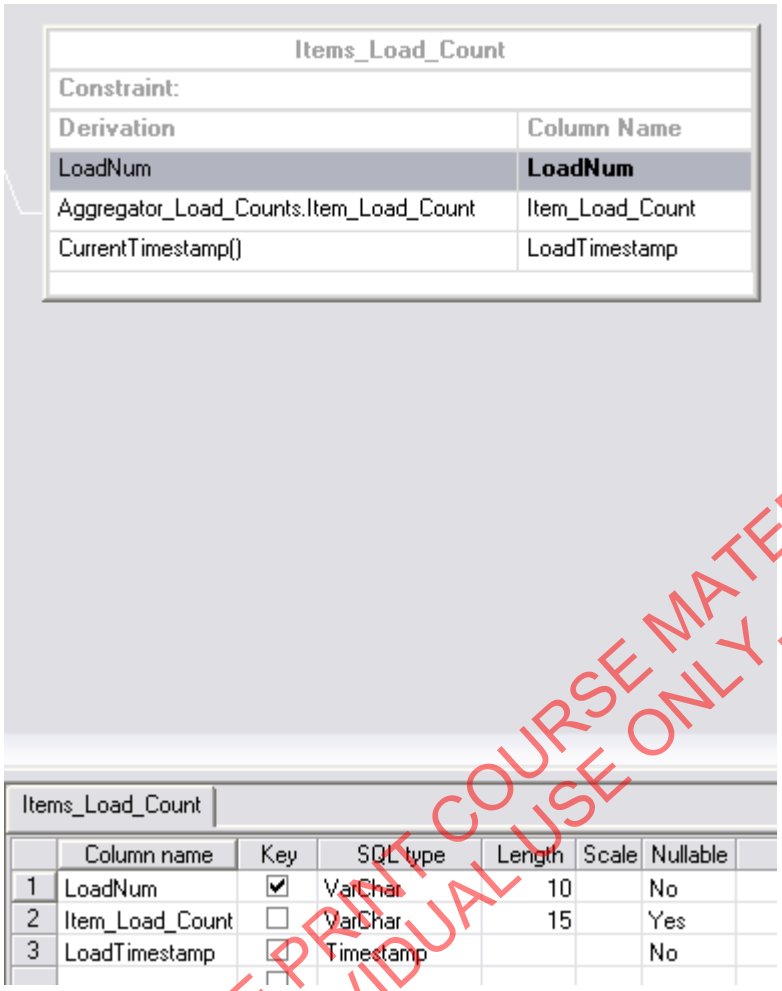


5. Only one column goes out of the Aggregator stage, namely, Item_Load_Count.

Task: Edit the Transformer stage

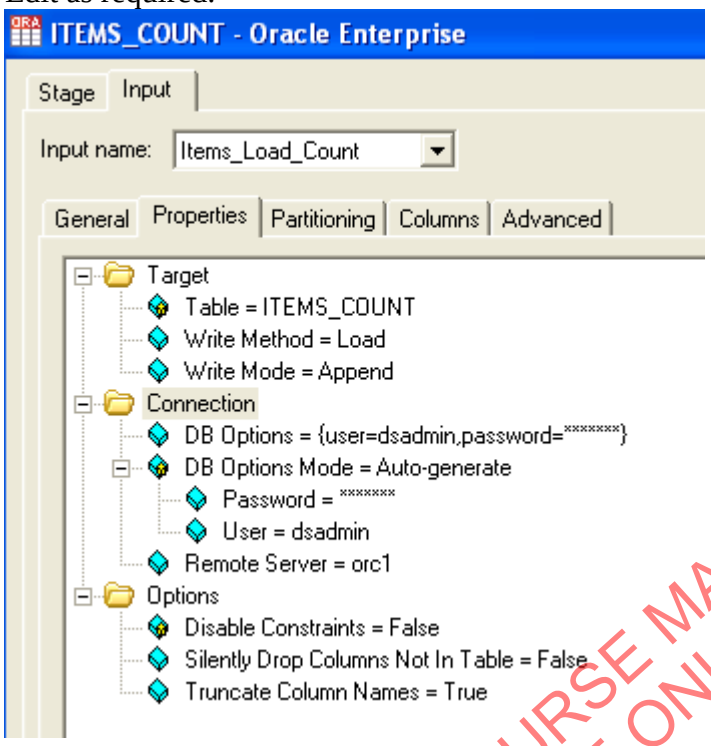
1. Open up the Transformer stage.
2. Create a new column named LoadNum. Make it a key column. It's derivation is a mapping from the LoadNum job parameter.
3. Map the Item_Load_Count column through the Transformer.

4. Create a new column named LoadTimestamp. Store the timestamp when the table is loaded.



Task: Edit the ITEMS_COUNT Oracle stage

1. Edit as required.



Task: Compile and run

1. Compile and run your job.
2. Open up Director and look at the runtime log.
3. Test and verify that your job is performing correctly.

Build Warehouse_02_Lookup Job

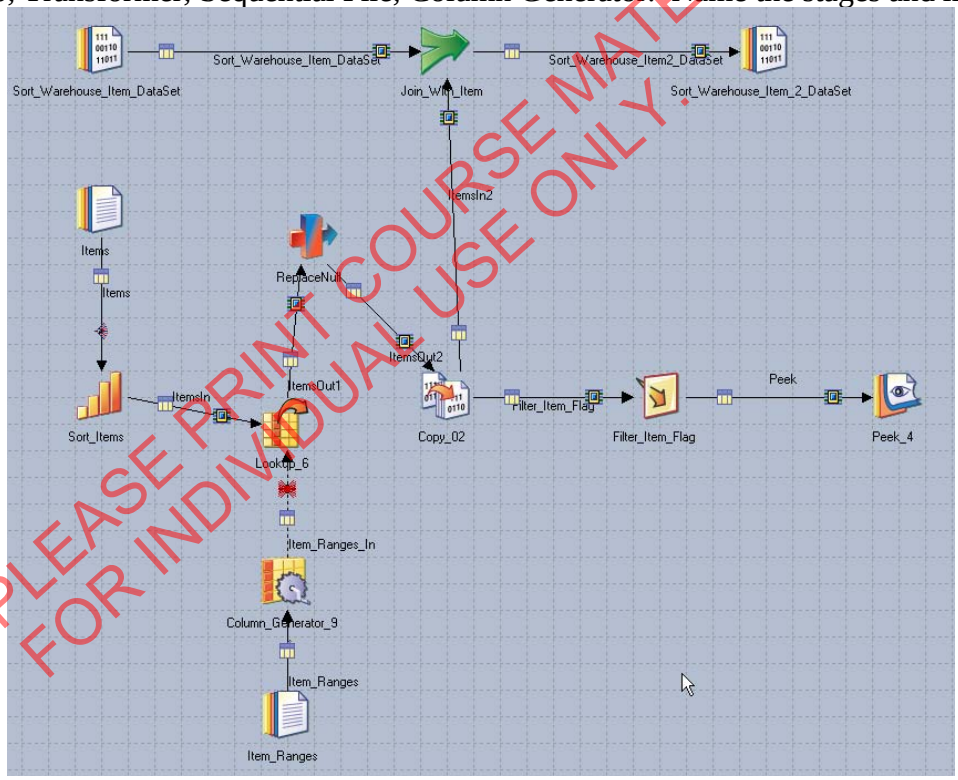
Description: In this job you set a flag by looking up the item from a list.

Assumptions

- You have created and successfully executed the Warehouse_02 job. If you haven't, import the Warehouse_02.dsx file and, if necessary, the Warehouse_01.dsx file from your Lab>Solutions folder on the Student CD.

