Creating a new Product (Line of Business)

Contents

[Introduction 3](#_Toc161296289)

[Set up Project Folder 3](#_Toc161296290)

[Check Product Type ID / Backup and Restore Databases 3](#_Toc161296291)

[Create Screens in Screen Designer 3](#_Toc161296292)

[Create the Product 3](#_Toc161296293)

[Create the main screen 4](#_Toc161296294)

[Creating additional screens 8](#_Toc161296295)

[Creating child screens 8](#_Toc161296296)

[Release to Dev TCAS 12](#_Toc161296297)

[Build the Screen Designer Project 12](#_Toc161296298)

[Create Product / Line of Business in Dev Transactor\_Live database 13](#_Toc161296299)

[Run the Build Script in Dev Transactor\_Live database 13](#_Toc161296300)

[Create Build Script Part 2 13](#_Toc161296301)

[Release TCAS Screens and Web service Stylesheets 15](#_Toc161296302)

[Release to UAT TCAS 17](#_Toc161296303)

[Release to Test Constructaquote.com web site 18](#_Toc161296304)

[TGSL LOB Scheme Builder steps 18](#_Toc161296305)

[Product database steps 19](#_Toc161296306)

[Synonyms 19](#_Toc161296307)

[Answer set function 20](#_Toc161296308)

[Copy files 21](#_Toc161296309)

[Testing UAT web site 21](#_Toc161296310)

[Product database fixes for UAT web site 22](#_Toc161296311)

[Parent Question IDs 22](#_Toc161296312)

[Assumptions defaults 23](#_Toc161296313)

[Calculated fields 23](#_Toc161296314)

[Question enablement 23](#_Toc161296315)

[Release to Test XBroker website 23](#_Toc161296316)

[TGSL LOB Scheme Builder steps 23](#_Toc161296317)

[Copy files 25](#_Toc161296318)

[Testing UAT web site 25](#_Toc161296319)

[Generate Miscellaneous LOB SQL Scripts 25](#_Toc161296320)

[TGSL LOB Scheme Builder steps 25](#_Toc161296321)

[Update uspPremiumCoverListSelect Procedure 27](#_Toc161296322)

[Check in to Git 27](#_Toc161296323)

[Release to Live TCAS 27](#_Toc161296324)

[Release to Live Constructaquote.com web site 28](#_Toc161296325)

[Release to Live XBroker web site 29](#_Toc161296326)

[Run Miscellaneous LOB SQL Scripts 29](#_Toc161296327)

[Troubleshooting 30](#_Toc161296328)

[Screen Designer Errors on Release Project 30](#_Toc161296329)

[“Screen Designer was unable to find a required file:C:\Program Files (x86)\Wix\candle.exe” on Release Project 30](#_Toc161296330)

[“Error in TES\_Screen\_Deployment.clsMSI” on Release Project 30](#_Toc161296331)

[Screen Designer freezes on Release Project 30](#_Toc161296332)

[TCAS Errors 31](#_Toc161296333)

[Error Loading List View 31](#_Toc161296334)

[Quote journey 31](#_Toc161296335)

[Appendices 32](#_Toc161296336)

[Appendix I: New list table script 32](#_Toc161296337)

[Appendix II: Files produced by Screen Designer 32](#_Toc161296338)

[Appenidx III: Product and Product links script 34](#_Toc161296339)

[Appendix IV: Mandatory and default fields script 35](#_Toc161296340)

[Appendix V: Product database question set query 39](#_Toc161296341)

[Appendix VI: Product database calculated questions query 40](#_Toc161296342)

[Appendix VII: Product database calculated question insert example 40](#_Toc161296343)

[Appendix VIII: Product database enablement criteria query 41](#_Toc161296344)

[Appendix IX: Product database question enablement insert example 41](#_Toc161296345)

# Introduction

A new Product (also called Line of Business) specification should be received from Operations, defining the risk question set screens that will be required, their order, and the questions on each screen.

These new question set screens are created using the TGSL Screen Designer tool which produces files and scripts to release the new Product to the TGSL application and database.

An in-house tool called the TGSL LOB Scheme Builder processes these files and creates additional output for the Constructaquote website, database objects for Scheme creation, etc.

This document describes the steps to follow to create a new Product and should be completed in order as far as possible.

# Set up Project Folder

This assumes you are using Git and have the TGSL repository cloned locally.

1. Create a new project folder in the following location:  
   …GIT\TGSL\LineOfBusiness\Projects  
   For the new folder name use ‘M’ followed by a maximum of seven characters. This will be the short code for the new Product used in the TGSL database. For example ‘MLIAB’ is Tradesman Liability and ‘MPROIND’ is Professional Indemnity.
2. Add the following sub-folders:
   * Documents
   * Schemes
   * SQL
   * Calculators
3. Save a copy of the spec from Operations into the Documents sub-folder.

# Check Product Type ID / Backup and Restore Databases

Each Product is assigned a Product Type ID in TGSL. It is important that this ID is the same between Dev, UAT and Live databases because the ID gets hard coded into the XML files that are produced by Screen Designer. The new screens will not work when released from Dev to UAT or from UAT to Live if the ID is different.

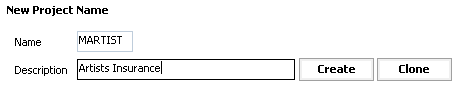
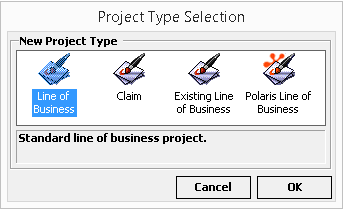
It is best practise to restore the Dev and UAT Transactor\_Live databases, and UAT Product database, from a backup of Live before beginning any new project. This is especially important if the IDs are out of sync. To check the current highest ID that is in use, launch SQL Server management Studio and connect to **MHGSQL01\TGSLDEV**. Run the following query in the Transactor\_Live database:

SELECT MAX([Product\_Type\_ID]) FROM [dbo].[LIST\_PRODUCT\_TYPE] WHERE [Product\_Type\_ID] <> 999

Then connect to both **MHGSQL01\TGSLTEST** and **MHGSQL01\TGSL** and repeat the query.

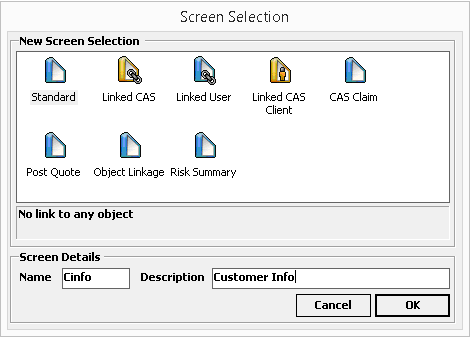
# Create Screens in Screen Designer

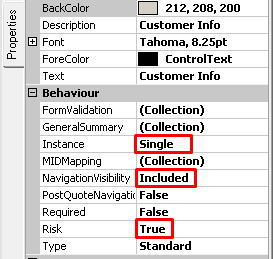
## Create the Product

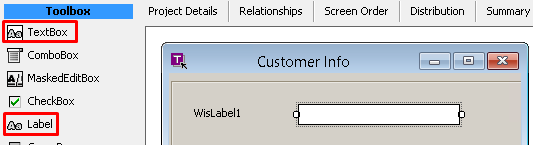
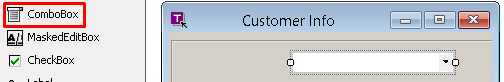
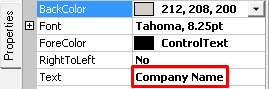
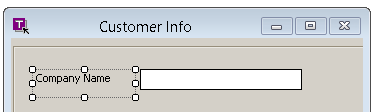
1. Start a Remote Desktop session and connect to the TGSL Development server **MHGTGSL01Dev**
2. Click the start button and search for the TES Screen Designer application, then open it:  
   
3. Screen Designer will open to the New Project screen. Enter your chosen new product code beginning with ‘M’ (see the [Set up Project Folder](#_Set_up_Project) section above) as the Name, and a full Description, for example:  
   
4. Click **Create**
5. From the Project Type Selection window that appears, choose **Line of Business** and click **OK**:  
   

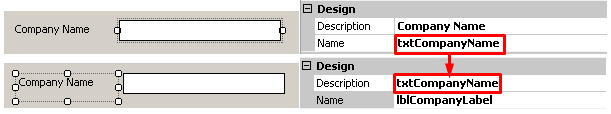
|  |
| --- |
| **WARNING:** The project will now appear in the Existing Project list as shown below but no files have actually been created yet. DO NOT click on this project link yet or it will return an error that the file cannot be found, and you will have to start again with a new product code! |

## Create the main screen

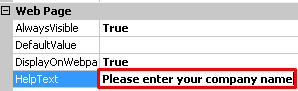
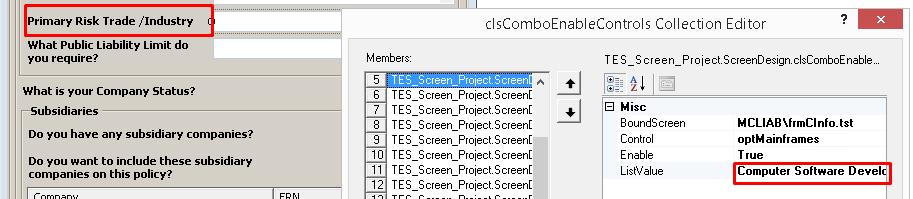
1. Click on **File > New > Screen** or click the new screen shortcut icon on the toolbar:  
   
2. Select **Standard** from the Screen Selection window that appears and fill in the Name and Description for the first screen defined in the specification. Then click **OK**:  
   
3. A new blank screen will appear. Click on the **Properties** tab at the right hand side of the screen and set the following three properties that are initially blank, to prevent errors when building the project:

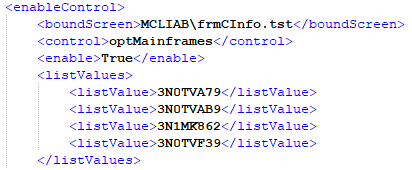
* **Instance**: Set to ‘Single’ so this screen only appears once per quote
* **NavigationVisibility**: Set to ‘Included’
* **Risk**:Set to ‘True’ as this is our main risk (quote) screen  
  

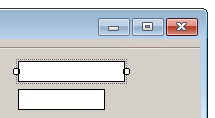
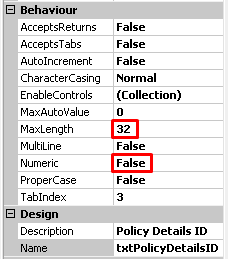
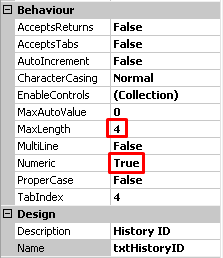
1. Double click control types in the Toolbox panel on the left of the screen and then click to place controls onto the new screen to create the question and answer fields required according to the specification.
   1. For example add a **Label** and a **Text Box**:  
      
   2. Or for a drop-down add a **ComboBox** and choose the relevant database table that contains the drop-down values in the **ListTable** property in the Properties tab at the right of the screen:  
        
        
        
      Note 1: If a new list table needs to be created, this will need to be done first in order to select it in Screen Designer. See the example script in [Appendix I: New list table script](#_Appendix_I:_New) for creating a table, product link table, and selection view. This will also require creating the Product ID first. For this see the [Create Database LOB](#_Create_Database_LOB) section below.  
        
      Note 2: The available values in drop-downs can be restricted by populating …LINK tables. For example, to restrict a LIST\_MH\_TRADE drop-down, insert rows into the LIST\_MH\_TRADE\_LINK table, linking the Trade IDs to the Agent and Product. There is a prompt for this step in the [Create Build Script Part 2](#_Create_Build_Script) section below.
2. Set the text for labels by changing the **Text** property in the Properties tab at the right of the screen. These will then be reflected on the new screen:  
    
3. Change the name on the data entry control to something friendlier than the default and then copy this name into the **Description** field of the **label**:



This is done because there is no inherent link between labels and fields in Screen Designer. The LOB Scheme Builder will use the label description to create the link and get the question text from the label for the Constructaquote web site.

