**CUD XML to SQL CUD Receiver Database**

The set-up of the SQL database, jobs and SSIS package have all been carried out on:

1. Windows 2008 R2 Server
2. SQL Server 2008 R2

Using:

1. SQL Server Management Studio
2. SQL Server Business Intelligence Development Studio
3. SQL Server Profiler

The requisite SQL Scripts and SSIS Package can be downloaded here in the folder named “*CUD XML to SQL and SSIS Solution*”:

<https://sharepoint.nexus.ox.ac.uk/sites/SSP/interfaces/SITS%20Student%20Data%20Feed/CUD%20XML%20to%20SQL%20and%20SSIS%20Solution.zip>

|  |
| --- |
| **Important – Please Read**  This document requires the transformed xml file produced by the PowerShell script here:  <https://downloads.oucs.ox.ac.uk/sysdev/cud/cudwspulltransform.ps1>  Details of how to:   1. Set up access to the CUD Web Service 2. Set up the CUD interface 3. Execute the above PowerShell 4. Archive xml files 5. Schedule batch files to run daily   and the location for the relevant:   1. Login.conf 2. NegotiateRestClient.jar 3. Krb5.conf 4. Template PowerShell scripts 5. The required XSL file   Can all be found in this document here:  <https://sharepoint.nexus.ox.ac.uk/sites/SSP/interfaces/SITS%20Student%20Data%20Feed/CUD%20Web%20Service%20Access.pdf>  Please be aware, the following set up instructions relies on certain pathfiles and filenames.  Any changes you make to a folder/file location or name needs to be reflected in the relevant stages of the import. |

**Index**

1. **Pre-requisites**
2. **Setting up the CUD Receiver Database**
   1. Creating the SQL DB CUD\_Receiver database
   2. Creating the CUD\_Receiver tables
3. **Setting up the SSIS package**
   1. Opening the SSIS Package
   2. Control Flow Items
      1. *Data Flow Tasks*
      2. *Inside a Data Flow Task*
         1. *The Data Flow Source tool*

*The Data Flow Transformation Tools*

* + - 1. *Sort*
      2. *Merge Join*
      3. *Data Conversion*
      4. *SQL Server Destination*
  1. Selecting the Correct SQL Server Destination
  2. Selecting and Mapping to the Correct Tables
  3. Importing the package to SQL Server

1. **Setting up SQL Server Jobs**
   1. The Truncate CUD\_Receiver Tables Job
   2. Run the SSIS Package Job
2. **Pre-requisites**
3. Folder on the C:\ Drive of your server named CUD-client i.e. *C:\CUD-client.*
4. XML file transformed and named “**CUD\_Output\_Transformed.xml**” using the XSL file named “**CUD\_XML\_Transform.xsl**”. This file must be saved in the above folder *C:\CUD-client.*

*Please see the documentation noted on page 1 of this document for details on the transformation, and locations of the requisite files.*

1. XSD file “**CUD\_Output\_Trans.xsd**”. This file must be saved in the above folder *C:\CUD-client.*

*The XSD file* “**CUD\_Output\_Trans.xsd**” can be found in the “*CUD XML to SQL and SSIS Solution*” folder.

*If you choose to alter the above folder location/name or any of the file names/locations you will need to alter these in Connection Manager of the SSIS package and each of the “XML Source” items in the data flow tasks.*

1. **Setting up the CUD Receiver Database.**

You will need to create the CUD\_Receiver Database. The scripts to create this and the relevant tables are supplied. You may wish to create your own database and tables, any differences between the supplied scripts and your own database/tables/attribute names would have to be reflected in the SSIS package.

* 1. **Creating the SQL DB CUD\_Receiver database**

|  |
| --- |
| *Whilst it is not necessary to keep the database name CUD\_Receiver, please be aware that the supplied SQL scripts will need to be altered to match whichever Database name you use.*  The SQL script “Create CUD\_Receiver Database.sql” will create the database named “CUD\_Receiver”:  **Please note** *“FILEGROWTH” has been set to 10% and no Max size has been specified. The recovery mode is automatically set to “Full” in order to set up “Transaction Log” back-ups which help to keep the “.ldf” size to a minimum.*   1. Open SQL Server Management Studio 2. Connect to Server Instance 3. Select File>Open>File… 4. Select “Create CUD\_Receiver Database.sql” 5. Enter the pathfile for the .mdf 6. Enter the pathfile for the .ldf 7. Execute   The CUD\_Receiver database should now be in the list of databases.     1. Click on “Databases” and refresh, when you expand the “Databases” list is should appear as above. |