Task: Create the job design

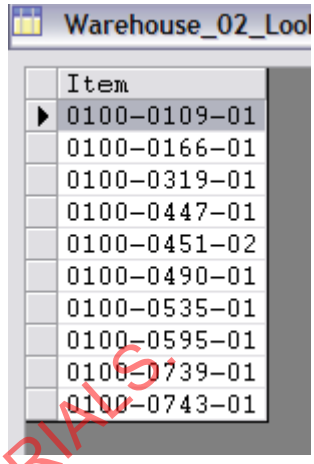
1. Open up your Warehouse_02 job and save it as Warehouse_02_Lookup. Store it in your _Training>Jobs.
2. Add the stages and links shown below just before the Copy stage. New stages are: Lookup, Transformer, Sequential File, Column Generator. Name the stages and links as shown.



Task: Edit the Item_Ranges Sequential File stage

1. This file contains a list of items of records that need to be audited, that is, for records that need to have their Item_Flag set to 'Y'.

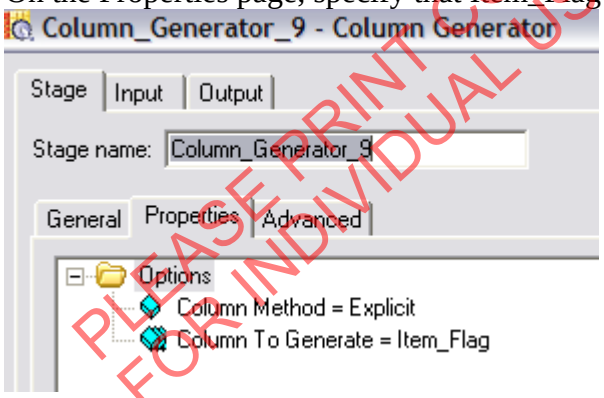
2. On the Properties tab, specify the path to the source file Item_Ranges.txt.
3. On the Columns tab load the Item column from the Table Definition for Items.txt.



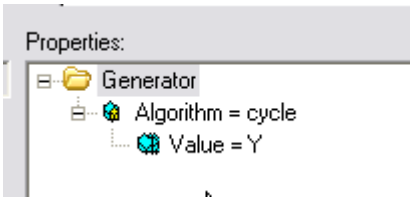
4. View the data in the file to see what items are to be flagged.

Task: Edit the ColumnGenerator stage

1. This stage is used to generate a new column named Item_Flag and assign it the value 'Y'.
2. Open up the ColumnGenerator stage.
3. On the Properties page, specify that Item_Flag field is to be generated.



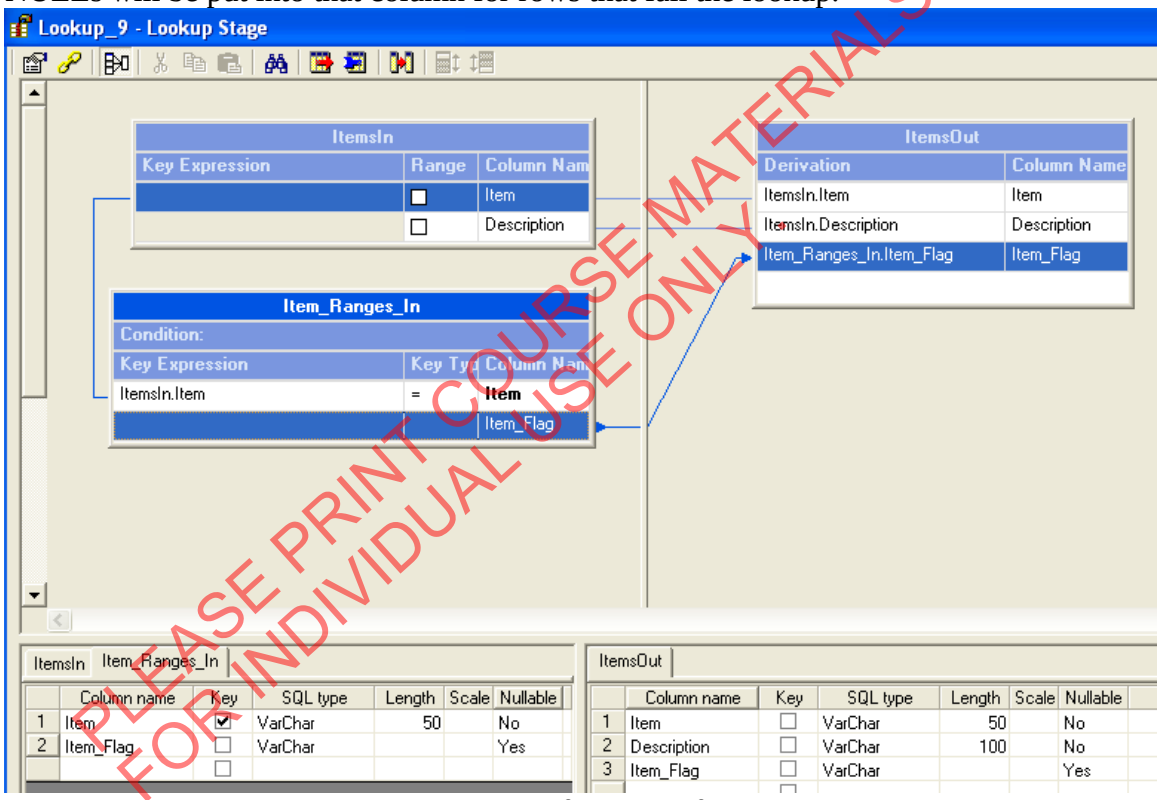
4. On the Output>Mappings tab, drag the two columns across.
5. On the Output>Columns tab, click the right mouse button over the Item_Flag field and click Edit row. This opens the Edit Column Meta Data window.
6. Select cycle as the algorithm to use to generate values. The list of values contains just one value, Y.



Task: Edit the Lookup stage

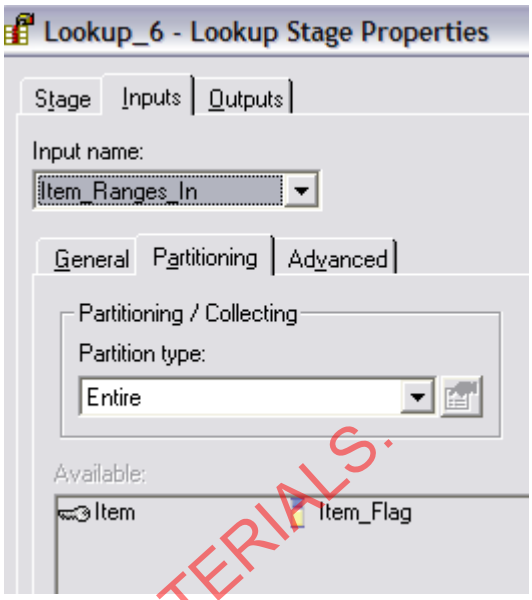
The Lookup stage is used to lookup whether the Item is for a record that needs to be audited. Those that should be audited will return a 'Y' in a one character field named Item_Flag. If the lookup fails, NULL will returned.

1. Open up the Lookup stage.
2. Specify the key expression for the lookup key, Item, by dragging the Item field from the incoming link to the lookup link.
3. Drag the Item_Flag lookup field to the target.
4. All other columns should be mapped across.
5. On the Constraints tab, specify that the job is to continue if a lookup match isn't found.
6. Important: Be sure the Item_Flag column on both the input and output is nullable so that NULLs will be put into that column for rows that fail the lookup.



7. Click the Stage Properties icon at the top left corner of the stage. Then click Inputs>Partitioning. Select the link coming from the lookup file (Item_Ranges_In).