1. Be sure to set the **HelpText** property in the Web Page section of the Properties tab as this will be used to create the text for the Constructaquote web site:  
   
2. If applicable, set the enablement criteria, i.e. if a question / field becomes available or unavailable based on the value of another field. To do this:
   1. Click on the ellipsis button of the **EnableControls** property in the Properties tab for the field containing the value that will drive the enablement:  
      
   2. Click **Add** and fill in the **BoundScreen** and **Control** (i.e. the screen and field) that is to be enabled or disabled, set whether **Enabled** is true or false, and enter the **ListValue** that will drive the enablement. For example, on the MCLIAB Product the Primary Risk Trade value enables the Mainframes question only for relevant trades:  
      

Note, while it is possible to enter multiple enablement criteria on the same field, if there are lots of rules it may be easier to edit them XML in the .tst (screen) file, e.g. for the example above:  
L:\Dev\TCAS\Projects\MCLIAB\frmCInfo.tst  


1. Expand the edge of the new window to the right and create two “hidden” text box fields, one of them a 32 character non-numeric field for PolicyDetailsID and the other a four digit numeric field for HistoryID:  
     

These fields do not need labels. One they are placed, drag the edge of the new window back to the left so that the text boxes are hidden.  
This is done because TGSL does not automatically pass POLICY\_DETAILS\_ID and HISTORY\_ID fields to the WPD when quoting. We will update the stored procedures that Screen Designer produces later to set these fields to the same values as POLICY\_DETAILS\_ID and HISTORY\_ID. They can then be passed to the WPD and referenced in Scheme calculator procedures.

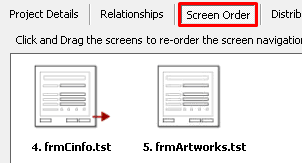
1. To save progress so far, click the save icon on the toolbar, or click **Build > Build Project**, which is equivalent to saving:  
   

|  |
| --- |
| For full instructions on all the differenct controls available in Screen Designer see the official Transactor Screen Builder User Guide PDF. This is available from the Process Documentation board on Monday.com:  <https://constructaquote.monday.com/boards/671792069/pulses/2866214929>    And also from the Git Documentation repository:  \GIT\Documentation\Product |

## Creating additional screens

Most question sets will require multiple screens that are filled in one after the other when quoting in TGSL. To add a second screen, simply click the new screen icon again and follow the same process as above. Be sure to set the **Instance** and **NavigationVisibility** properties otherwise the new screen will not appear in the next step.

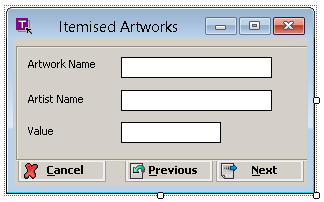
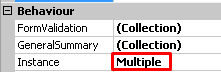
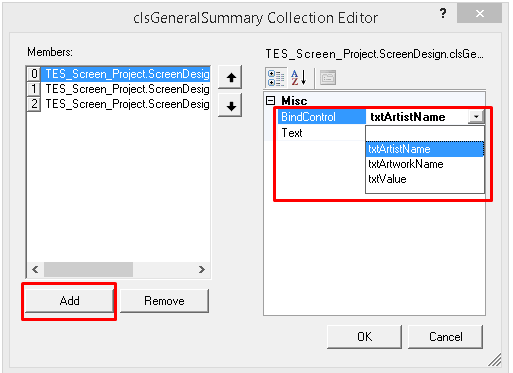
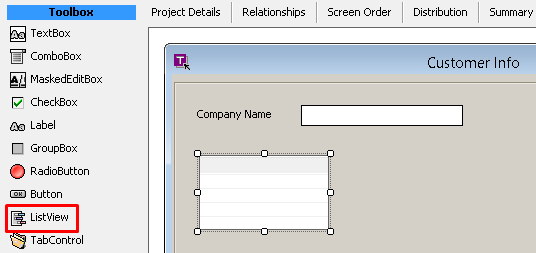
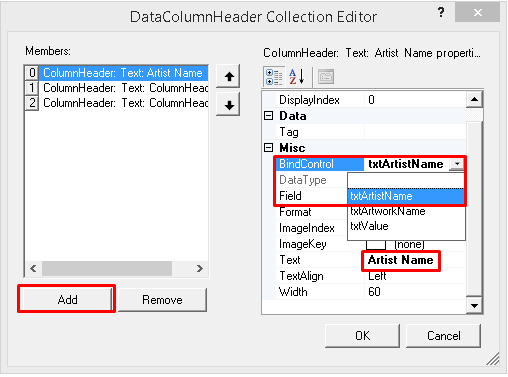
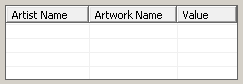
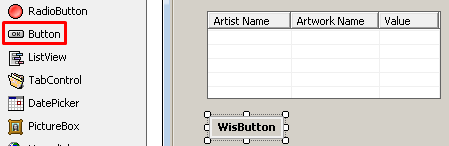
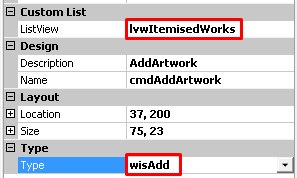
To set the order that the screens are presented when quoting, click on the **Screen Order** tab. Click the refresh button on the right if the new screen is not yet visible on this tab. Then simply drag and drop to arrange the screens in the desired order:

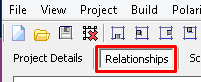


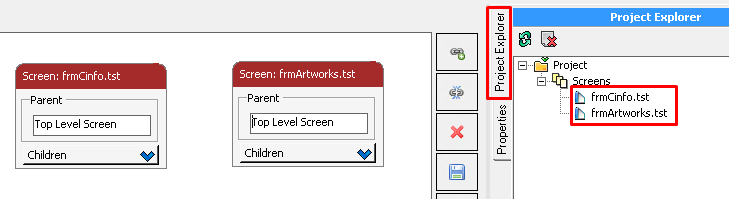
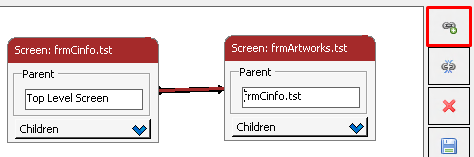
After making any changes, click the save button on the right.

## Creating child screens

In some cases a sub-screen will be invoked by clicking a button on another screen. These are called child screens and are commonly used in our products for adding lists of partners and principals at a company, or lists of vehicles.

1. Follow the steps detailed above to create a new screen that will function as the child screen. For example:  
   
2. Set the **Instance** property at the screen level to Multiple because multiple values can be entered for the same quote:  
   
3. Click on the ellipsis button for the **GeneralSummary** property under Behaviour at the screen level to create a summary view for the ListView. This step has to be completed to avoid an error when releasing the project:  
   
4. In the General Summary Collection Editor window that opens, click the **Add** button and select the first column from the child screen from the **BindControl** drop-down. Also enter the column name in the **Text**:  
   
5. Repeat this step for the remaining columns on the child screen.
6. Return to the main or parent screen and add a **ListView** control from the Toolbox. This will create a blank list box as shown:  
   
7. In the properties tab on the right of the screen, set the **BoundScreen** property for the ListView to the name of the child screen created in step 1:  
   
8. Then click the ellipsis button on the **Columns** property under Behaviour:  
   
9. In the collection Editor window that opens, click the **Add** button and select the first column from the child screen from the **BindControl** drop-down. Also enter the column name in the **Text** field as shown:  
   
10. Repeat this step for the remaining columns on the child screen.
11. Click **OK** to return to the main screen. The ListView box will now show the columns which can be resized accordingly:  
    
12. Add, Edit and Remove buttons should be added below the ListView box to invoke the child screen and allow editing of contents. Click the **Button** control in the Toolbox and add a button below the ListView:  
    
13. In the Properties tab, change the **Type** to **wisAdd** and then select the ListView that the button will invoke:  
    
14. Repeat this process to add a **wisEdit** and **wisRemove** button, editing the settings accordingly:  
    
15. Click on the **Relationships** tab near the top of the window:



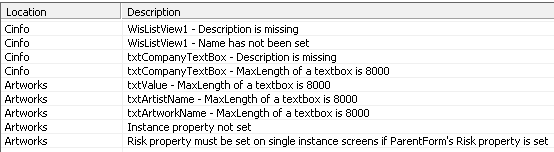
1. Select the Project Explorer tab on the right. Double-click the main screen and then single click in the white space to add it to the Relationships diagram (drag and drop doesn’t work). Do the same for the child screen:  
   
2. Click the **Set Relationships** button and then click the main screen following by the child screen. This will add an arrow to indicate the relationship:  
   
3. Click the **Save Layout** button on the right. This step ensures that the database table for the child screen will be created with a primary key that relates to the main table instead of having POLICY\_DETAILS\_ID and HITORY\_ID columns. This can cause errors when quoting in TGSL if not set correctly (see [Error Loading List View](#_Error_Loading_List) in the Troubleshooting section below).

NOTE: Relationships can also be set by editing the <parentForm> XML attribute in the .tpj (screen designer project) file that gets saved in the following location:

L:\Dev\TCAS\Projects\<project code>

# Release to Dev TCAS

## Build the Screen Designer Project

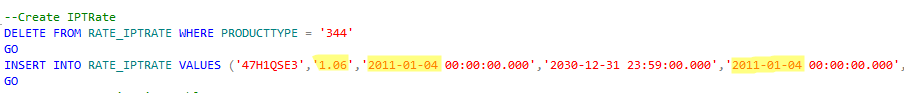
1. Ensure you have saved all changes by clicking the save icon on the toolbar, or clicking **Build > Build Project**.
2. Click on **Build > Release Project** and select **Yes** from the confirmation dialogue that appears.
3. It is likely that the “Errors Encountered in Project Build” message will appear. If so:
   1. Click **OK**. The error will be displayed at the bottom of the screen. For example:  
      
   2. These errors should be fairly self-explanatory. Work through them and make changes in the Properties tab for the affected fields.
   3. When all errors have been addressed, try the Release Project step again.
4. Wait for the project to compile successfully. This may take several minutes. In case of errors or the program freezing, see [Sreen Designer Errors on Release Project](#_Screen_Designer_Errors) in the Troubleshooting section below. Otherwise click **OK**.
5. Click on **Build > Release Website** and select **Yes** from the confirmation dialogue that appears.
6. Wait for the project to compile successfully. This may take several minutes. In case of errors or the program freezing, see [Sreen Designer Errors on Release Project](#_Screen_Designer_Errors) in the Troubleshooting section below. Otherwise click **OK**.

|  |
| --- |
| These steps will save a number of files that will be released in the following steps for TCAS, WebService (Constructaquote web site) and Website (XBROKER). The files are saved in project folder L:\Dev\TCAS\Projects\<project code>  For a list of the files produced see [Appendix II: Files produced by Screen Designer](#_Appendix_II:_Files) |

## Create Product / Line of Business in Dev Transactor\_Live database

1. Open SQL Server Management Studio and connect to **MHGSQL01\TGSLDEV**.
2. Copy the example SQL script from [Appenidx III: Product and Product links script](#_Appenidx_III:_Product), making changes where indicated.
3. Save the script as L:\Dev\TCAS\Projects\<project code>\<project code> Product.sql
4. Run the script in the Transactor\_Live database.