* 1. **Creating the CUD\_Receiver tables**

|  |
| --- |
| **Step 2** – Create CUD\_Receiver Tables  *Whilst it is not necessary to keep the table names in the script, please be aware that documents and the SSIS data tasks all refer to these table names.*  The SQL script “Create CUD\_Receiver Tables.sql”will create the tables required in the CUD\_Receiver database:   1. Select File>Open>File… 2. Select “Create CUD\_Receiver Tables.sql” 3. Execute 4. Click on the CUD\_Receiver database and refresh   The CUD\_Receiver database should now have the following tables listed     1. Click on “CUD\_Receiver” and refresh, when you expand the “Tables” list, the tables should appear as above. |

1. **Setting up the SSIS package**

This package has pre-configured connections to the:

1. Folder on the C:Drive of your server named CUD-client i.e. *C:\CUD-client*
2. XML file named “**CUD\_Output\_Transformed.xml**” saved in the folder *C:\CUD-client.*
3. XSD file “**CUD\_Output\_Trans.xsd”** saved in the folder *C:\CUD-client.*

As previously noted, if any of the above path files or file names have been changed; those changes will need to be reflected in the pre-configured SSIS package.

* 1. **Opening the SSIS Package**

|  |
| --- |
| Using the pre-configured SSIS package “CUDXMLtoCUDTables.dtsx”:   1. Open Business Intelligence Development Studio 2. Select File>New>Project>Integration Services Project      1. Input a name for your project e.g. CUD XML to CUD\_Receiver 2. In the Solution Explorer, right click on SSIS Package and select Add Existing Package 3. Alter Package Location to File System      1. Select the Package path: and navigate to the location where you have saved the above package CUDXMLtoCUDTables.dtsx. 2. Click on OK   You should be presented with a screen below:  *If the Control flow screen below does not appear, double click on the Package you have just added, which should have appeared in the Solution Explorer.*   1. There will be a delay at this point as BIDS attempts to validate all the tasks within the package.   Once validation is complete (all of the data flow tasks below will have little red crosses in them as there is no destination in the connection managers) you should be presented with a screen as follows:  This is the “Control Flow” screen. This screen displays all the tasks within the package and the order they will run.    [Type a quote from the document or the summary of an interesting point. You can position the text box anywhere in the document. Use the Drawing Tools tab to change the formatting of the pull quote text box.]  [Type a quote from the document or the summary of an interesting point. You can position the text box anywhere in the document. Use the Drawing Tools tab to change the formatting of the pull quote text box.]  It also displays the connection pre-configured within the package. In this case the  *CUD\_Output\_Transformed.xml* |

* 1. **Control Flow Items**
     1. *Data Flow Tasks*

|  |
| --- |
| You can see the following tasks are all “Data Flow Tasks” which are all named for the tables in the CUD\_Receiver database you have created.    Each “Data Flow Task” selects the data for each table from the *CUD\_Output\_Transformed.xml* and loads it into the corresponding CUD\_Receiver table.  Precedence Constraints have been set so that only one task at a time accesses the XML file. |

* + 1. *Inside a Data Flow Task*

|  |
| --- |
| To access the individual data flow tasks select the Data Flow tab    You can use the drop down menu to select the Data Flow Task you want to view/edit |

|  |
| --- |
| I have used the “Applicant” data flow task as an example to describe what makes up each of the data flow tasks in this package.  When the Applicant task is open you will see a screen like this:    There are five types of tools in the data flow tasks:   1. **One** Data Flow Source: 2. Here an XML Source - named CUD XML Transformed: 3. **Three** Flow Transformations tools: 4. 4 Sorts – named cudid, applicant, Application, and SorOnApplicantId 5. 2 Merge Joins – named cudid+applicant, and +Application 6. 1 Data Conversion 7. **One** Data Flow Destination: 8. Here an SQL Server Destination - named for the target table in the CUD\_Receiver database |