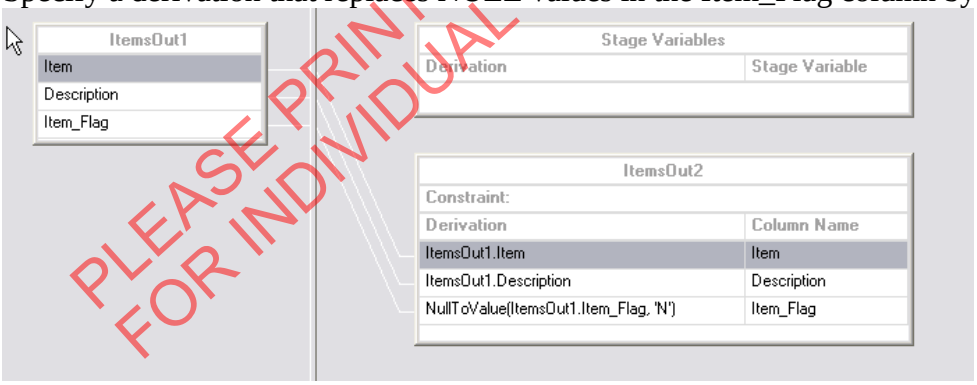
Select Entire as the Partitioning algorithm. This will insure that the values from the lookup file are available in every partition.



Task: Edit the Transformer stage

The Transformer stage is used replace the NULL values returned in the Item_Flag column when the lookup fails. These are to be replaced by 'N', meaning "Don't audit."

1. Open up the Transformer stage.
2. Map all columns across the Transformer.
3. Specify a derivation that replaces NULL values in the Item_Flag column by 'N'.



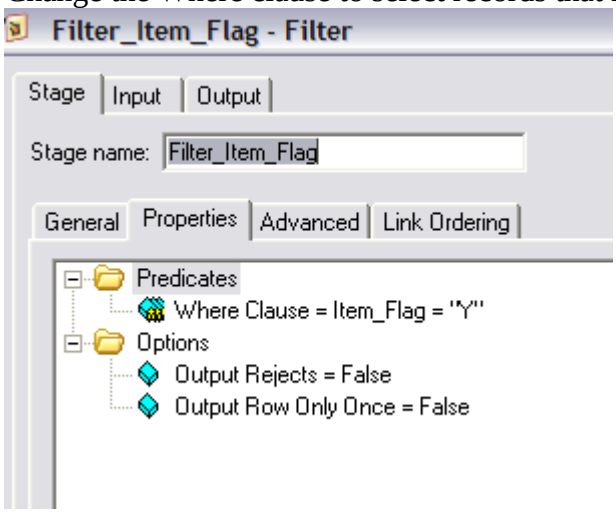
Task: Edit the Copy stage

1. On the Copy Mappings tab, verify that columns correctly map across to the two output links. Also drag the Item_Flag column across to the output link going to the Filter stage.

Task: Edit the Filter stage

Here we change the Where clause to select records that have an Item_Flag equal to 'Y'.

1. Open up the Filter stage.
2. Change the Where clause to select records that have an Item_Flag equal to 'Y'.



Task: Compile and run

1. Compile and run your job.
2. Open up Director and look at the runtime log. You see Peeks for the 10 rows in which the lookups find matches.

Item	Description
0100-0319-01	Test Board, POSEM 1
0100-0447-01	Power Detacher II,BEIGE,W/UNIV
0100-0451-02	ASSY,MULT,VIDEO OVERLAY CARD,W
0100-0739-01	IBM 4680/4690
0100-0109-01	U*M Doublechecker UNDRCTR MNT
0100-0166-01	Enclosure,DMS,PKG,W/HARDWARE
0100-0490-01	WALL MOUNT, 48
0100-0535-01	PACKAGED ASSY,UNDERCONVEYOR,RP
0100-0595-01	JBOX,SENSORNET,SIX POSITION
0100-0743-01	NCR 2127

Special Topic 9: Business Glossary

Lab 1: Accessing the IBM Information Server Business Glossary

Objective: Access the Glossary

Although performing tasks within the Business Glossary is very intuitive, the following exercises have been created to aid in your familiarization with the Glossaries basic activities and navigation.

The glossary is accessed by using Microsoft® Internet Explorer version 6.

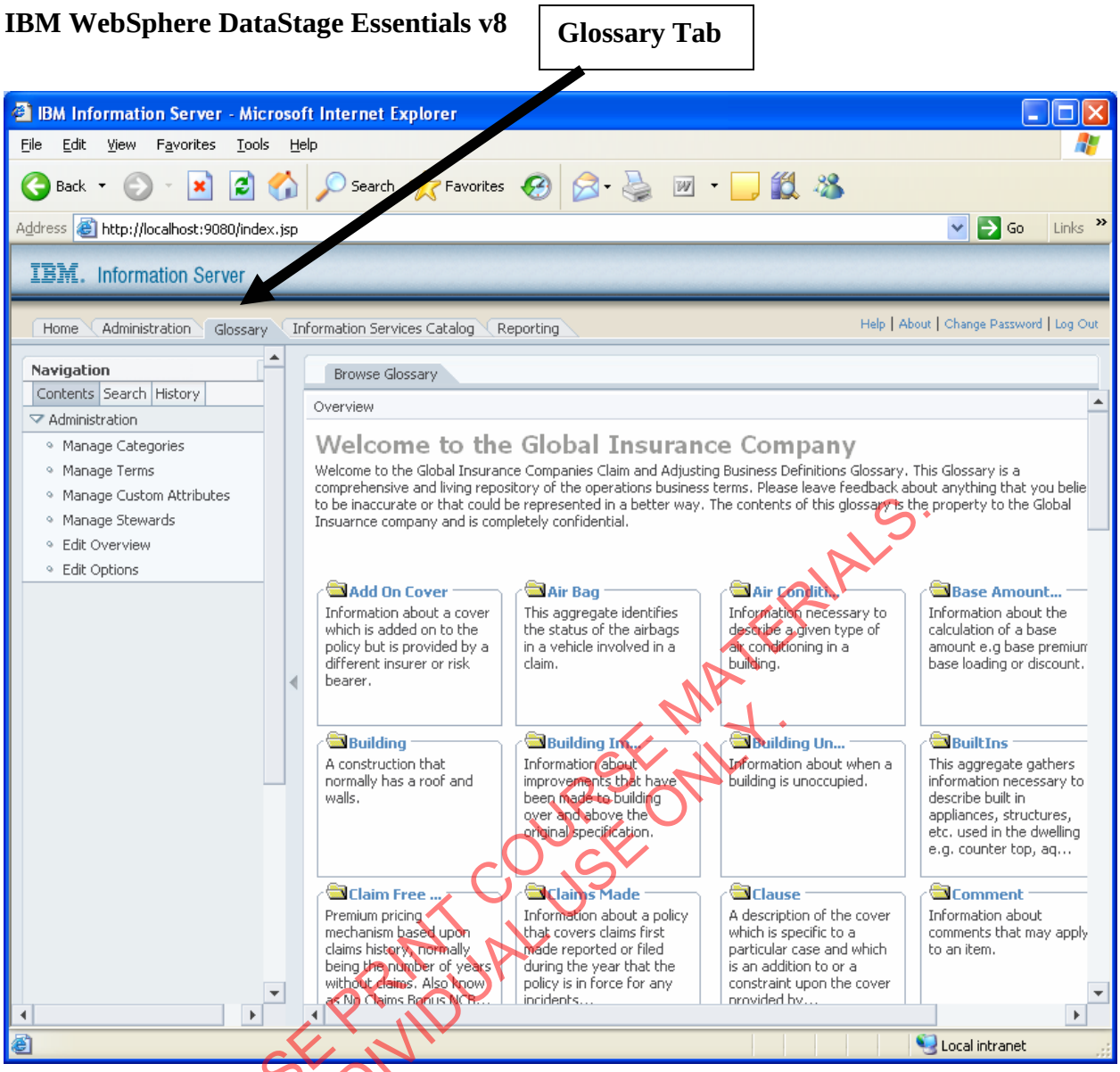
To access IBM Information Server Business Glossary:

1. Open Internet Explorer and connect to the following URL: http://host_server:port, where *host_server* is the name or IP address of the IBM Information Server Web console for the WebSphere Metadata Server that you want to connect to.
2. Enable popups for this site in your Web browser.
3. Type the user name and password for the IBM Information Server Web console, and click **Enter**.
4. Select the **Glossary** tab.