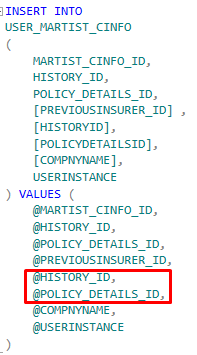
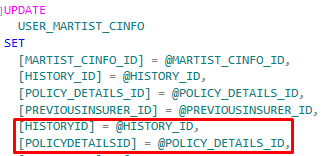
## Run the Build Script in Dev Transactor\_Live database

1. Still within SQL Server Management Studio, connected to **MHGSQL01\TGSLDEV**, open the script Build.sql that the Screen designer release will have saved as L:\Dev\TCAS\Projects\<project code>\<project code> Build.sql  
   This script creates NAV… navigation tables that define the screens and their fields, and USER… data entry tables that will store risk information when quoting.
2. Comment out or delete the first three INSERT statements (for SYSTEM\_LIST\_PRODUCTTYPE, SYSTEM\_LIST\_PRODUCTTYPE\_ALIAS and SYSTEM\_LIST\_PRODUCTTYPE\_VERSION) as these inserts have already been done in the Product script using a dynamic Product Type ID. However, it is best to check that Screen Designer has picked up the same Product Type ID that was created in the Product script by selecting from the SYSTEM\_LIST\_PRODUCTTYPE table. If the ID is different, Screen Designer will likely have output XML files in the sub folders under L:\Dev\TCAS\Projects\<project code>\WebService with the wrong IDs in their file names. These will need to be corrected.
3. Edit the insert into the RATE\_IPTRATE table. The script defaults to 1.06 (i.e. 6%) which was the IPT rate from 4th January 2011 as shown:  
   

Change this to 1.12 (i.e. 12%) and the highlighted dates to 1st June 2017, which is when the rate increased to 12%. Note, this is correct as of 2024. The Insurance Premium Tax is set by the government and the current rate can be confirmed by performing an internet search. This may also be corrected by a future upgrade of TGSL and Screen Designer.

1. Save the changes to the script.
2. Run the script in the Transactor\_Live database.

## Create Build Script Part 2

1. Still within SQL Server Management Studio, open a new query window to create a secondary script for customisations and corrections where Screen Designer hasn’t created things exactly how we want them.
2. Save the new script as L:\Dev\TCAS\Projects\<project code>\<project code> Build part 2.sql
3. Copy the CREATE PROCEDURE statements from the main build script for USER\_SAVE\_<projectcode>\_<mainscreen> and USER\_UPDATE\_<projectcode>\_<mainscreen> where <mainscreen> is the name of the main risk screen in Screen Designer that contains the hidden PolicyDetailsID and HistoryID text boxes, e.g the CInfo screen.
4. Paste the statements into the new script and change them from ‘CREATE PROCEDURE’ to ‘ALTER PROCEDURE’.
5. Change the USER\_SAVE\_<projectcode>\_<mainscreen> procedure so that it inserts POLICYDETAILSID from @POLICY\_DETAILS\_ID instead of @POLICYDETAILSID and inserts HISTORYID from @HISTORY\_ID instead of @HISTORYID, e.g.  
   
6. Change the USER\_UPDATE\_<projectcode>\_<mainscreen> procedure so that it sets POLICYDETAILSID from @POLICY\_DETAILS\_ID instead of @POLICYDETAILSID and sets HISTORYID from @HISTORY\_ID instead of @HISTORYID, e.g.:  
   
7. Run the following three SELECT statements to view the order that the screens will appear in TCAS: SELECT \* FROM [Nav\_Forms\_<product code>\_NEW] ORDER BY [SEQUENCE\_044]  
   SELECT \* FROM [Nav\_Forms\_<product code>\_ADD] ORDER BY [SEQUENCE\_044]

SELECT \* FROM [Nav\_Forms\_<product code>\_IND] ORDER BY [SEQUENCE\_044]

1. Create three UPDATE statements in the Build part 2 script to set the SEQUENCE\_044 field to the desired order in each of the three tables. Although screen order was set in Screen Designer, there are standard screens (Contact and Marketing Details, Target Premium and Cross Sales) that should be moved from the start to the end of the sequence). Rows where ENABLED\_044 = 0 can be ignored as these screens are not enabled. Note that the …IND table contains more rows as child tables appear as separate records. An example update statement is as follows:

UPDATE

[dbo].[Nav\_Forms\_MARTIST\_NEW]

SET

[Sequence\_044] = CASE [NAV\_MCYBER2\_NEW\_044]

WHEN 'Client Details' THEN 1

WHEN 'Customer Info' THEN 2

WHEN 'Target Premium' THEN 3

WHEN 'Cross Sales' THEN 4

WHEN 'Contact and Marketing Details' THEN 5

WHEN 'Artists Insurance Summary' THEN 6

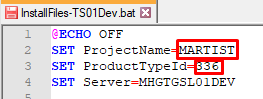
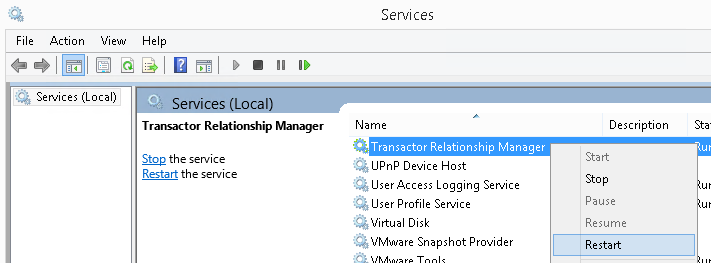
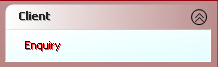
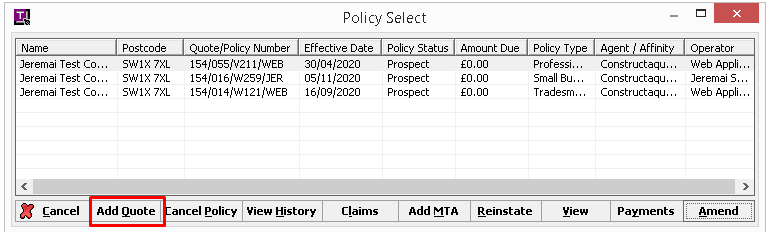
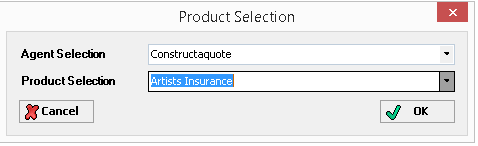
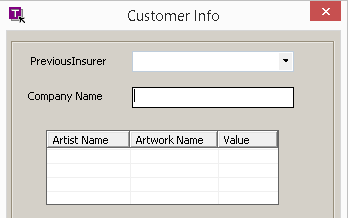
ELSE 0 -- these are not enabled

END

1. Screen Designer does not provide any functionality to set fields as mandatory as there are tools for non-technical users to edit these settings. In order to create a releasable script, mandatory fields and their validation messages can be set by updating the MANDATORY\_044 and MANDATORY\_MESSAGE\_044 fields in the Nav\_Fields\_<product code>\_NEW, …ADD and …IND tables. Default values can also be set in the DEFAULT\_VALUE\_044 field whilst updating these tables. See [Appendix IV: Mandatory and default fields script](#_Appendix_IV:_Mandatory) for an example script and save any updates in the Build part 2 script.
2. The total calculated premium can be broken down into different policy sections or coverages. For example, property policies can have buildings and contents cover. Add an INSERT statement in the Build part 2 script to insert any new sections required into the LIST\_POLICY\_SECTION section. These should be defined in the specification document. The POLICY\_SECTION\_ID should consist of eight characters including a suffix of PREM e.g. BUSUPREM.
3. Add in any list table restrictions required. For example, it is likely that only certain trades should be available to select in the Trade drop-down. To restrict the drop-down, create an INSERT script to add rows to the LIST\_MH\_TRADE\_LINK table, linking the Trade IDs to the AGENT\_ID (Constructaquote, Constructaquote.com, XBroker, etc.) and the new PRODUCT\_ID.
4. Run the Build part 2 script in the Transactor\_Live database.

## Release TCAS Screens and Web service Stylesheets

This assumes you are using Git and have the TGSL repository cloned locally.

1. Locate the batch file GIT\TGSL\LineOfBusiness\Install Project\InstallFiles-Dev.bat but DO NOTdouble-click as this will run it!
2. Right-click on InstallFiles-Dev.bat and choose to edit with Notepad++ or other text editor that you have installed.
3. Update the **ProjectName** and **ProductTypeId** values at the top of the batch script:  
     
   The ProjectName is the code beginning with ‘M’ that has been used throughout the process so far. The ProductTypeId would have been created when the first build script was run. You can find it by selecting from the SYSTEM\_LIST\_PRODUCTTYPE\_ALIAS table in the Transactor\_Live database.
4. Save the changes and close the file.
5. Double click InstallFiles-TS01Dev.bat to run it. For information, this will perform the following actions to install the new project into TCAS:
   * Copies all files from L:\Dev\TCAS\Projects\<product code> to L:\Dev\TCAS\Templates\<product code>
   * Copies all files from L:\Dev\TCAS\Projects\<product code> to L:\Dev\TCAS\Stylesheets\<product code>
   * Copies XML files from L:\Dev\TCAS\Projects\<product code>\WebService and subfolders folder to corresponding folders on C:\inetpub\wwwroot on the Dev TGSL server MHGTGSL01Dev
6. Start or return to your Remote Desktop session for the TGSL Development server **MHGTGSL01Dev**.
7. Close TCAS if it is already running.
8. Click the start button and search for Services, then click to open:  
   
9. Scroll down to the **Transactor Relationship Manager** service, right-click it and choose **Restart**:  
   
10. Click the start button and search for TCAS, then click to open:  
    
11. Log in
12. Perform an enquiry to find a suitable test customer:  
    
13. Click on **Add Quote**:  
    
14. Select an Agent that the new Product has been created under (i.e. XBroker, Constructaquote, or Constructaquote.com, unless changed when creating the Product). The new Product should then be available to select from the Product Selection drop-down:  
    
15. Click **OK**
16. The main new screen should appear:  
    
17. Try filling in the screens and check for any errors. If successful, you should reach the User Summary screen. It will not be possible to quote because no Schemes have been created yet for the new Product.

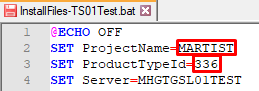
# Release to UAT TCAS

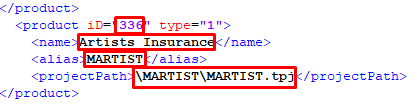
When happy that the Product is ready for UAT:

1. Open SQL Server Management Studio and connect to MHGSQL01\TGSLTEST.
2. Copy the entire folder L:\Dev\TCAS\Projects\<product code> to L:\**Test**\TCAS\Projects\<product code>
3. Run each of the three scripts that were created and run in Dev into the Transactor\_Live database, i.e.:  
   L:\Test\TCAS\Projects\<project code>\<project code> Product.sql

L:\Test\TCAS\Projects\<project code>\<project code> Build.sql

L:\Test\TCAS\Projects\<project code>\<project code> Build part 2.sql

1. Locate the batch file GIT\TGSL\LineOfBusiness\Install Project\InstallFiles-Test.bat but DO NOTdouble-click, as this will run it!
2. Right-click on InstallFiles-Test.bat and choose to edit with Notepad++ or other text editor that you have installed.
3. Update the **ProjectName** and **ProductTypeId** values at the top of the batch script:  
   
4. Double click InstallFiles-Test.bat to run it. For information, this will perform the following actions to install the new project into TCAS:
   1. Copies all files from L:\Test\TCAS\Projects\<product code> to L:\Test\TCAS\Templates\<product code>
   2. Copies all files from L:\Test\TCAS\Projects\<product code> to L:\Test\TCAS\Stylesheets\<product code>
   3. Copies XML files from L:\Test\TCAS\Projects\<product code>\WebService and subfolders folder to corresponding folders on C:\inetpub\wwwroot on the Test TGSL server MHGTGSL01Test
5. Locate the file L:\Test\TCAS\Templates\Screenbuilder.tsb and open with Notepad++ or another text editor.
6. Copy and paste an existing five line product element and change the Product ID attribute to the ProductTypeID for the new Product, and change the name, alias and projectPath child elements accordingly:



If the new ID already exists in this file from a previous test Product that was never released to live, it can be overwritten with the new Product.