* + - 1. *The Data Flow Source tool*

|  |
| --- |
| The Data Flow Source in this task is an XML Source, which is the CUD\_Output\_Transformed.xml.  The tool has already been pre-configured, but you can check it and confirm it by:   1. Right clicking on the tool 2. Selecting Edit   You can see the XML Source Editor, the XML and the XSD Location, if you have altered any of the “source” pathfiles and filenames you will need to alter the locations and files here.    If you click on Columns  You will get a screen like this:    Selecting the drop down Menu will display the outputs available from the xml.  You do not need to select an output at this point, just click OK, if you have made any changes, or Cancel if you have not. |

* + - 1. *The Data Flow Transformation Tool – Sort*

|  |
| --- |
| There are four “Sorts” in the example task. The sorts are in place to:   1. Select the data item we need to load into the database table 2. Match that item to the other items in the dataset   In this case we start with the “cudid” and we want to add the applicant data and the application data from the xml.    We select the items “cudid”, “applicant” and “Application” from the source and sort each of them on their IDs. They are then ready for the “Merge Join” tool.  The final “Sort” is named “SortOnApplicationId”. This takes the output from the “Merge Join cudid+application” and re-sorts it ready for the next “Merge Join”  ***Note:*** *Each dataset, cudSubject, Student, Enrolment etc. has its own ID e.g. EnrolmentId, StudentId. These IDs are not persistent and change each day. These IDs are not output to the database in order to reduce confusion. The “Merge Join” section below explains how this is done.*  The Package is set up to match everybody in the xml on their “cudid” as this is unique and persistent. This is the ID which is loaded to nearly all the data bases as the Primary Key (PK) |

* + - 1. *The Data Flow Transformation Tool – Merge Join*

|  |
| --- |
| There are two “Merge Joins” in the example task. The Merge Joins are in place to:   1. join two data items/datasets together on a single ID 2. output selected data in a single row.   The first Merge Join in this task matches the “cudid” and “applicant” on the “cudSubject\_Id”    It then outputs the cudid, ApplicantCode, ApplicantNumber and applicant\_Id. You will note that the Merge Join, does not output the “cudSubject\_Id”, as noted in 3.2.3.2 above, this ID is not persistent and has been removed to reduce confusion. |

*The Data Flow Transformation Tool – Merge Join (continued)*

|  |
| --- |
| After this data has been re-sorted on ApplicantId, it is then “Merge Joined” in +Application:    Again you will note, in both the “cudid+applicant” and “+Application” merge Joins, the non-persistent IDs are removed i.e. the cudSubject\_Id and the applicant\_Id. And the primary ID is the cudid.  This Merge Join uses a “Left Outer Join” type. Which means all the data from the “left” input (the SortOnApplicationId) are selected, but only data which matches that data are selected form the right input (the Application)  This data are sorted on the “cudid” and then output to the Data Conversion tool. |

* + - 1. *- The Data Flow Transformation Tool – Data Conversion*

|  |
| --- |
| The CUD\_Receiver Database has been designed to be as simple as possible, for this reason all fields have been set to nvarchar (255).  As a result, some of the data types output form the xml need to be converted in order to load them to the CUD\_Receiver database.  In the example task:  The following items were converted from their original type to Unicode string [DT\_WSTR], length 255.  ApplicantCode, ApplicantNumber, OSSCourseCd StartDt, EndDt, ApplyAttendModeCd    All of the other data items are output in Unicode string [DT\_WSTR], length 255, and therefore need no conversion.  The data are then ready to be output to the target table in CUD\_receiver. |

* + - 1. *The Data Flow Destination Tool – SQL Server Destination*

|  |
| --- |
| The SQL Server Destination for this Task and all the other tasks in the package is CUD\_Receiver  You will need to alter this destination and the destination table in each of the Data Flow tasks in this package.  The instructions on how to alter this are in section 3.3 of this document. |

* 1. **– Selecting the Correct SQL Server Destination**

|  |
| --- |
| From the Data Flow Screen:   1. Right Click on the Connection managers Pane at the bottom of the screen.      1. Select NEW OLE DB Connection 2. Select New… 3. Select/Enter the server name:      1. Select the CUD\_Receiver database      1. Click Test Connection. 2. If this Test Connection has succeeded Click OK 3. Click OK again.   The connection should now appear in the Connection Managers pane. |