For example:

- To connect to a server on your network named Andros, type **http://Andros:9080**.
- To connect to a server whose IP address is 666.555.44.333, type **http://666.555.44.33:9080**.
- Very possibly for the purpose of this class the server address will be **http://localhost:9080**.

The following screenshot is the IBM Information Server Web console with the Glossary tab showing. This is only visible is you have be granted Business Glossary user, author or administrator role. You will initially have different content than that in the screenshot. If the Glossary tab is not visible then you will have to add the appropriate suite user role. For the purpose of these exercises, grant an existing or new user the Business Glossary Administrator Role. This must be done by a suite administrator.



Lab 2: Administration Console

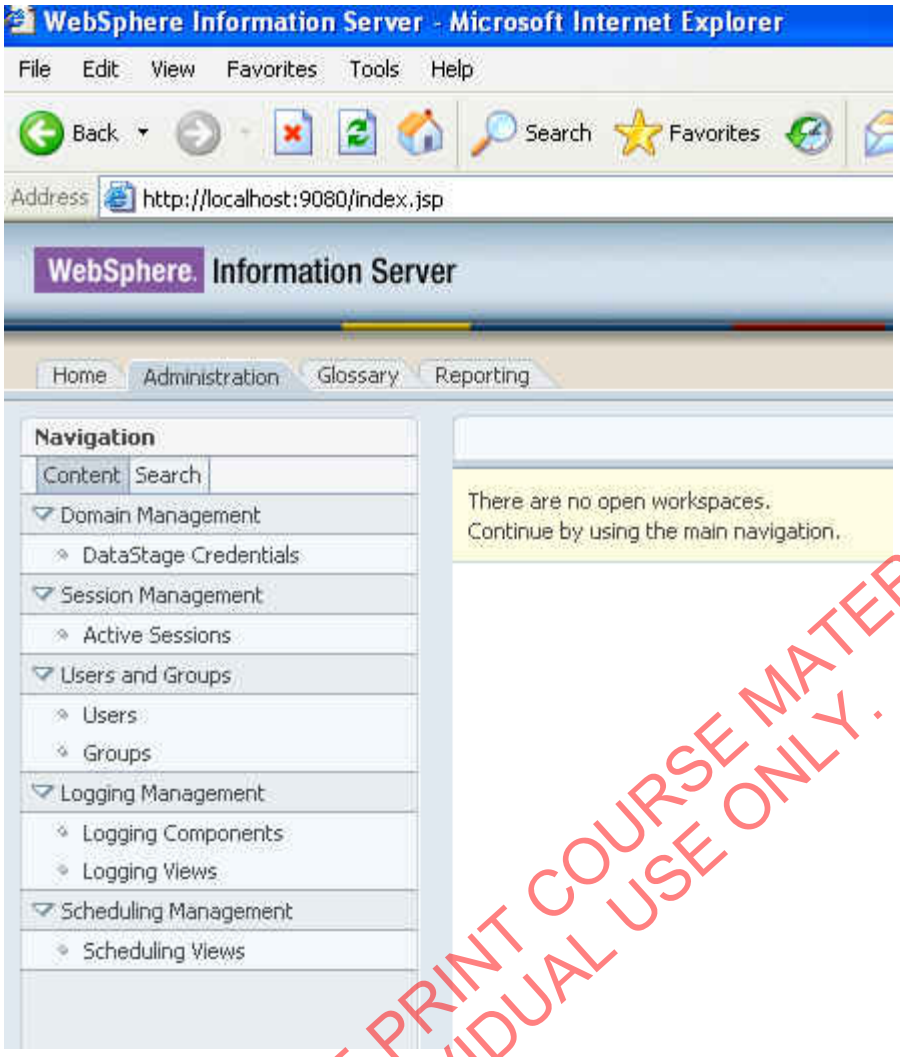
Assumptions

- A Suite user named demohawk has been defined. In this exercise will grant the demohawk user the Business Glossary Administrator role. It is possible that this role has already been granted to the demohawk user. But going through these steps will help in understanding the administration console and the types of privileges and roles utilized in the suite.

Task: Open the Administration Console

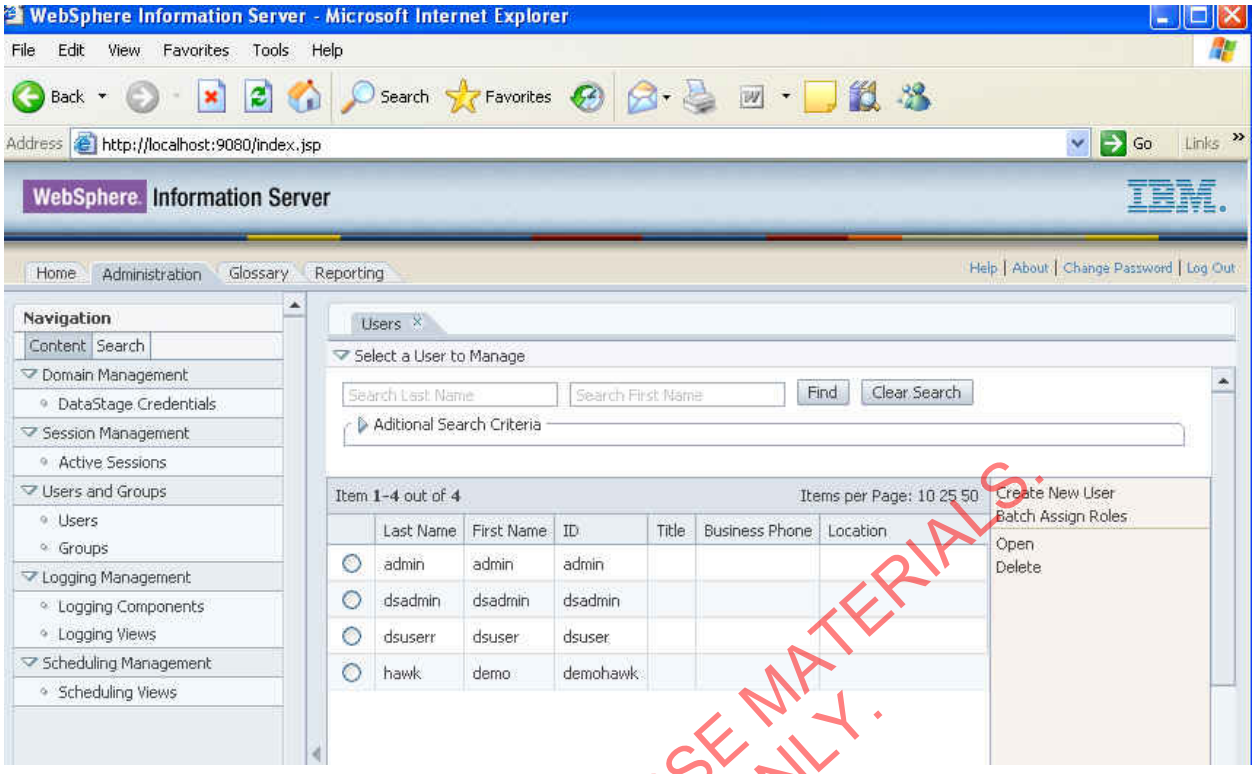
3. Open your web browser.
4. Enter the address to your Administration Console, e.g., http://localhost:9080. Here, localhost is an alias for your local machine. If the Administration console is running on a different machine, use the name or IP address of this machine instead of localhost.