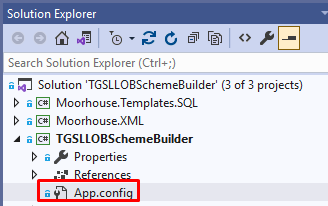
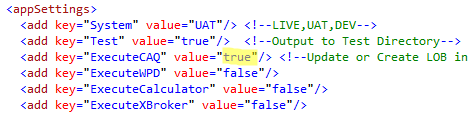
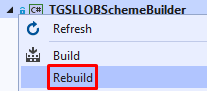
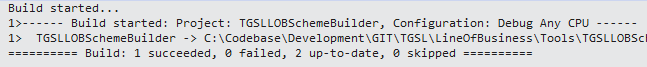
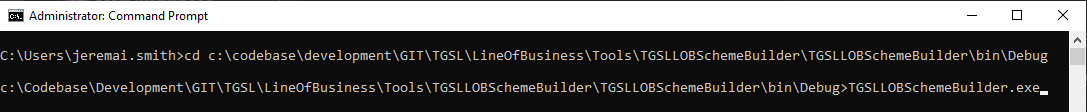
1. Save and close the file
2. Start a Remote Desktop session and connect to the TGSL UAT server **MHGTGSL01Test**
3. Restart the Transactor Relationship Manager service (instructions are the same as for Dev in the [Release TCAS Screens and Web service Stylesheets](#_Release_TCAS_Screens) section above)
4. Check the Product screens in TCAS (instructions are the same as for Dev in the [Release TCAS Screens and Web service Stylesheets](#_Release_TCAS_Screens) section above

# Release to Test Constructaquote.com web site

There is currently no Dev environment for the web site, so these steps apply to Test only.

## TGSL LOB Scheme Builder steps

This assumes you are using Git and have the TGSL repository cloned locally.

1. From your developer desktop, launch Microsoft Visual Studio
2. Open the solution GIT\TGSL\LineOfBusiness\Tools\TGSLLOBSchemeBuilder\ TGSLLOBSchemeBuilder.sln
3. From the Solution Explorer on the right of the screen, double click **App.config** under TGSLLOBSchemeBuilder to open it:  
   
4. Ensure that the **ProductUAT** connection string is set to the MHGSQL01\TGSLTEST server and Product database:  
   
5. Set System to UAT, Test to true, and ensure that **ExecuteCAQ** is the only Execute… setting set to true:  
   
6. Ensure that the **ProjectRootPath** is set to the test Projects location and set **LOBName** and **LineOfBusiness** to the Product code and name for the new Product:  
     
   The majority of the remaining settings only relate to creating a Scheme / calculator, so can be ignored for now.
7. Right-click on TGSLLOBSchemeBuilder in the Solution Explorer and click **Rebuild**:  
   
8. The output window will show if the build succeeded (this may auto hide if not pinned):  
   
9. Launch Command Prompt and change directory to your local path for the following folder: GIT\TGSL\LineOfBusiness\Tools\TGSLLOBSchemeBuilder\TGSLLOBSchemeBuilder\bin\Debug  
   Then type or paste TGSLLOBSchemeBuilder.exe and press Enter to run it:  
   

Note, you can also double-click to run TGSLLOBSchemeBuilder.exe from within File Explorer, but this method does not allow you to read any error messages returned.

1. To check that the Product was correctly released to the Product database open SQL Server Management Studio and connect to **MHGSQL01\TGSLTEST**. Run the following query in the Product database, changing the text value to the name of the new Product:  
   SELECT \* FROM [QuestionSet].[QuestionSet] WHERE [QuestionSetName] = 'Artists Insurance'

## Product database steps

When the build scripts from Screen Designer were run, a set of tables would have been created in the Transactor\_Live database, prefixed with ‘USER\_<product code>’. These tables hold the risk information entered when quoting.

### Synonyms

Synonyms are required in the Product database to refer to each of these tables without prefixing the database name.

In SQL Server Management Studio, connect to **MHGSQL01\TGSLTEST.** For each ‘USER\_<product code>’, create a synonym using the following example syntax but changing the table name:

CREATE SYNONYM [dbo].[USER\_MARTIST\_CINFO] FOR [Transactor\_Live].[dbo].[USER\_MARTIST\_CINFO]

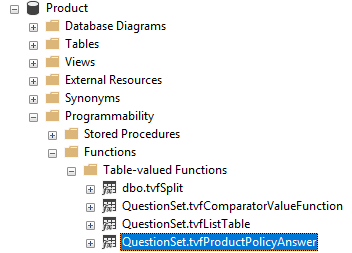
Execute the script and save it in the following folder:

GIT\TGSL\LineOfBusiness\Projects\<product code>\SQL\Product

This script will then form part of the live release.

### Answer set function

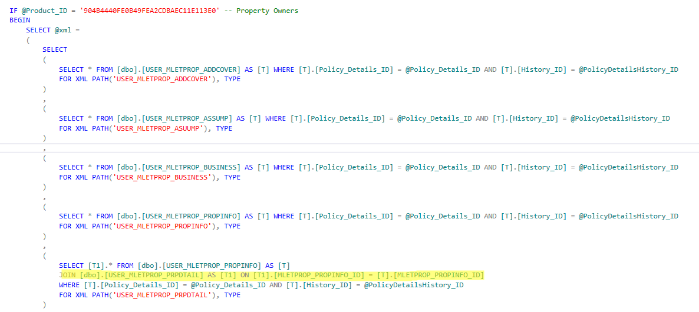
Still within SQL Server Management Studio, connected to **MHGSQL01\TGSLTEST**, locate the function QuestionSet.tvfProductPolicyAnswer in the Product database:



Right-click and select **Modify**. The bulk of the function is made up of IF statements for different PRODUCT\_IDs followed by a section of code between BEGIN and END that selects all columns from all of the USER… tables for that product into an XML variable.

Create a new section for the new PRODUCT\_ID. The PRODUCT\_ID would have been assigned when running the [Product / Line of Business creation script](#_Create_Product_/) and can be retrieved by querying the RM\_PRODUCT table in the Transactor\_Live database.

Within the new section, add a SELECT statement for each of the USER… tables for the new Product. Most of these will select where the Policy\_Details\_ID and History\_ID match the @Policy\_Details\_ID and @PolicyDetailsHistory\_ID parameters. However, in the case of child tables, these need an additional join to the parent table that contains the Policy\_Details\_ID and History\_ID fields as highlighted in the example below:



Execute the script and save it in the following folder:

GIT\TGSL\LineOfBusiness\Projects\<product code>\SQL\Product

This script will then form part of the live release.

## Copy files

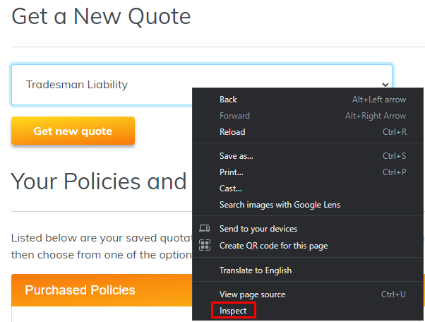
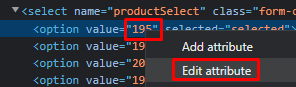
Although the following refers to an XBroker folder, the live server equivalent of this folder is named TGSLWebsiteLive and is used by both CAQ and XBroker (possibly only the stylesheet XSLT that is needed y CAQ).

1. Navigate to the folder below on the UAT web server (this can be done via File Explorer and connecting as moorhouseinternal1 or moorhouseinternal2; contact the web developers if you need the password):  
   \\MHGWEBUAT01\d$\web\_roots\XBrokerQuote\quote
2. Copy the project folder L:\Test\TCAS\Projects\<product code>\Website\<product code> into the location above.
3. Copy the following files into the respective subfolders of the location above:
   * L:\Test\TCAS\Projects\<product code>\website\config\<product code>\_config.xml to .\config
   * L:\Test\TCAS\Projects\<product code>\website\content\<product code>\_ content.xml to

.\ content

* + L:\Test\TCAS\Projects\<product code>\website\content\<product code>\_content.xml to .\App\_Data\
  + L:\Test\TCAS\Projects\<product code> \website\App\_Data\<product code>.xml to .\App\_Data\
  + L:\Test\TCAS\Projects\<product code>\Website\stylesheets\client\read\policy\user\ds<ProductTypeID>Read.xslt to .\stylesheets\client\read\policy\user

## Testing UAT web site

1. Launch Chrome (the inspection instructions below apply specifically to the Chrome browser)
2. Go to <https://quote-test.constructaquote.com/Admin/Admin>
3. Right-click on the Product drop-down under Get a New Quote and click **Inspect**:  
   
4. A panel will open on the right, with the drop-down element highlighted. Click the small triangle on this line:  
   
5. This will show all the available values in the drop-down. We need to temporarily change one of these to the new ProductTypeId in order to test it.
6. Right-click on any ProductTypeId and then elect **Edit attribute**:  
   
7. Change the value to the ProductTypeId for the new Product (this would have been created when the first build script was run and can be found by selecting from SYSTEM\_LIST\_PRODUCTTYPE\_ALIAS in the Transactor\_Live database) and hit the **Enter** key when done. Note, you can also edit the Product name text value attribute but this is not necessary.  
   
8. With the edited value selected from the drop-down, click the **Get new quote** button:  
   
9. The newly created question set should appear. Try filling in the screens and check for any errors. There will likely be issues with the format and appearance of the questions that need correcting in the Product database (see next section).
10. You can click the Quote button at the bottom of the page but it will not be possible to quote because no Schemes have been created yet for the new Product.

## Product database fixes for UAT web site

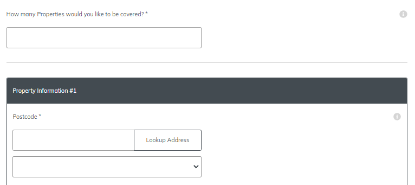
The TGSL LOB Scheme Builder is a work in progress and will likely not have transferred all elements of the screens successfully to the UAT web site question set. The web site page is driven by the configuration inserted into the Product database on **MHGSQL01\TGSLTEST**. A useful query is shown in [Appendix V: Product database question set query](#_Appendix_V:_Product) that can be run (by changing the Product name) to show some of the main fields that make up the question set.

For any fixes required, create a script in the following folder:

GIT\TGSL\LineOfBusiness\Projects\<product code>\SQL\Product

This script will then form part of the live release. The script may include some or all of the following fixes.

### Parent Question IDs

For any child screens that have been created in Screen Designer, the child screen questions should appear in a grey box followed by ADD and DELETE buttons, to allow entry of multiple child records, similar to this example:  


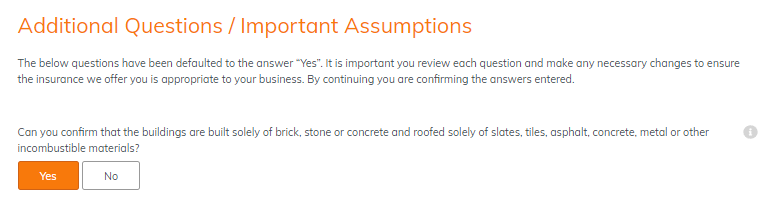


If this is not the case, set the QuestionSet.AgentQuestionDetails.ParentQuestionID field to the QuestionID of the preceding question.

