

DESIGN STUDIO

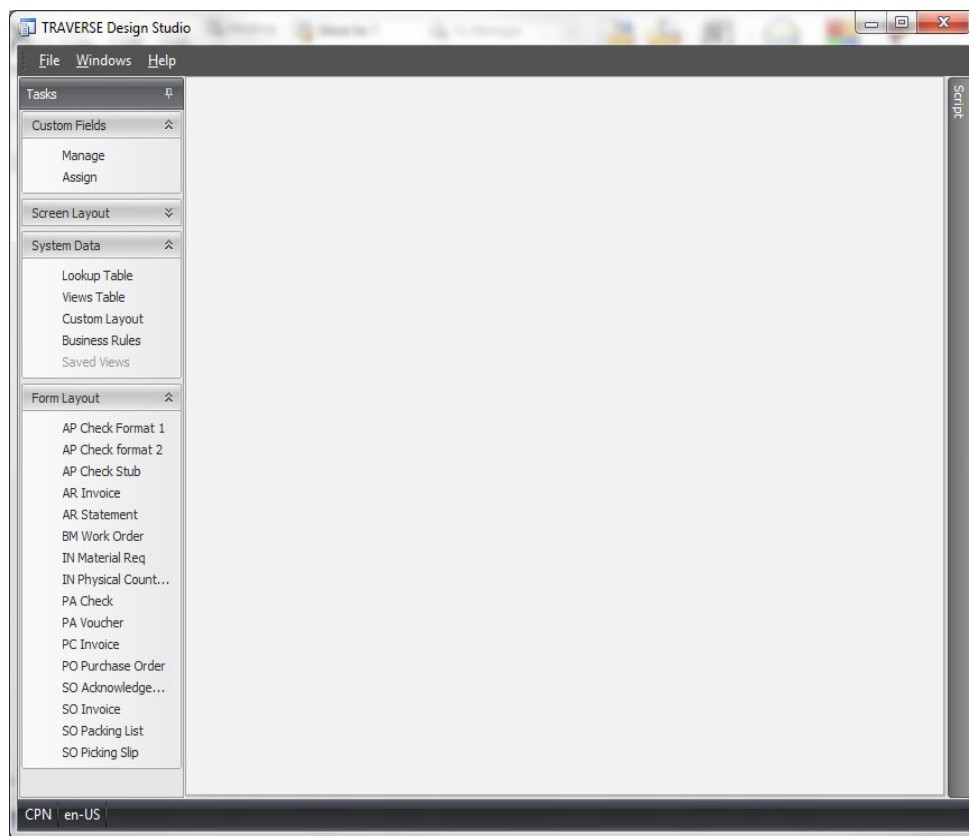
Design Studio Overview	3-3
The TRAVERSE Design Studio Main Screen	3-3
The Script Window	3-4
Using Design Studio	3-9
Custom Fields	3-9
Screen Layout	3-16
Customize the Lookups and Interactive Views	3-29
System Data	3-30
Business Rules	3-38
Form Layout	3-42
Scripting	3-46
Add an image (logo) to a form in Design Studio	3-60
Design Studio Examples	3-47
Custom Fields	3-47
Add an image (logo) to a form in Design Studio	3-60
Design Studio - System Data	3-66
Example of a Lookup Table modification:	3-68
Editing the System Views Table	3-72
Custom Scripting	3-76

DESIGN STUDIO OVERVIEW

TRAVERSE Design Studio is a set of tools that allows you to make many kinds of customizations to your TRAVERSE applications. Though the tools vary in function and use, the goal is the same: to empower you to adapt the TRAVERSE software to your business needs.

The TRAVERSE Design Studio Main Screen

The main screen of the TRAVERSE Design Studio gives you access to each of the design tools on the Tasks menu on the left side of the screen.



Click on one of the options under the Tasks menu to open the associated TRAVERSE Design Studio tool.