5. Click the Administration tab.

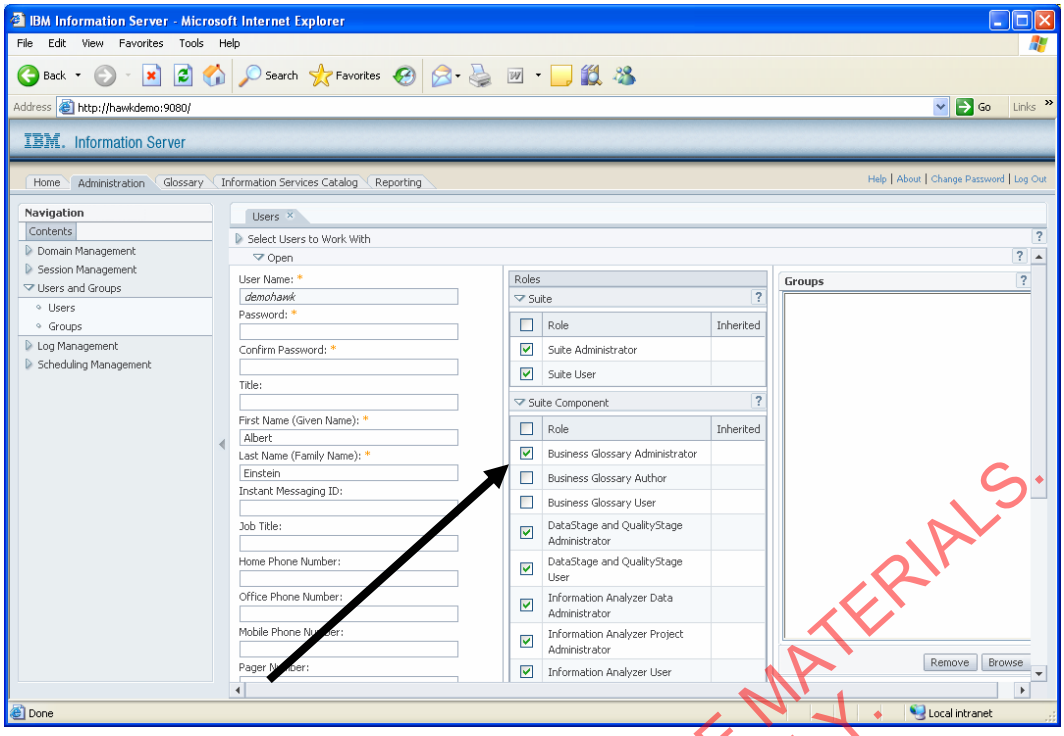


Task: Create add Business Glossary Admin Role to user demohawk

11. Expand the Users and Groups folder and then click Users.



12. Select the demohawk user and then click Open.
13. Note the first and last names of this user. Note what Suite Roles and Product Roles that have been assigned to this user.
14. Assign demohawk the Business Glossary Administrator product role.



Lab 3: Upload Categories and Terms

Objectives:

Upload the Business Categories and Terms into the Global Insurance Glossary of business categories and terms.

Assumptions:

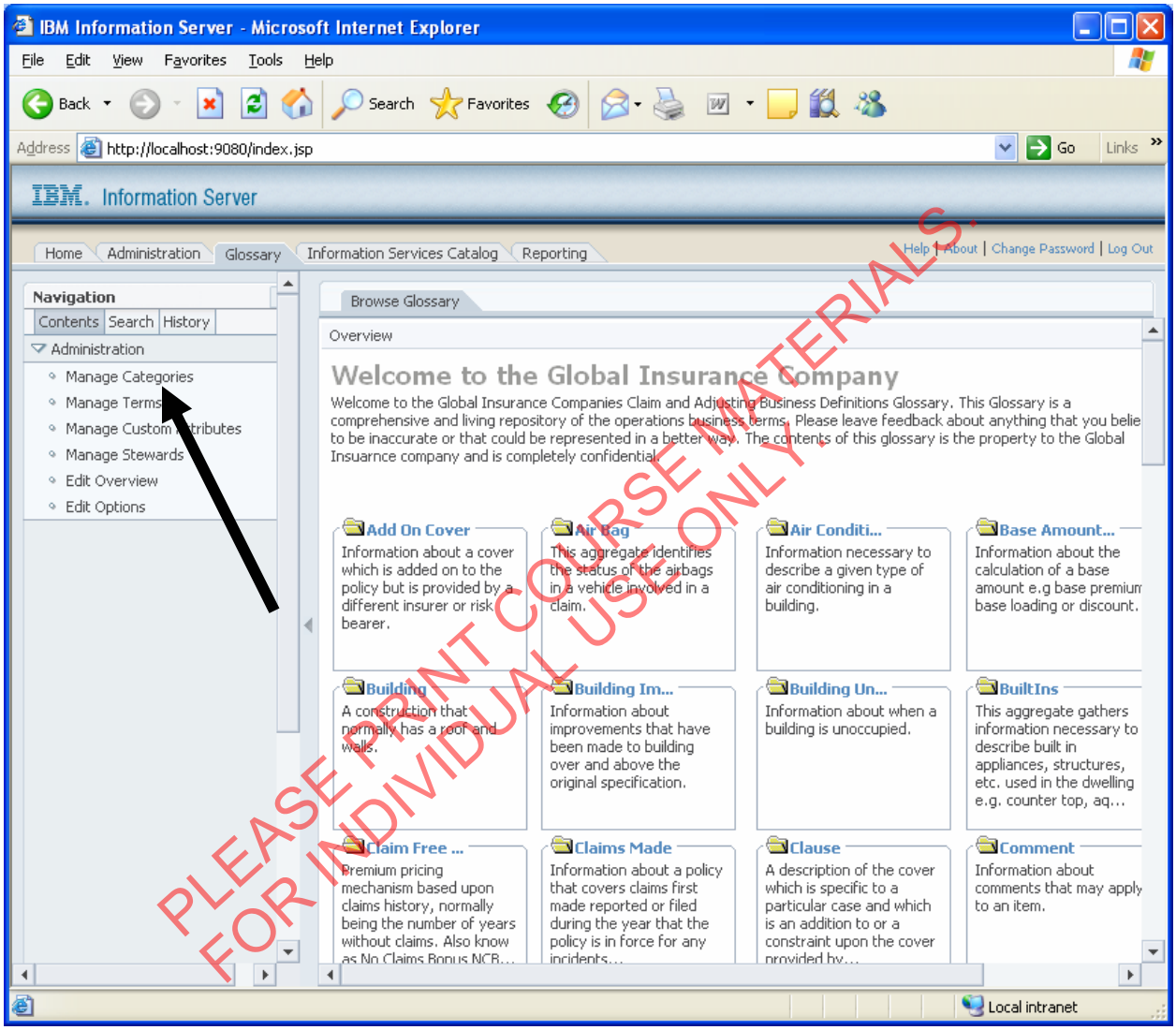
- You must have the Business Glossary Administrator role or Business Glossary Author role to perform this task.
- You must prepare an XML file that complies with the IBM® WebSphere® Business Glossary category and term data file schema. Links to the schema and to a sample XML file that complies with the schema appear in the Upload Categories and Terms page.
- Names of categories and terms must start and end with a character that is not a space. Names cannot contain any of the following characters:
  - . (period)
  - , (comma)
  - ; (semicolon)
  - % (percentage sign)
  - " (quotation marks)
- You must use the appropriate entity references for reserved characters in XML:

For this reserved character	Use this entity reference
< (left angle bracket)	<
> (right angle bracket)	>
" (quotation marks)	"
' (apostrophe)	'
& (ampersand)	&