WARNING: Do not hard code Question IDs in this script! Product development is an iterative process and it is likely that the Product database will need to be restored and the TGSL LOB Scheme Builder run again after further changes are made in Screen Designer, at which point new IDs will be assigned. Instead, the script should select the Question ID based on the underlying field name in the QuestionSet.Question table.

### Assumptions defaults

Assumptions should all be defaulted to Yes, as in this example:



This is achieved by setting the QuestionSet.AgentQuestionDetailsAnswerDefaultValueOrID field to ‘3MQT5MC6’ (ID for ‘Agree’ in LIST\_MH\_ASSUMPT table in Transactor\_Live database) for all questions on the assumptions table.

### Calculated fields

Calculated field logic is held in the QuestionSet.CalculatedQuestionOperation table, for example where we calculate the number of years since the business was established. A useful query to show how calculated questions are built can be found in [Appendix VI: Product database calculated questions query](#_Appendix_VI:_Product).

To insert a new calculation, the stored procedure **QuestionSet.uspCalculatedQuestionOperationInsert** should be used.

Since the values are worked out automatically, calculated fields do not need to be presented to the client for data entry. Therefore, check that the QuestionSet.AgentQuestionDetails.Enabled, Visible, and Mandatory fields are all set to zero.

An example insert is shown in [Appendix VII: Product database calculated question insert example](#_Appendix_VII:_Product).

### Question enablement

The logic to enable questions based on the answers to other questions is held in the QuestionSet.EnablementCriteria table. A useful query to show how enablement criteria are built can be found in [Appendix VIII: Product database enablement criteria query](#_Appendix_VIII:_Product).

To insert a new enablement rule, the stored procedure **QuestionSet.uspEnablementInsert** should be used.

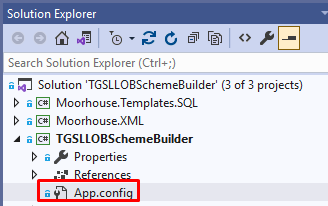
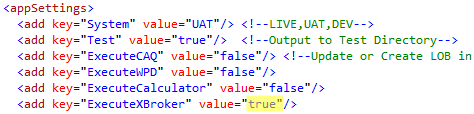
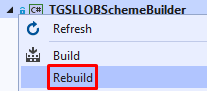
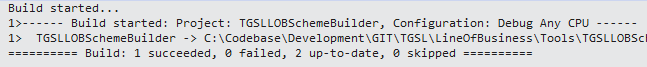
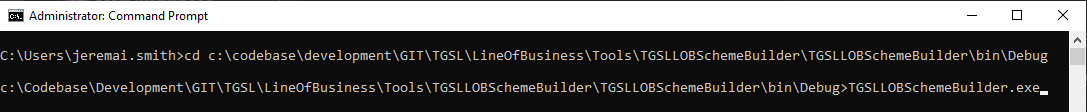
Note that QuestionSet.AgentQuestionDetails.Visible must be set to 1 for enabled fields.

An example insert is shown in [Appendix IX: Product database question enablement insert example](#_Appendix_IX:_Product).

# Release to Test XBroker website

## TGSL LOB Scheme Builder steps

This assumes you are using Git and have the TGSL repository cloned locally.

1. From your developer desktop, launch Microsoft Visual Studio
2. Open the solution GIT\TGSL\LineOfBusiness\Tools\TGSLLOBSchemeBuilder\ TGSLLOBSchemeBuilder.sln
3. From the Solution Explorer on the right of the screen, double click **App.config** under TGSLLOBSchemeBuilder to open it:  
   
4. Set **System** to UAT, **Test** to true, and ensure that **ExecuteXBroker** is the only Execute… setting set to true:  
   
5. Ensure that the **ProjectRootPath** is set to the test Projects location and set **LOBName** and **LineOfBusiness** to the Product code and name for the new Product:  
     
   The majority of the remaining settings only relate to creating a Scheme / calculator, so can be ignored for now.
6. Right-click on TGSLLOBSchemeBuilder in the Solution Explorer and click **Rebuild**:  
   
7. The output window will show if the build succeeded (this may auto hide if not pinned):  
   
8. Launch Command Prompt and change directory to your local path for the following folder: GIT\TGSL\LineOfBusiness\Tools\TGSLLOBSchemeBuilder\TGSLLOBSchemeBuilder\bin\Debug  
   Then type or paste TGSLLOBSchemeBuilder.exe and press Enter to run it:  
   

Note, you can also double-click to run TGSLLOBSchemeBuilder.exe from within File Explorer, but this method does not allow you to read any error messages returned.

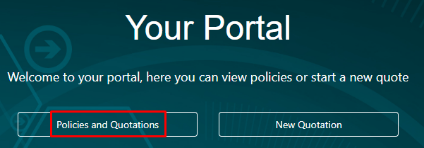
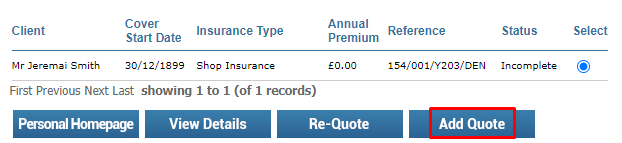
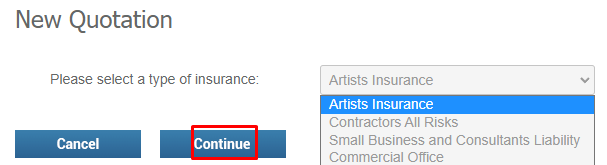
## Copy files

If the steps for releasing to the test Constructaquote.com web site have been followed, the files should already have been copied. If not, follow the instructions in the [Copy files](#_Copy_files) section above.

## Testing UAT web site

You should be able to go straight to a quote page by changing the product code in the following URL:  
http://xbrokerquote.test.moorhouseinsurance.co.uk/quote/martist/start.aspx

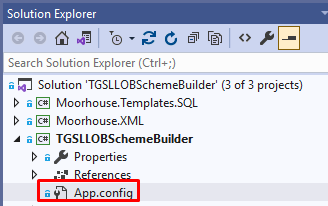
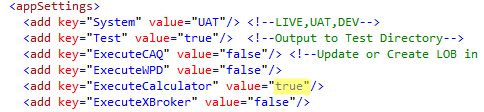
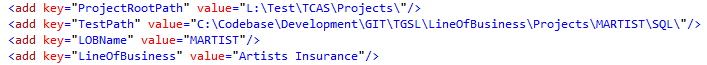
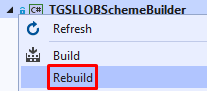
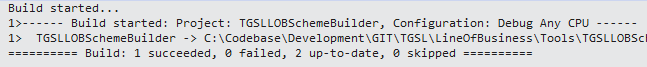
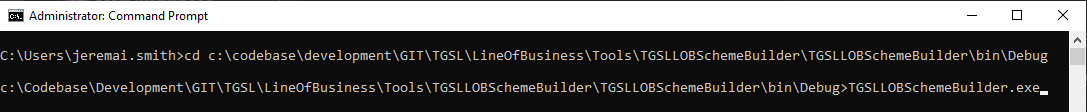
If this does not work, perform the following steps:

1. Browse to the XBroker test site at <https://staging.xbroker.com>
2. Click the **Sign in** button at top right
3. Log in as **denco3813** using the password **pipeline**
4. Click on **Policies and Quotations** (the new product won’t appear on the New Quotation page yet):  
   
5. Search for an existing customer, e.g. by searching for Surname / Company = ‘Smith’.
6. Select a search result and click **Add Quote**:  
   
7. If the new product exists in the drop-down, select it and click **Continue**:  
   
8. The newly created question set should appear. Try filling in the screens and check for any errors. It will not be possible to quote because no schemes have been created yet for the new Product.

# Generate Miscellaneous LOB SQL Scripts

## TGSL LOB Scheme Builder steps

This assumes you are using Git and have the TGSL repository cloned locally.

1. From your developer desktop, launch Microsoft Visual Studio
2. Open the solution GIT\TGSL\LineOfBusiness\Tools\TGSLLOBSchemeBuilder\ TGSLLOBSchemeBuilder.sln
3. From the Solution Explorer on the right of the screen, double click **App.config** under TGSLLOBSchemeBuilder to open it:  
   
4. The **System** setting is irrelevant as this step will only output some script files to a file share.
5. **ExecuteCalculator** should be the only Execute… setting set to true:  
   
6. Set the **TestPath** to the location where you would like to save scripts, for example the SQL subfolder within the Project folder that was created in Git. The folder must already exist. Also make sure the **LOBName** and **LineOfBusiness** are set correctly:  
   
7. The insurer and Scheme options can be ignored for now. Finally make sure the **PremiumSections** value is set to the sections that have been created or are required for this Product (multiple values can be separated with a comma):  
   
8. Right-click on TGSLLOBSchemeBuilder in the Solution Explorer and click **Rebuild**:  
   
9. The output window will show if the build succeeded (this may auto hide if not pinned):  
   
10. Launch Command Prompt and change directory to your local path for the following folder: GIT\TGSL\LineOfBusiness\Tools\TGSLLOBSchemeBuilder\TGSLLOBSchemeBuilder\bin\Debug  
    Then type or paste TGSLLOBSchemeBuilder.exe and press Enter to run it:  
    

Note, you can also double-click to run TGSLLOBSchemeBuilder.exe from within File Explorer, but this method does not allow you to read any error messages returned.

1. Check that scripts have been created in the TestPath folder location specified above.
2. Open SQL Server Management Studio and connect to **MHGSQL01\TGSLDEV**.
3. Run the following scripts that have been produced into the Transactor\_Live database:
   * dbo.<product code>\_LOBRiskView.View.sql: This view is used to create a standard bordereau report for the new Product
   * dbo. <product code>\_DocumentFormuale.Script.sql: This will insert formulae for the new premium section types into the SYSTEM\_DM\_FORMULA table. Document formulae are used in creation of printed policy documents.

Note, the other files produced can be ignored or deleted for now. They will form the basis of the first Scheme to be created for the new Product (see separate documentation for creating a Scheme).

1. Connect SQL Server Management Studio to **MHGSQL01\TGSLTEST**.
2. Run the same scripts again, in the Transactor\_Live database, to create the objects in UAT.

## Update uspPremiumCoverListSelect Procedure

* Connect SQL Server Management Studio to **MHGSQL01\TGSL** to take a copy of the latest live version of the stored procedure but do not make changes in live database yet!
* In the Transactor\_Live database in Object Explorer expand **Programmability > Stored Procedures**
* Locate the uspPremiumCoverListSelect procedure
* Right click and select **Modify**
* Save the script in GIT\TGSL\LineOfBusiness\Projects\<product code>\SQL
* Add a new section to the stored procedure for the new Product that will return all the coverages available on the new Product and the amount that has been selected on a quote or policy. Use the existing products in the procedure as a guide. You can either code the coverages directly in the procedure or create an external table valued function as has been done, for example, for MLIAB and MCLIAB.
* Save the changes.
* Run the script into the Transactor\_Live databases on **MHGSQL01\TGSLDEV** and **MHGSQL01\TGSLTEST**.