The Menu Bar

Click **File** to change TRAVERSE companies or to exit the TRAVERSE Design Studio.

Click **Windows** to view a list of the open windows.

TRAVERSE Design Studio includes the following tools:

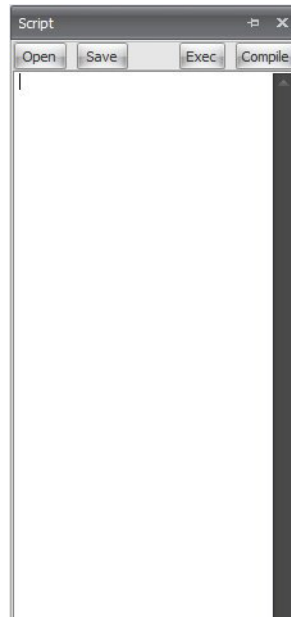
- Custom Fields
- Screen Layout
- System Data
- Form Layout
- Scripting

The Script Window

Scripts are a specific type of programming that can be used to add functionality to the software without making changes to the source code of the program. Scripts can be used to add validations, restrictions, calculations and otherwise modify the behavior of the software.

In TRAVERSE, scripts are stored in files outside of the core software. This allows you to receive updates and enhancements to your software while generally leaving your customizations intact.

Click **Script** on the right side of the screen to open the Script window.



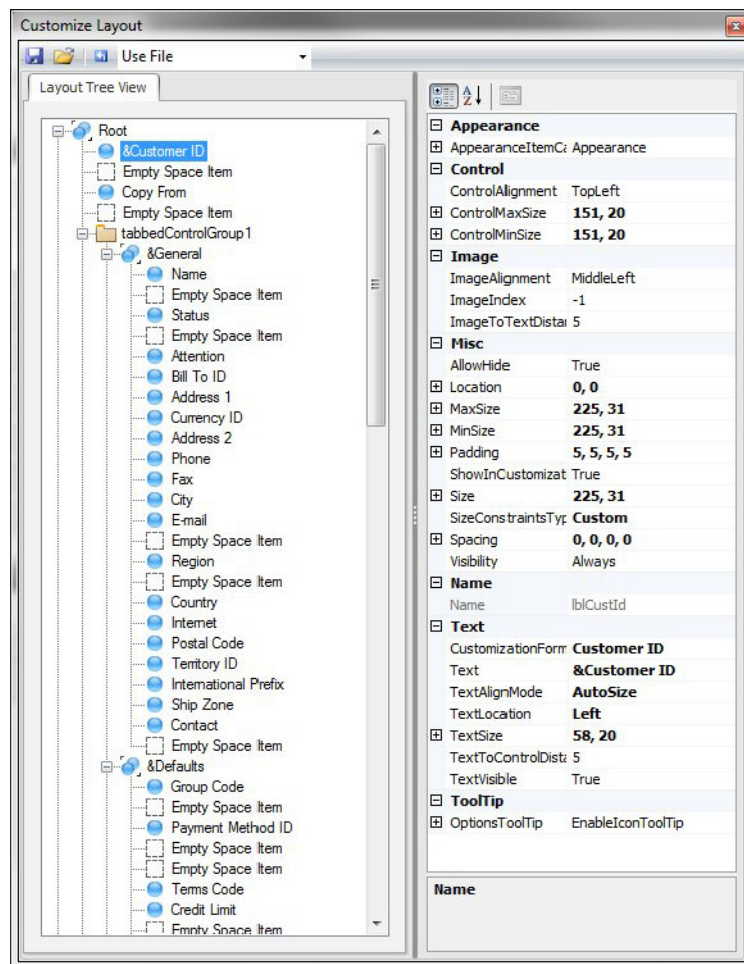
Using the Script Window, you can apply a script to a form, function, or field to modify its behavior.

NOTE: Iron Python scripting language is the recommended scripting language. You should be familiar with the scripting language prior to attempting to use scripting to change the behavior of a field.