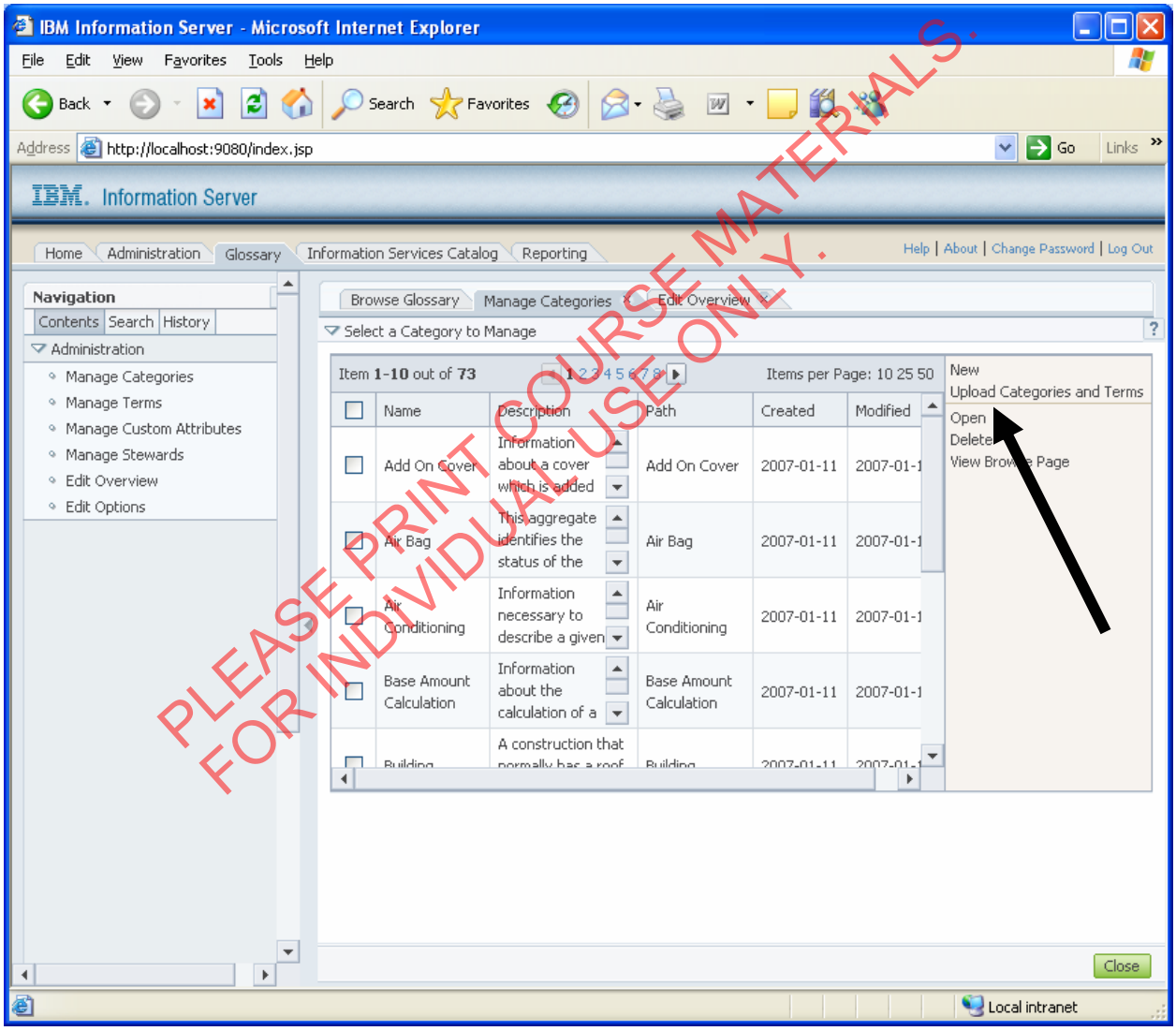
Task: Upload the Categories and Terms

Several categories and terms have been provided in the appropriate xml format. The filename is **Business Glossary Upload Sample (Insurance version 1).xml**. This file is contained in a folder on your student CD called Business Glossary.

1. To upload categories, terms, and custom attributes:
   In the Navigation pane on the Glossary tab, select **Contents** > **Administration** > **Manage Categories**.



2. Click **Upload Categories and Terms**.



3. Click **Browse**, navigate to an XML file, select it and click **Open**.

4. Click **Upload**. The Upload Categories and Terms page displays the number of categories and terms that were uploaded.
5. Click **OK**.

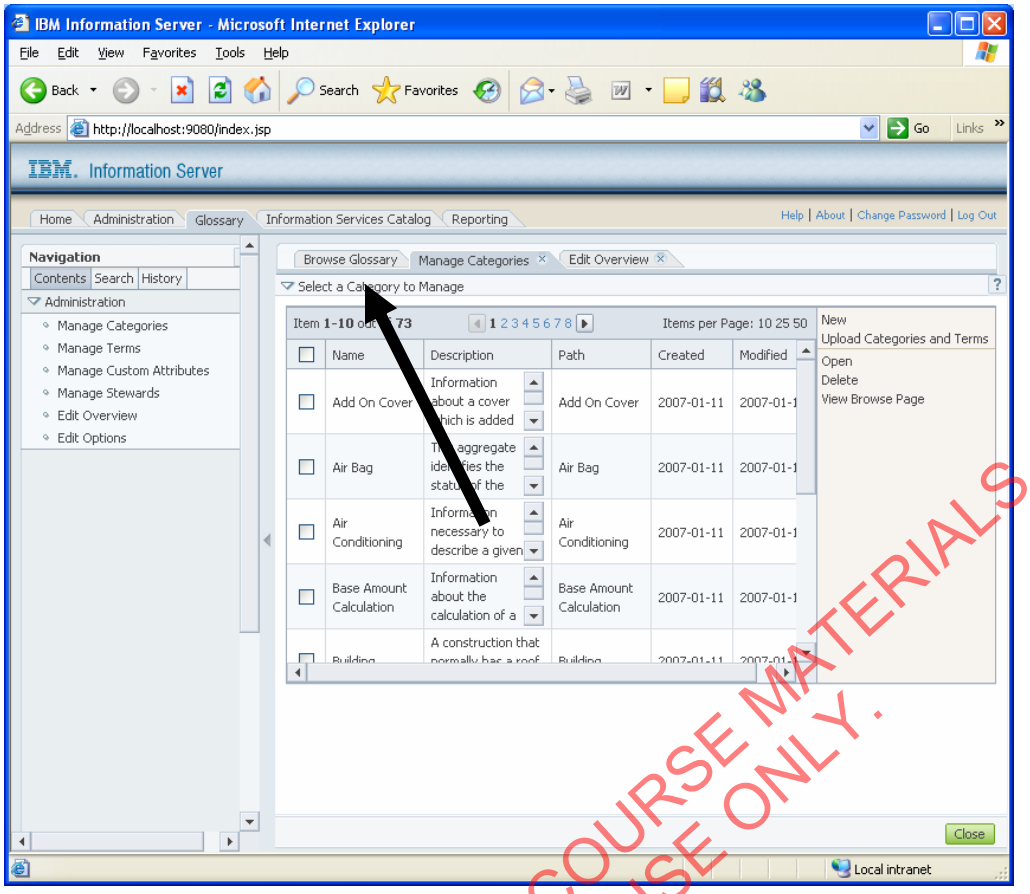
At this point you should now have a variety of categories and terms to browse and search in the glossary.

Lab 4: Browsing the glossary

Overview:

You can browse the glossary structure to explore categories, terms, and objects in the repository of your IBM Information Server.
You can start browsing the glossary from the Overview page, which displays the top-level categories that the glossary administrator has designated as most important for navigation in the metadata repository. You can also search for objects and select an object from the search results. When you select an object, the browse page of the object is displayed on the Browse Glossary tab, which lists the name, class, steward and other important properties of the object. You can inspect the attributes of the object, browse its relationships to other objects, and send feedback to the administrator. Administrators and authors can add and edit notes about the object.

1. Navigate back to the “Home” page. The “Browse Glossary tab will bring you home.



2. Select a category such as **Coverage Option** and browse its contents and available tasks. Remember that the availability of the tasks is governed by the user role.