# Check in to Git

1. Copy all files from L:\Test\TCAS\Projects\<project code> folder and its subfolders to your local Git repository, i.e. GIT\TGSL\LineOfBusiness\Projects\<product code>\
2. Open Github Desktop or your preferred Git client
3. Commit the changes
4. Push the changes to Github

# Release to Live TCAS

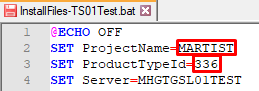
After a Scheme has been created and tested against the new Product (see separate Scheme documentation):

1. Open SQL Server Management Studio and connect to **MHGSQL01\TGSL**.
2. Copy the entire folder L:\Test\TCAS\Projects\<product code> to L:\TCAS\Projects\<product code>
3. Run each of the three scripts that were created and run in Live into the Transactor\_Live database, i.e.:  
   L:\TCAS\Projects\<project code>\<project code> Product.sql

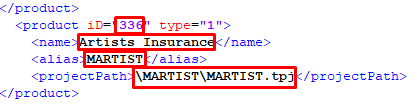
L:\TCAS\Projects\<project code>\<project code> Build.sql

L:\ TCAS\Projects\<project code>\<project code> Build part 2.sql

1. Locate the batch file GIT\TGSL\LineOfBusiness\Install Project\InstallFiles-TS01.bat but DO NOTdouble-click as this will run it!
2. Right-click on InstallFiles-TS01.bat and choose to edit with Notepad++ or other text editor that you have installed.
3. Update the **ProjectName** and **ProductTypeId** values at the top of the batch script:



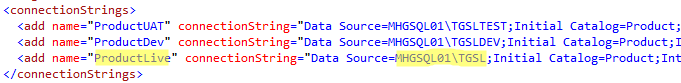
1. Double click InstallFiles-TS01.bat to run it. For information, this will perform the following actions to install the new project into TCAS:
   1. Copies all files from L:\Test\TCAS\Projects\<product code> to L:\TCAS\Templates\<product code>
   2. Copies all files from L:\Test\TCAS\Projects\<product code> to L:\TCAS\Stylesheets\<product code>
   3. Copies XML files from L:\Test\TCAS\Projects\<product code>\WebService and subfolders folder to corresponding folders on C:\inetpub\wwwroot on the Test TGSL server MHGTGSL01
2. Repeat these steps for InstallFiles-TS02.bat and InstallFiles-TS03.bat. This will copy the WebService files to the additional two live application servers.
3. Locate the file L:\TCAS\Templates\Screenbuilder.tsb and open with Notepad++ or another text editor.
4. Copy and paste an existing five line product element and change the **Product ID** attribute to the ProductTypeID for the new Product, and change the **name**, **alias** and **projectPath** child elements accordingly:



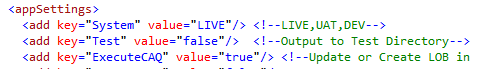
1. Save and close the file
2. Inform Operations that the Product has been released, pending a restart of the Transactor Relationship Manager service. DO NOT restart the service on any of the live servers as it will affect users using TCAS. There are two automatic restarts scheduled every day, or Operations may schedule an additional restart.

# Release to Live Constructaquote.com web site

Follow the steps in the [Release to Test Constructaquote.com web site](#_Release_to_Test) section above but make the following changes. In the App.config, add a ProductLive connection string:



Set **System** to LIVE and **Test** to false:



In Configuration.cs, where the ProductDatabase variable is set, comment out the “UAT” line and uncomment the “LIVE” line (this is left commented out to prevent accidentally overwriting anything in the live Product database if the System value in App.config is unintentially left pointing to live):



When copying the files to the server, the location will be:

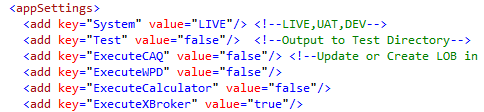
\\MHGWEB01\d$\web\_roots\Live\TGSLWebsiteLive\quote

Connect to **MHGSQL01\TGSL** in SQL Server Management Studio and run the synonym, answer set function and any CAQ web site fix scripts into the Product database. These scripts should all have been saved in the following location:

GIT\TGSL\LineOfBusiness\Projects\<product code>\SQL\Product

# Release to Live XBroker web site

Follow the steps in the [Release to Test XBroker website](#_Release_to_Test_1) section above, but with **System** set to LIVE and **Test** set to false:



When copying the files to the server, the location will be (files should already have been copied if Product released to CAQ):

\\MHGWEB01\d$\web\_roots\Live\TGSLWebsiteLive\quote

# Run Miscellaneous LOB SQL Scripts

1. Open SQL Server Management Studio and connect to **MHGSQL01\TGSL**.
2. Run the following scripts that were saved to location defined in the [Generate Miscellaneous LOB SQL Scripts](#_Generate_Miscellaneous_LOB) section above (i.e. GIT\TGSL\LineOfBusiness\Projects\<product code>\SQL):
   * dbo.<product code>\_LOBRiskView.View.sql
   * dbo. <product code>\_DocumentFormuale.Script.sql

Note, if the Product is going live it is likely that the first Scheme will be released live at this point as well (see separate Scheme documentation).

# Troubleshooting

## Screen Designer Errors on Release Project

### “Screen Designer was unable to find a required file:C:\Program Files (x86)\Wix\candle.exe” on Release Project

The latest version of the WiX toolset can be downloaded from <https://wixtoolset.org/releases/>. Once installed, Screen Designer expects two executables to be in the root C:\Program Files (x86)\Wix folder instead of the bin subfolder. Move candle.exe and light.exe out of bin into the root footer. If you do not have file extensions visible, ensure you are moving the executables candle.exe and light.exe, and not the config files candle.exe.config and light.exe.config!

### “Error in TES\_Screen\_Deployment.clsMSI” on Release Project

Needs investigating; seem to be just warnings though, as screens release to TCAS OK



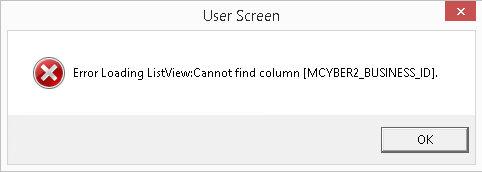
### Screen Designer freezes on Release Project

Needs investigating but screens release to TCAS OK

## TCAS Errors

### Error Loading List View

This error is encountered when relationships have not been set correctly between parent and child screens in Screen Designer:

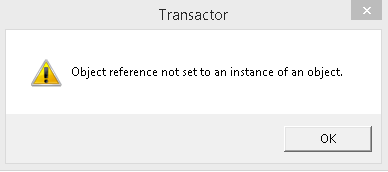


The issue is that the child USER\_... database table has been created with POLICY\_DETAILS\_ID and HISTORY\_ID columns instead of a primary key that relates to the parent table.

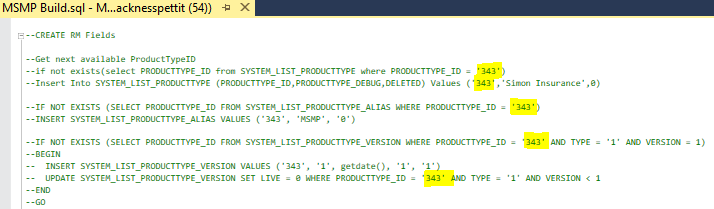
To fix this, follow the steps to set a relationship under the [Creating child screens](#_Creating_child_screens) section above. This can be done visually in Screen Designer or by editing the <parentForm> XML attribute in the .tpj (screen designer project) file build script, copy the files to Templates and Stylesheets and copy the XML to C:\inetpub\wwwroot\XML\user again

### Quote journey

This error will be encountered when the Product ID is incorrect. Check the @ProductID parameter in the ‘MSMP Product.sql’ script is the same as the once commented out at the beginning of the ‘MSMP Build.sql’ script







# Appendices

## Appendix I: New list table script

EXEC [dbo].[SP\_PM\_CREATE\_LISTTABLE] 'MH\_BUSINESS\_ACTIVITY'; -- Procedure creates LIST\_MH\_BUSINESS\_ACTIVITY, LIST\_MH\_BUSINESS\_ACTIVITY\_LINK and LIST\_MH\_BUSINESS\_ACTIVITY\_VIEW, with all necessary constraints, and grants permissions so they work with Screen Designer and TGSL

GO

INSERT INTO [dbo].[LIST\_MH\_BUSINESS\_ACTIVITY](

[MH\_BUSINESS\_ACTIVITY\_ID]

,[MH\_BUSINESS\_ACTIVITY\_DEBUG]

,[DELETED]

,[ABIVALUE]

)

VALUES

('0', NULL, 0, 0) -- Required if blank value is allowable (e.g. if new field is not mandatory or adding new list table field to an existing line of business where existing records won’t have a vlue populated), otherwise the INNER JOIN in the USER\_READ… procedure will prevent these quotes / policies with NULL value from loading in TGSL

,('ABATTOIR', 'Abattoir/Slaughterhouse', 0, 0)

,('AMUSEARC', 'Amusement Arcades', 0, 0)

,('ASBESTOS', 'Asbestos Manufacturer', 0, 0)

;

GO

INSERT INTO [dbo].[LIST\_MH\_BUSINESS\_ACTIVITY\_LINK](

[MH\_BUSINESS\_ACTIVITY\_ID]

,[PORTFOLIOKEY]

,[AGENT\_ID]

,[PRODUCT\_ID]

,[MH\_BUSINESS\_ACTIVITY\_DEFAULT]

)

SELECT

[MH\_BUSINESS\_ACTIVITY\_ID]

,154

,NULL -- NULL means list values will be available under all agents

,NULL -- NULL means list values will be available under all products, but since the table is being created for a new Product, it won't be uses for any other Products yet

,0

FROM

[dbo].[LIST\_MH\_BUSINESS\_ACTIVITY]

;

GO

## Appendix II: Files produced by Screen Designer

Your Project folder will be populated with the Screen Builder output files:

* frm[A-Za-z]{8}.tst - xml - Screen Control Information
* {LOB} Build.sql -SQL -Risk tables, CRUD Procedures and Navigation
* {LOB} v1 Definition.docx - Word Doc - Describes the Project, TCAS screens, controls and properties
* {LOB}\_ProjectFiles\_v{n}.msi - MS Installer - (Don't use)
* Build[ Edit\_Version\_{n}].html - Webpage - Describes the content of the website folder
* {LOB}.tpj - TCAS Project file - Form Properties and the summary form and control properties
* {LOB}.trv - Relationships - Initial screen position of forms
* {LOB}.tsd - versioning - Tracks version when Screens and controls were created and deleted
* Images - Folder - Contains screen jpegs used in {LOB} v1 Definition.docx
* Webservice - Folder - Root folder - Schemas, Transforms for distribution to web server

| Folder | Filename | Type | Description |
| --- | --- | --- | --- |
| outcome | outcome{ProductTypeID}.xslt | transform | Outputs an xml instance of outcome.xsd from risk and client data. |
| read | {ProductTypeID}Read.xslt | transform | Outputs an xml of Screens populated with risk data |
| read | ds{ProductTypeID}Read.xslt | transform | Outputs an xml instance of dsCustomer.xsd user and customer dataset. Apart from boolean to numeric conversion and namespace it's identical to {ProductTypeID}Read.xslt. |
| save | {ProductTypeID}Delete.xslt | transform | Outputs a SQL Deletion procedure call to remove the risk for a policy history from all LOB risk tables. |
| save | {ProductTypeID}Save.xslt | transform | Outputs an array of procedure names and parameters for saving all risk data. |
| XML | {ProductTypeID}Read.xml | data | an array of procedure names and parameternames for reading all risk data. |
| XML | {ProductTypeID}Schema1.xsd | Schema | microsoft dataset schema describing the LOB risk tables and relationships |

* Website - Folder- Files for distribution to a TGSL based quote website such as XBroker

| Folder | Filename | Type | Description |
| --- | --- | --- | --- |
| App\_Data | {LOB}.xml | data | Control value defaults and visibility logic |
| config | {LOB}\_config.xml | config | Screen Configuration (Defaults genearally OK) |
| content | {LOB}\_content.xml | data | Extended question text for the web pages. Initially populated from the controls description property but this has a length restriction. Can be regenerated using the [LOB Scheme Builder](http://localhost:8800/doku.php?id=development:tools:tgsllobschemebuilder#xbrokerwriter) |
| documents | empty | | |
| {LOB} | {ScreenName}.aspx | asp.net pages | Basic screen content and layout for processing to pure html risk pages |
| Stylesheets..user | ds195Read.xslt | Transform | copy of file in webservice\read |
| tracking | empty | | |
| usercontrols | empty | | |