* 1. **Selecting and Mapping to the Correct Tables**

|  |
| --- |
| From the Data Flow Screen:   1. Select the first Data Flow task in the drop down menu 2. When it has populated, Right Click on the SQL Destination tool 3. Select Edit 4. In the SQL Destination Editor, under Connection Manager, click on “New…”      1. In The Configure OLE DB Connection Manager select the Correct Destination database 2. Click OK 3. Back in the SQL Destination Editor, under “Use a table or view”, click on drop down option.      1. Select the correct table to load to. 2. In the left hand navigation pane click on “Mappings” 3. It is here you can see whether the mappings have been maintained and make changes in the “Input Column”.     In this example you can see that the “ApplicantCode” has an input of “Data Conversion.ApplicantCode” which is the output from the data conversion tool noted in section 3.2.3.4 of this document.   1. Click on OK 2. Repeat this process for each of the tasks in the package, until all of the data flow tasks are pointing at the correct server, database and table. 3. You can test run the data into the database after this by executing the package. |

* 1. **Importing the package to SQL Server**

|  |
| --- |
| In order to set up the Job which will run the SSIS package on a daily basis, and to protect the package from untested/unwanted alterations, you need to import it to the SQL Server Integration Services.   1. Open SQL Server Management Studio 2. On the “Connect to Server pane, under “Server type:” select “Integration Services” 3. Select the server 4. In the navigation pane on the left expand “Stored Packages”      1. Right Click on “MSDB” 2. Select “Import Package” 3. In the Import Package pane under “Package Location” select File System      1. Under “Package path:” navigate to the SSIS Package location 2. Give the Package name – recommended the same as the SSIS package name 3. Select the “Protection level:” – I have it set as shown 4. Click on OK 5. Expand the folder you have imported to. 6. The CUDXMLtoCUDTables package should now appear in the list. |

1. **Setting up SQL Server Jobs**
   1. **The “Truncate CUD\_Receiver Tables” Job**

|  |
| --- |
| If you do not wish for the data in the tables to be removed prior to loading of fresh data you can ignore this step.   1. Open SQL Server Management Studio and connect to the server 2. Expand the “SQL Server Agent” icon      1. Then right click on the “Jobs” folder 2. Select “New Job.” 3. Insert the name for your job e.g. Truncate CUD\_Receiver tables 4. Select the “Owner” if different.   *You can select a category & enter a description but it’s not necessary*     1. Select Steps 2. Click on “New…” 3. Enter a Step name & select the “Database” from the drop down menu      1. You can write your version of the truncate SQL into the command pane or copy the complete SQL script from the “Truncate CUD\_Receiver Tables.sql” file in the “CUD XML to SQL and SSIS solution” zip folder you have already downloaded. 2. Paste the SQL Script into the Command Pane, then Click on OK |

*4.1 The “Truncate CUD\_Receiver Tables” Job (continued)*

|  |
| --- |
| 1. Click on Schedules      1. Click on “New…” 2. Give the Schedule a name *e.g. “Truncate Schedule”* 3. Reset the options as follows *(or to how you wish to set your schedule up)*      1. Click on OK 2. Then Click on OK again   The Job Should now appear in the list of “Jobs” in the navigation pane.    You can test this job by right clicking on it and selecting “Start Job at Step” |

* 1. **The“Run the SSIS Package” Job**

|  |
| --- |
| 1. Open SQL Server Management Studio and connect to the server 2. Expand the “SQL Server Agent” icon      1. Then right click on the “Jobs” folder 2. Select “New Job.” 3. Insert the name for your job e.g. “Load CUD Data” 4. Select the “Owner” if different.   *You can select a category & enter a description but it’s not necessary*     1. Select Steps 2. Click on “New…” 3. Enter a Step name e.g. “Run Package”      1. From the “Type” drop down select “SQL Server Integration Services Package” 2. From the “Server” select the location of the SSIS package you imported in section 3.5 of this document. 3. Navigate to the location of the Package 4. Click OK |

*4.1 The “Run the SSIS Package” Job (continued)*

|  |
| --- |
| 1. Click on Schedules      1. Click on “New…” 2. Give the Schedule a name *e.g. “Run Package”* 3. Reset the options as follows *(or to how you wish to set your schedule up)*     *Note: I have set the “Load” Job 2 hours after the “Truncate” job.*   1. Click on OK 2. Then Click on OK again   The Job Should now appear in the list of “Jobs” in the navigation pane.    You can test this job by right clicking on it and selecting “Start Job at Step” |