Task: Searching the Glossary

Overview:

Using the search tool is often the quickest way to find an object in the repository of WebSphere® Metadata Server.

You can perform simple and advanced searches to find repository objects of all classes, including, but not limited to, categories, terms, tables, columns, job definitions, users, and groups. The more information that you can specify about the object that you are searching for, the faster the search results are returned.

The containment path of the object is displayed in the Path column in the search results so you can distinguish between objects with similar or identical names. For example, the containment path for a column might display the names of the containing table, schema, database, and host computer the column was imported from.

When you locate the object in the search results, you can click its name to display the browse page for the object. You can then inspect its attributes, browse its relationships to other objects, and send feedback to the administrator. Administrators and authors can add and edit notes about the object.

Finding Objects with the Simple Search

1. In the Navigation pane on the Glossary tab, select **Search** > **Simple Search**.

2. In the **Simple Search** field, type the search criteria. You can use all or part of a name or short description, or you can use multiple keywords from names or descriptions, separated by spaces or commas. In this case search for **Claims Made.**

3. Click **Search**. The Search Results page displays a list of objects in the metadata repository whose names or short descriptions match the search string. If you typed multiple keywords, the list includes the objects whose names or short descriptions include all of the keywords. As you will notice, many items will be returned. We will learn how to perform a more exacting search in the next task.

4. In the list, click an object name to view the object, its relationships, and its attributes.

Finding objects with an advanced search

You can use multiple criteria when you search for objects that are stored in the metadata repository. In this case, we will once again search for the term **Claims Made**, but for **Claims Made** only.

To find objects with an advanced search:

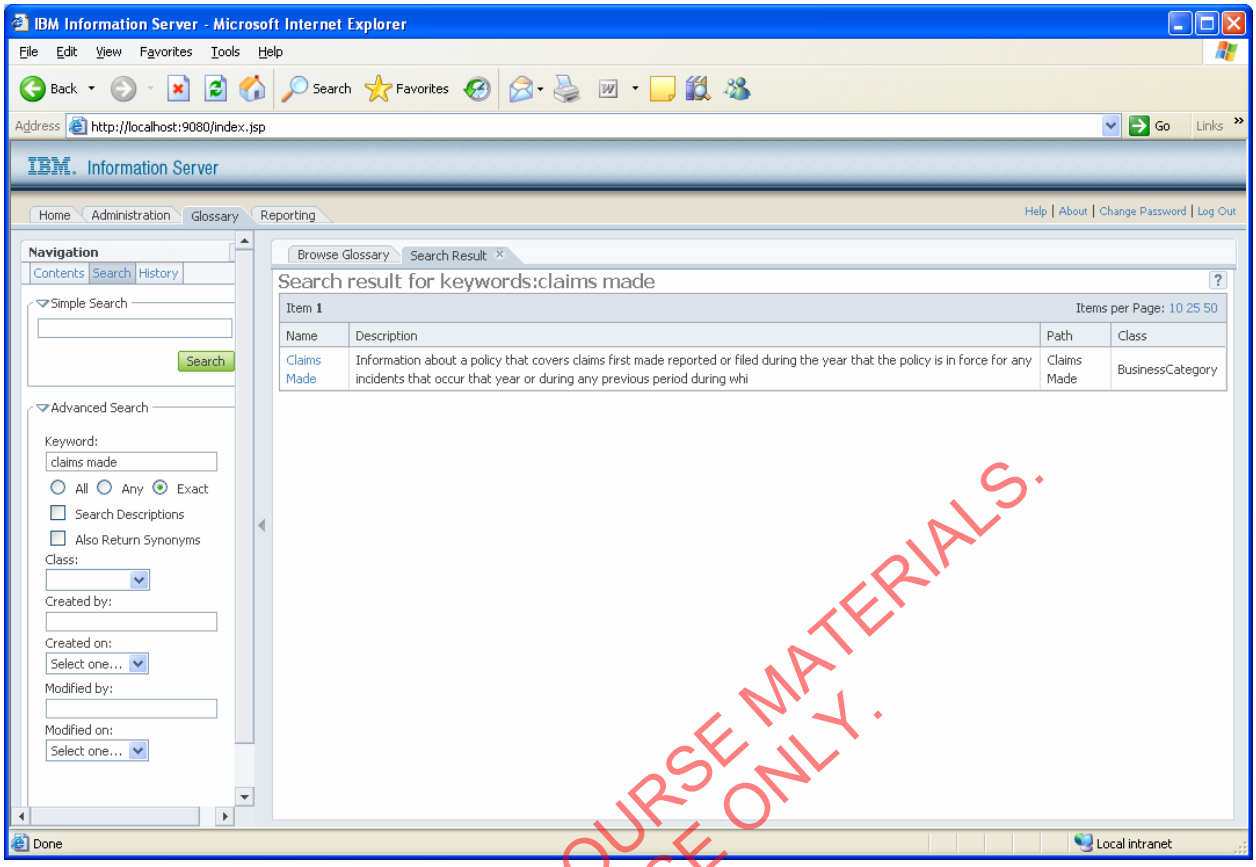
1. In the Navigation pane on the Glossary tab, select **Search** > **Advanced Search**.

   Specify the criteria for the search:

   - The keyword as before will be **Claims Made**.
   - Filter for only exact matches
   - Uncheck the Search Descriptions check box.

.

2. Click **Search**. A list of search results is displayed on the Search Results page. In this case, only the **Claims Made** term should be returned.



Lab 5: Business Category and Term Creation and Editing

Overview:

Administrators and authors can create and edit categories that contain or reference terms and that serve as subcategories or parent categories to other categories.

Assumptions:

In this task you will create a category named **MyCategory** or any other name you wish. If possible consider something that categorizes a facet of your business that could become a parent of a subcategory or business term. For example: the construction category contains terms such as material, material code, material percent etc.

To create or edit a category:

1. In the Navigation pane on the Glossary tab, select **Contents** > **Administration** > **Manage Business Categories**.

2. On the Manage Categories page, specify whether to create or edit a category:

    - To create a category, click **New**.
    - To edit a category, select a category from the list of all categories, and click **Open**.

3. If you are creating a category, in the **Name** field, type a name for the category. If you are editing an existing category, you can change the name.

4. **Optional:** Specify or change information about the category. You can add or edit descriptions, specify a steward, define relationships to other categories and terms, specify values for custom attributes, and specify whether the category is displayed on the Overview page.

5. Click **Save and Close** to save your changes and close the category.

Task: Business Term Creation and Editing

Overview:

Administrators and authors can create and edit terms to categorize one or more metadata objects in the metadata repository.

Assumptions:

In this task you will create a business term named **MyBusinessTerm** or any other name you wish. Think about how well the category you just created is named in relation to the term you are about to create.

To create or edit a category:

Creating and editing terms

Administrators and authors can create and edit terms to categorize one or more metadata objects in the metadata repository.

Prerequisites: You must have the Business Glossary Administrator role or Business Glossary Author role to perform this task. A category must exist to contain the term. As in the previous To create or edit a term:

1. In the Navigation pane on the Glossary tab, select **Contents** > **Administration** > **Manage Business Terms**.

2. On the Manage Terms page, specify whether to create a term or edit a term:

    - To create a term, click **New**.
    - To edit a term, select a term from the list of all terms, and click **Open**.

3. If you are creating a new term, specify its name and parent category. If you are editing the term, you can edit the name and parent category.

    - In the **Name** field, type a name for the term.
    - Next to the **Parent Category** field, click **Select** to select a parent category to contain the term.

4. **Optional:** Specify a Data Steward when the term is created. This can be done at a later time by editing the term. Data Stewardship is a role that must be granted to a user or group.