Top of Form

Bottom of Form

## Appenidx III: Product and Product links script

DECLARE @ProductID CHAR(32) = REPLACE(NewID(),'-','')

DECLARE @ProductTypeID VARCHAR(8) = 336 – Change to ID given in the build script produced by the release (that we comment out on next step)

DECLARE @ProductName VARCHAR(255) = 'Cyber' -- CHANGE

DECLARE @Reference VARCHAR(30) = 'CY' -- CHANGE

DECLARE @PARAMETER\_GROUP\_ID CHAR(32) = '00EB5E136E9F40D4A384E24D6B717667' --Commercial Product Parameters

DECLARE @LOBName VARCHAR(50) = 'MCYBER2' -- CHANGE

DECLARE @ConstructaquotecomID CHAR(32) = '5208F39A498E4706A91BEEC84ED25686'

DECLARE @ConstructaquoteID CHAR(32) = 'E2F376B621E14C1FB532CED74C7EDCE1'

DECLARE @XBrokerID CHAR(32) = '0F849A389DD4477CAF66BBCBECA49AA4'

INSERT INTO [dbo].[LIST\_PRODUCT\_TYPE] ([PRODUCT\_TYPE\_ID],[PRODUCT\_TYPE\_DEBUG],[DELETED],[ABIVALUE]) VALUES (@ProductTypeID,@ProductName,0,0);

INSERT INTO SYSTEM\_LIST\_PRODUCTTYPE (PRODUCTTYPE\_ID,PRODUCTTYPE\_DEBUG,DELETED) VALUES (@ProductTypeID,@ProductName,0);

INSERT SYSTEM\_LIST\_PRODUCTTYPE\_ALIAS VALUES (@ProductTypeID, @LOBName, '0');

INSERT SYSTEM\_LIST\_PRODUCTTYPE\_VERSION VALUES (@ProductTypeID, '1', getdate(), '1', '1');

UPDATE SYSTEM\_LIST\_PRODUCTTYPE\_VERSION SET LIVE = 0 WHERE PRODUCTTYPE\_ID = @ProductTypeID AND TYPE = '1' AND VERSION < 1;

INSERT INTO [dbo].[RM\_PRODUCT]([PRODUCT\_ID], [NAME], [REFERENCE], [PARAMETER\_GROUP\_ID], [PRODUCTTYPE\_ID], [NEWQUOTENAVTABLE], [ADDQUOTENAVTABLE], [POSTQUOTENAVTABLE], [GENERALNAVTABLE], [USERCLAIMNAVTABLE], [DELETED], [DELETEDDATE])

VALUES (@ProductID, @ProductName,@Reference,@PARAMETER\_GROUP\_ID,@ProductTypeID,'NAV\_FORMS\_' + @LOBName + '\_NEW','NAV\_FORMS\_' + @LOBName + '\_ADD','NAV\_FORMS\_' + @LOBName + '\_PST','NAV\_FORMS\_' + @LOBName + '\_IND',NULL ,0,NULL);

INSERT INTO [dbo].[RM\_AGENT\_PRODUCT\_LINK]([AGENT\_ID], [PRODUCT\_ID])

VALUES

(@ConstructaquotecomID,@ProductID) -- Constructaquote.com

,(@ConstructaquoteID,@ProductID) -- Constructaquote

,(@XBrokerID,@ProductID) -- XBroker

;

INSERT INTO [dbo].[RM\_PRODUCT\_PAYMENT\_PLAN\_LINK]

(

[AGENT\_ID]

,[PRODUCT\_ID]

,[PAYMENT\_PLAN\_ID]

,[DEFAULTYN]

)

SELECT

[AGENT\_ID]

,@ProductID

,[RMPP].[PAYMENT\_PLAN\_ID]

,CASE

WHEN [RMA].[NAME] = 'XBroker' AND [RMPP].[NAME] = 'Sub Agent Only' THEN 1

WHEN [RMA].[NAME] IN ('Constructaquote.com', 'Constructaquote') AND [RMPP].[NAME] = '(Annual) One Payment In Full Inc Fee' THEN 1

ELSE 0

END AS [DEFAULTYN]

FROM

[dbo].[RM\_PAYMENT\_PLAN] AS [RMPP]

CROSS JOIN [dbo].[RM\_AGENT] AS [RMA]

WHERE

([RMA].[NAME] = 'XBroker' AND [RMPP].[NAME] = 'Sub Agent Only')

OR

([RMA].[NAME] IN ('Constructaquote.com', 'Constructaquote')

AND [RMPP].[NAME] IN ('(Annual) One Payment In Full Inc Fee'

,'Annual Instalments (10% Deposit) Bank Details to Follow'

,'Annual Instalments Excluding CV & PC (10% Deposit)'

,'Annual Instalments Excluding CV & PC (30% Deposit)'

,'Deposit Payment (10%) Balance Invoiced'

,'Multiple Policies on Annual Instalments'

,'Multiple Policies on Annual Instalments Bank Details to Follow'

,'Renewal Roll Over on Annual Instalments'

,'Winback Payment Plan')

AND [RMPP].[DELETED] <> 1

)

;

INSERT INTO [dbo].[RM\_WEBBRANDING]

(

[WEBBRANDING\_ID]

,[AGENT\_ID]

,[PRODUCT\_ID]

,[URL\_LINKFROM]

,[SALES\_TEL]

,[SERVICE\_TEL]

,[RENEWALS\_TEL]

,[CLAIMS\_TEL]

,[COMPLAINTS\_TEL]

,[ERROR\_TEL]

,[REJECTED\_PAYMENT\_TEL]

,[FROM\_EMAIL]

,[BCC\_EMAIL]

)

VALUES

(

REPLACE(NEWID(),'-','')

,@ConstructaquotecomID

,@ProductID

,'CAQ' + @Reference

,'0808 168 6868'

,'0808 168 6868'

,'0808 168 6868'

,'0808 168 6868'

,'0808 168 6868'

,'0808 168 6868'

,'0808 168 6868'

,'noreply@constructaquote.com'

,'tgsl-mailbox@moorhouseinsurance.co.uk'

)

,

(

REPLACE(NEWID(),'-','')

,@XBrokerID

,@ProductID

,'XB' + @Reference

,'029 2080 8963'

,'029 2080 8963'

,'029 2080 8963'

,'029 2080 8963'

,'029 2080 8963'

,'029 2080 8963'

,'029 2080 8963'

,'noreply@xbroker.com'

,'tgsl-mailbox@moorhouseinsurance.co.uk'

)

;

## Appendix IV: Mandatory and default fields script

--ADD

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_ADD\_044]

FROM

[dbo].[Nav\_Forms\_MCYBER2\_ADD] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_ADD] AS [F] ON [S].[NAV\_MCYBER2\_ADD\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_ADD\_044] = 'Business Details'

AND [F].[FIELDNAME] IN ('cboInsurer', 'cboBusinessDesc', 'cboBusinessDesc2', 'cboCompStatus', 'cfdTurnover', 'txtRecords', 'lvwPandPList', 'cboIndemnity')

GO

UPDATE

[F]

SET

[DEFAULT\_VALUE\_044] = CASE [FieldName]

WHEN 'cboBusinessDesc2' THEN 'None'

WHEN 'optDomicile' THEN '1'

WHEN 'OptWorkLoc' THEN '1'

WHEN 'optIndustry' THEN '0'

WHEN 'optSellShare' THEN '0'

WHEN 'optLossIncome' THEN '1'

ELSE NULL

END

FROM

[dbo].[Nav\_Forms\_MCYBER2\_ADD] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_ADD] AS [F] ON [S].[NAV\_MCYBER2\_ADD\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_ADD\_044] = 'Business Details'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_ADD\_044]

FROM

[dbo].[Nav\_Forms\_MCYBER2\_ADD] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_ADD] AS [F] ON [S].[NAV\_MCYBER2\_ADD\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_ADD\_044] = 'Claim Summary'

AND [F].[FIELDNAME] = 'lvwClmSumList'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_ADD\_044]

,[DEFAULT\_VALUE\_044] = 'Agree'

FROM

[dbo].[Nav\_Forms\_MCYBER2\_ADD] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_ADD] AS [F] ON [S].[NAV\_MCYBER2\_ADD\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_ADD\_044] = 'Assumptions'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = CASE [F].[FIELDNAME]

WHEN 'QE140\_Cbo\_Campaign' THEN 'Please enter a Campaign'

WHEN 'QE140\_Cbo\_Activity' THEN 'Please enter an Activity'

WHEN 'QE140\_Cbo\_Source' THEN 'Please enter a Source of Business'

WHEN 'QE140\_Cbo\_Preferred' THEN 'Please select the Preferred method of contact'

END

,[DEFAULT\_VALUE\_044] = CASE [F].[FIELDNAME]

WHEN 'QE140\_Cbo\_Campaign' THEN 'Call Center'

WHEN 'QE140\_Cbo\_Preferred' THEN 'Email'

ELSE ''

END

FROM

[dbo].[Nav\_Forms\_MCYBER2\_ADD] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_ADD] AS [F] ON [S].[NAV\_MCYBER2\_ADD\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_ADD\_044] = 'Contact and Marketing Details'

AND [F].[FIELDNAME] IN ('QE140\_Cbo\_Campaign', 'QE140\_Cbo\_Activity', 'QE140\_Cbo\_Source', 'QE140\_Cbo\_Preferred')

GO

--NEW

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_NEW\_044]

FROM

[dbo].[Nav\_Forms\_MCYBER2\_NEW] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_NEW] AS [F] ON [S].[NAV\_MCYBER2\_NEW\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_NEW\_044] = 'Business Details'

AND [F].[FIELDNAME] IN ('cboInsurer', 'cboBusinessDesc', 'cboBusinessDesc2', 'cboCompStatus', 'cfdTurnover', 'txtRecords', 'lvwPandPList', 'cboIndemnity')

GO

UPDATE

[F]

SET

[DEFAULT\_VALUE\_044] = CASE [FieldName]

WHEN 'cboBusinessDesc2' THEN 'None'

WHEN 'optDomicile' THEN '1'

WHEN 'OptWorkLoc' THEN '1'

WHEN 'optIndustry' THEN '0'

WHEN 'optSellShare' THEN '0'

WHEN 'optLossIncome' THEN '1'

ELSE NULL

END

FROM

[dbo].[Nav\_Forms\_MCYBER2\_NEW] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_NEW] AS [F] ON [S].[NAV\_MCYBER2\_NEW\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_NEW\_044] = 'Business Details'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_NEW\_044]

FROM

[dbo].[Nav\_Forms\_MCYBER2\_NEW] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_NEW] AS [F] ON [S].[NAV\_MCYBER2\_NEW\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_NEW\_044] = 'Claim Summary'

AND Fieldname = 'lvwClmSumList'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_NEW\_044]

,[DEFAULT\_VALUE\_044] = 'Agree'

FROM

[dbo].[Nav\_Forms\_MCYBER2\_NEW] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_NEW] AS [F] ON [S].[NAV\_MCYBER2\_NEW\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_NEW\_044] = 'Assumptions'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = CASE [F].[FIELDNAME]

WHEN 'QE140\_Cbo\_Campaign' THEN 'Please enter a Campaign'

WHEN 'QE140\_Cbo\_Activity' THEN 'Please enter an Activity'

WHEN 'QE140\_Cbo\_Source' THEN 'Please enter a Source of Business'

WHEN 'QE140\_Cbo\_Preferred' THEN 'Please select the Preferred method of contact'

END

,[DEFAULT\_VALUE\_044] = CASE [F].[FIELDNAME]

WHEN 'QE140\_Cbo\_Campaign' THEN 'Call Center'

WHEN 'QE140\_Cbo\_Preferred' THEN 'Email'

ELSE ''

END

FROM

[dbo].[Nav\_Forms\_MCYBER2\_NEW] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_NEW] AS [F] ON [S].[NAV\_MCYBER2\_NEW\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_NEW\_044] = 'Contact and Marketing Details'

AND [F].[FIELDNAME] IN ('QE140\_Cbo\_Campaign', 'QE140\_Cbo\_Activity', 'QE140\_Cbo\_Source', 'QE140\_Cbo\_Preferred')

GO

--IND

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_IND\_044]

FROM

[dbo].[Nav\_Forms\_MCYBER2\_IND] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_IND] AS [F] ON [S].[NAV\_MCYBER2\_IND\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_IND\_044] = 'Business Details'

AND [F].[FIELDNAME] IN ('cboInsurer', 'cboBusinessDesc', 'cboBusinessDesc2', 'cboCompStatus', 'cfdTurnover', 'txtRecords', 'lvwPandPList', 'cboIndemnity')

GO

UPDATE

[F]

SET

[DEFAULT\_VALUE\_044] = CASE [FieldName]

WHEN 'cboBusinessDesc2' THEN 'None'

WHEN 'optDomicile' THEN '1'

WHEN 'OptWorkLoc' THEN '1'

WHEN 'optIndustry' THEN '0'

WHEN 'optSellShare' THEN '0'

WHEN 'optLossIncome' THEN '1'

ELSE NULL

END

FROM

[dbo].[Nav\_Forms\_MCYBER2\_IND] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_IND] AS [F] ON [S].[NAV\_MCYBER2\_IND\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_IND\_044] = 'Business Details'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_IND\_044]

FROM

[dbo].[Nav\_Forms\_MCYBER2\_IND] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_IND] AS [F] ON [S].[NAV\_MCYBER2\_IND\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_IND\_044] = 'Claim Summary'

AND Fieldname = 'lvwClmSumList'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = 'Please validate - ' + [F].[Nav\_MCYBER2\_IND\_044]

,[DEFAULT\_VALUE\_044] = 'Agree'

FROM

[dbo].[Nav\_Forms\_MCYBER2\_IND] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_IND] AS [F] ON [S].[NAV\_MCYBER2\_IND\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_IND\_044] = 'Assumptions'

GO

UPDATE

[F]

SET

[Mandatory\_044] = 1

,[Mandatory\_Message\_044] = CASE [F].[FIELDNAME]

WHEN 'QE140\_Cbo\_Campaign' THEN 'Please enter a Campaign'

WHEN 'QE140\_Cbo\_Activity' THEN 'Please enter an Activity'

WHEN 'QE140\_Cbo\_Source' THEN 'Please enter a Source of Business'

WHEN 'QE140\_Cbo\_Preferred' THEN 'Please select the Preferred method of contact'

END

,[DEFAULT\_VALUE\_044] = CASE [F].[FIELDNAME]

WHEN 'QE140\_Cbo\_Campaign' THEN 'Call Center'

WHEN 'QE140\_Cbo\_Preferred' THEN 'Email'

ELSE ''

END

FROM

[dbo].[Nav\_Forms\_MCYBER2\_IND] AS [S]

JOIN [dbo].[Nav\_Fields\_MCYBER2\_IND] AS [F] ON [S].[NAV\_MCYBER2\_IND\_ID] = [F].[ParentID]

WHERE

[S].[NAV\_MCYBER2\_IND\_044] = 'Contact and Marketing Details'

AND [F].[FIELDNAME] IN ('QE140\_Cbo\_Campaign', 'QE140\_Cbo\_Activity', 'QE140\_Cbo\_Source', 'QE140\_Cbo\_Preferred')

GO

## Appendix V: Product database question set query

SELECT

[QS].[QuestionSetName]

,[S].[Text] AS [Section Name]

,[Q].[QuestionID]

,[Q].[AnswerTableName]

,[Q].[AnswerFieldName]

,[AQD].[Text] -- Screen field label

,[AQD].[HelpText] -- Help text shown when clicking 'i' symbol

,[AQD].[SectionID]

,[AQD].[SortOrder] -- Sort order from TabIndex in Screen Designer

,[AQD].[AnswerDefaultValueOrID] -- Should be set to 3MQT5MC6 (Yes) for assumptions

,[AQD].[AnswerDefaultSet] -- Should be set to 1 for assumptions

,[AQD].[Enabled]

,[AQD].[Visible]

,[AQD].[Mandatory]

FROM

[QuestionSet].[QuestionSet] AS [QS]

INNER JOIN [QuestionSet].[Question] AS [Q] ON [QS].[QuestionSetID] = [Q].[QuestionSetID]

INNER JOIN [QuestionSet].[AgentQuestionDetails] AS [AQD] ON [Q].[QuestionID] = [AQD].[QuestionID]

LEFT JOIN [QuestionSet].[Section] AS [S] ON [AQD].[SectionID] = [S].[SectionID]

WHERE

[QS].[QuestionSetName] = 'Artists Insurance'

ORDER BY

[AQD].[SectionID]

,[AQD].[SortOrder]

## Appendix VI: Product database calculated questions query

SELECT

[QS].[QuestionSetName]

,[CQO].[CalculatedQuestionID]

,[CQ].[AnswerTableName] AS [Calculated Question Table]

,[CQ].[AnswerFieldName] AS [Calculated Question Field]

,[AQD].[Enabled]

,[AQD].[Visible]

,[CQO].[ParameterQuestionID]

,[PQ].[AnswerTableName] AS [Parameter Question Table]

,[PQ].[AnswerFieldName] AS [Parameter Question Field]

,[CQO].[ParameterValueFunctionID] -- Refers to the types in QuestionSet.tvfComparatorValueFunction

,[CQO].[Position] -- If calculated from multiple fields

,[CQO].[Operator]

FROM

[QuestionSet].[CalculatedQuestionOperation] AS [CQO]

INNER JOIN [QuestionSet].[Question] AS [CQ] ON [CQO].[CalculatedQuestionID] = [CQ].[QuestionID]

INNER JOIN [QuestionSet].[QuestionSet] AS [QS] ON [CQ].[QuestionSetID] = [QS].[QuestionSetID]

LEFT JOIN [QuestionSet].[AgentQuestionDetails] AS [AQD] ON [CQ].[QuestionID] = [AQD].[QuestionID]

LEFT JOIN [QuestionSet].[Question] AS [PQ] ON [CQO].[ParameterQuestionID] = [PQ].[QuestionID]

## Appendix VII: Product database calculated question insert example

DECLARE @CalculatedQuestionID bigint

DECLARE @ParameterQuestionID bigint

DECLARE @ParameterValue decimal(18,2)

DECLARE @ParameterValueFunctionID int = 2

DECLARE @Position int = 1

DECLARE @Operator varchar(6) = '+'

DECLARE @InsertUserID bigint = 1

SELECT @CalculatedQuestionID = [QuestionID] from [QuestionSet].[Question] WHERE [AnswerTableName] = 'USER\_MLETPROP\_PROPINFO' AND AnswerFieldName = 'Yrs'

EXEC [QuestionSet].[uspCalculatedQuestionOperationInsert] @CalculatedQuestionID, @ParameterQuestionID, @ParameterValue, @ParameterValueFunctionID, @Position, @Operator, @InsertUserID

SELECT @ParameterQuestionID = questionid from questionset.question WHERE AnswerTableName = 'USER\_MLETPROP\_PROPINFO' AND AnswerFieldName = 'Established'

SET @Position = 2

SET @ParameterValueFunctionID = NULL

SET @Operator = '-yyyy'

EXEC [QuestionSet].[uspCalculatedQuestionOperationInsert] @CalculatedQuestionID, @ParameterQuestionID, @ParameterValue, @ParameterValueFunctionID, @Position, @Operator, @InsertUserID

GO

UPDATE

[AQD]

SET

[AQD].[Enabled] = 0

,[AQD].[Visible] = 0

,[AQD].[Mandatory] = 0

FROM

[QuestionSet].[AgentQuestionDetails] AS [AQD]

JOIN [QuestionSet].[Question] AS [Q] ON [AQD].[QuestionID] = [Q].[QuestionID]

WHERE

[Q].[AnswerTableName] = 'USER\_MLETPROP\_PROPINFO'

AND [Q].[AnswerFieldName] = 'Yrs'

GO

## Appendix VIII: Product database enablement criteria query

SELECT

[EC].[EnablementCriteriaID]

,[EC].[EnablementCriteriaSetID]

,[EC].[ComparatorQuestionID]

,[QS].[QuestionSetName]

,[Q].[AnswerTableName]

,[Q].[AnswerFieldName]

,[EC].[Operator]

,[EC].[ComparatorValueOrID]

,[EC].[EnablementCriteriaListID]

,(SELECT DISTINCT [ListValue] + ',' FROM [QuestionSet].[EnablementCriteriaListItem]

WHERE [EnablementCriteriaListID] = [EC].[EnablementCriteriaListID]

FOR XML PATH('')) AS [CriteriaListItems]

,(SELECT DISTINCT [LINKQ].[AnswerTableName] + '.' + [LINKQ].[AnswerFieldName] + ',' FROM [QuestionSet].[QuestionEnablementCriteriaSetLink] AS [CSL]

INNER JOIN [QuestionSet].[Question] AS [LINKQ] ON [CSL].[QuestionID] = [LINKQ].[QuestionID]

WHERE [CSL].[EnablementCriteriaSetID] = [EC].[EnablementCriteriaSetID]

FOR XML PATH('')) AS [EnabledQuestions]

FROM

[QuestionSet].[EnablementCriteria] AS [EC]

INNER JOIN [QuestionSet].[Question] AS [Q] ON [EC].[ComparatorQuestionID] = [Q].[QuestionID]

INNER JOIN [QuestionSet].[QuestionSet] AS [QS] ON [Q].[QuestionSetID] = [QS].[QuestionSetID]

WHERE

[QS].[QuestionSetName] = 'Artists insurance'

## Appendix IX: Product database question enablement insert example

IF NOT EXISTS (SELECT 1 FROM [QuestionSet].[EnablementCriteria] AS [EC]

INNER JOIN [QuestionSet].[Question] AS [Q] ON [EC].[ComparatorQuestionID] = [Q].[QuestionID]

WHERE [Q].[AnswerTableName] = 'USER\_MLETPROP\_PROPINFO' AND [Q].[AnswerFieldName] = 'Insurer\_ID')

BEGIN

DECLARE @ComparatorQuestionId bigint = (SELECT [QuestionID] FROM [QuestionSet].[Question] WHERE [AnswerTableName] = 'USER\_MLETPROP\_PROPINFO' AND [AnswerFieldName] = 'Insurer\_ID')

DECLARE @EnablementListValueString varchar(max) = NULL -- Multiple answer values to the Comparator Question, separated by commas, that enable the Question; leave NULL if single value and use @ComparatorValueOrID

DECLARE @EnabledQuestionID bigint = (SELECT [QuestionID] FROM [QuestionSet].[Question] WHERE [AnswerTableName] = 'USER\_MLETPROP\_PROPINFO' AND [AnswerFieldName] = 'PolicyExpire')

DECLARE @Operator varchar(5) = '!=' -- Note, 'not equal to' must use '!=' instead of '<>', equal to must use "=="

DECLARE @ComparatorValueOrID varchar(255) = '998' -- 998 = No Previous Insurer. -- The answer value to the Comparator Question that enables the Question; think this can be left NULL if @EnablementListValueString is provided

DECLARE @EnablementCriteriaSetID bigint = NULL

DECLARE @InsertUserID bigint = 1 -- Not sure what this supposed to reference; they are all set to 1

EXECUTE [QuestionSet].[uspEnablementInsert] @ComparatorQuestionId, @EnablementListValueString, @EnabledQuestionID, @Operator, @ComparatorValueOrID, @EnablementCriteriaSetID, @InsertUserID

END

GO

Bottom of Form