5. **Optional:** Specify or change information about the term. You can add or edit descriptions, specify a steward, define relationships to other terms, classify objects, specify values for custom attributes, give an example of the term, specify the status of the term, specify how the term is used, and specify abbreviations for the term.

6. Click **Save and Close** to save your changes and close the term.

Lab 6: Working with Annotations. (Notes)

Overview:

Administrators and authors can add, edit and delete notes on the browse page of any object.

Assumptions:

In this task you will create a note on the business term of your choice. Then you can edit or delete it. The note might be an observation you have made about the data that you feel should be shared with others. This is an example of where it can be very helpful to have a Data Steward assigned to an object.

Task: Work with a note:

1. Display the browse page of an object by any of these methods:

- Browsing from the Overview page
- Finding objects by using a simple search
- Finding objects with an advanced search
- Browsing the properties and relationships of objects

2. Add, edit, or delete the note:

To add a note:	a. In the **Tasks** list, click **Add Note**. b. In the New Note window, type a label and comment for the note and click **OK**. The note is added to the Notes tab.
To edit a note:	a. On the Notes tab, in the row that describes the note that you want to edit, click  (edit note). The icon is not displayed if you do not have authority to edit the note. b. In the Edit Note window, type a label and comment for the note and click **OK**.
To delete a note:	a. On the Notes tab, in the row that describes the note that you want to delete, click  (delete note). The icon is not displayed if you do not have authority to delete the note. b. Click **Yes** to confirm deletion.

Lab 7: Working with Custom Attributes

Overview:

Administrators can create custom attributes to store information about terms and categories, when that information does not fit into the standard attributes and relationships of the glossary model. You can use custom attributes to apply governance standards, enable architecture frameworks, or provide other metadata that is standard for your organization.

When you create a custom attribute, you specify that it applies to either terms or categories, or to both terms and categories. If you apply the custom attribute to both terms and categories, two separate custom attributes are created, one that applies to terms, and one that applies to categories.

Each custom attribute has a name, a description, and a valid value type. The valid value type can be any string or an enumerated list of string values.

You can change the valid value type for a custom attribute at any time. When you change the type, the change does not affect any values that are currently assigned for the attribute. The change determines what will happen the next time a user edits the value for a custom attribute. If you change the type of a custom attribute to String, when users subsequently edit the attribute for any object, they can enter any string value. If you change the type of a custom attribute to Enumerated, when users subsequently edit the attribute for any object, they must select values from the enumerated list of values.

The value of the custom attribute for any particular term or category is initially null. After you create the custom attribute, you can specify its value separately for each term or category that it applies to.

For example, you might create a custom attribute named Data Sensitivity with the following description

A number from 1 to 5, which indicates the sensitivity of the data. Sensitivity is a subjective measure of the impact of the data being released to unauthorized consumers.

You can specify that Data Sensitivity attribute applies only to terms. You choose the enumerated valid value type and enter the numbers 1 through 5 as valid values. After you create the custom attribute, you choose one of those valid values for each particular term that you want to specify a value for.

Assumptions:

In this task you will create a custom attribute named **Data Sensitivity.** This attribute will be available to all of your business terms. It does not make sense to associate this attribute with any categories, although this is possible within the glossary to do so. This custom attribute will be of the enumerated type. A custom attribute can be created first and then associated with a term or category, or it can be done as part of the term or category editing process.

Task: Create and work with a custom attribute:

Prerequisite: You must have the Business Glossary Administrator role to perform this task.

To create a custom attribute:

1. In the Navigation pane on the Glossary tab, select **Contents** > **Administration** > **Manage Custom Attributes**.

2. Click **New**.

3. Type a name and description for the custom attribute. The name will be **Data Sensitivity**. Valid values are numbers 1-5.

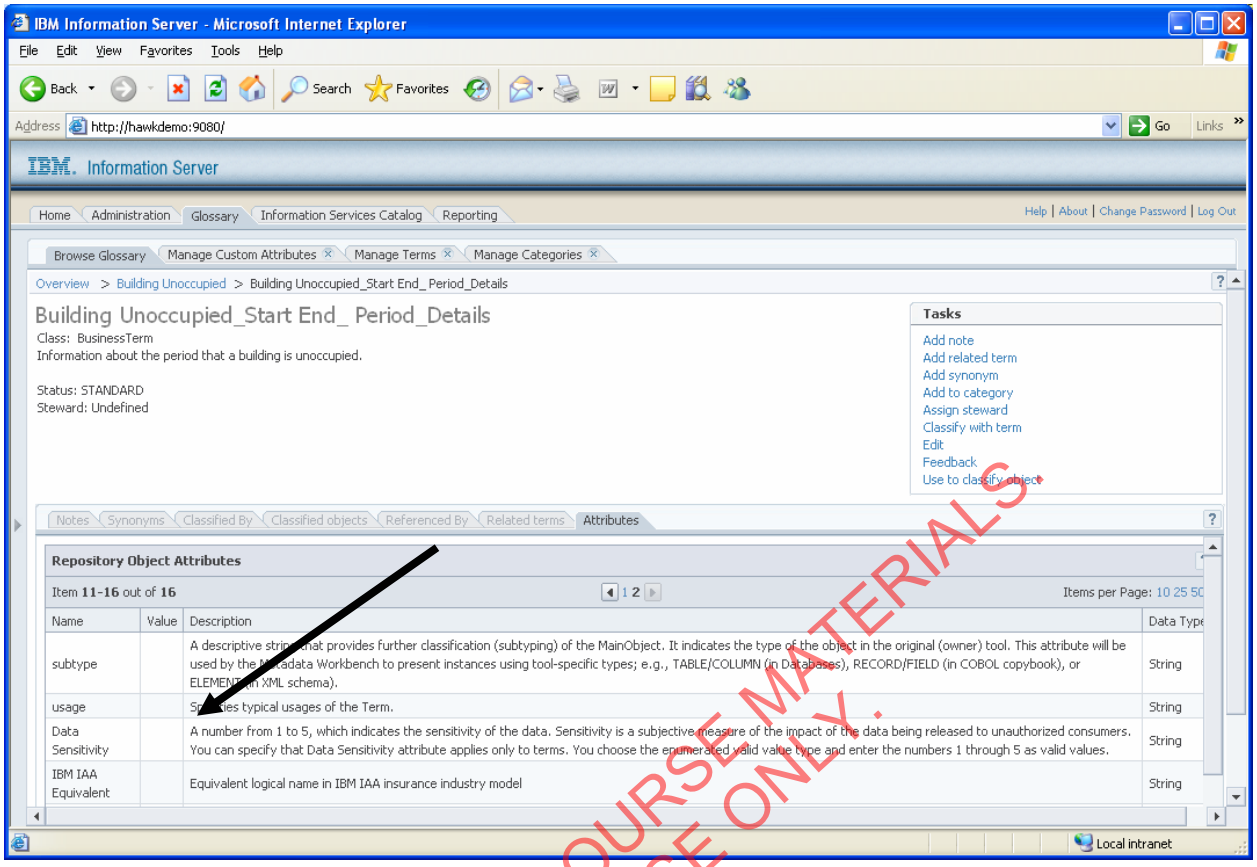
4. Select the class of object that the attribute applies to either categories or terms, or both. If you select both, two custom attributes are created with the same name and properties. One custom attribute applies to terms and the other applies to categories. In this case, it only makes sense to select **terms.**

5. From the **Attribute Type** drop-down list, select the type of valid value:

Option	Description
To specify that any string is a valid value:	Select **String**.
To specify a list of valid string values:	a. Select **Enumeration**. b. Type a single valid value, and click **Add**. c. Repeat step b to add valid values.

6. Click **Save and Close** to save your changes

7. Now browse any of your business terms and find the custom attribute you just created. This will be on the second page of any term you choose. You should notice the custom attribute and that it has no value.



8. Next enter a value for your new custom attribute **Data Sensitivity.** You will have to manage the term to do this.