The Properties Window

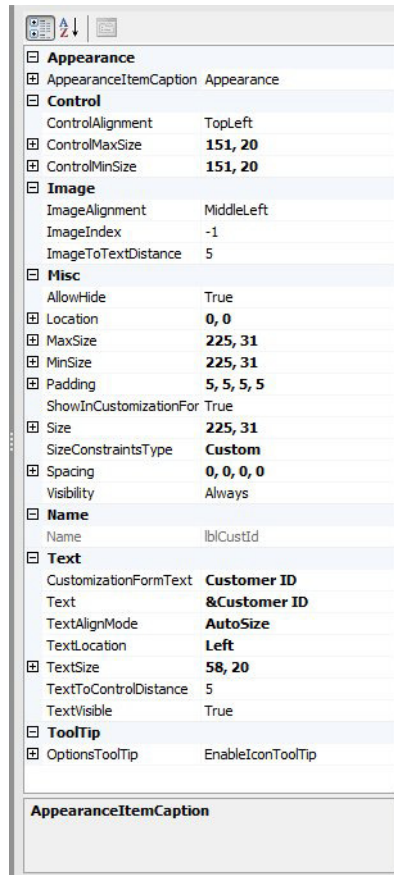
The properties of fields can be viewed and edited using the Customize Layout window.

When a Screen Layout is open with the customize layout window displayed the properties of the selected field will be displayed on the right side of the window.



A field can be selected in multiple manners: You can select the field from the layout tree view and the properties will be displayed in the properties window, or you can select the field in the design view of the form and in the customize layout window the field will be selected in the layout tree with the properties displayed.

The Properties box contains several sections to control different values within the field selected. The options for what can be changed will vary depending on the type of field that is selected.



Some of the common properties available to edit are:

Appearance: The appearance area will control things like the bgcolor of the field, the font used, font size, font color, attributes of the font such as bold, italics and underline, options used for the fonts and colors and text options.

Control: The control area will control things like the alignment of the field and the minimum and maximum size of the field.

Image: The image area will control any image alignment, indexes and distance from the label to the text field.

Misc: The miscellaneous area will control things like allowing to hide the field, location, maximum and minimum field size, padding used around the field, the field size spacing and visibility.

Text: The text area will control things like customized form text, where the text is coming from, the text align mode, text location, text size and text visibility.

ToolTip: The tool tip area will control things like the tool tip options to control things like displaying the tool tip text entered when the field was created.

USING DESIGN STUDIO

Custom Fields

The **Custom Fields** functions in TRAVERSE Design Studio allow you to create and assign fields to maintenance and transaction forms. This capability replaces the user-defined field functions found in previous versions of TRAVERSE, and expands the functionality throughout the applications.

The Custom Fields section of the TRAVERSE Design Studio contains two functions:

- **Manage**-Use the Manage Custom Fields function to create, edit, and delete custom fields.
- **Assign**-Use the Assign function to add, move, or remove custom fields in application tables. Once you assign a field to an application table, you can use the field in the related TRAVERSE function.

After you assign a custom field to an application table, you can use the tools in the Screen Layout section of the TRAVERSE Design Studio to place the field where you want it on the associated form. See the “Screen Layout” section for more details.

Create a Custom Field

To create a custom field, click **Manage** under the Custom Fields heading in the TRAVERSE Design Studio Tasks window. The Custom Fields screen appears.

The screenshot shows the 'Custom Fields' window in the TRAVERSE Design Studio. The left-hand 'Tasks' pane is expanded to show the 'Custom Fields' section, which includes 'Manage' and 'Assign' options. The main workspace displays a form for creating a new custom field. The 'Name' field is populated with 'Customer Since', the 'Description' field with 'Date of first sale', and the 'Field Type' dropdown is set to 'Date'. The window also features a 'Properties Script' pane on the right and a 'Form Layout' section in the left pane listing various pre-defined form templates.

1. Click the **New Record** button to create a new custom field.
2. Enter the name of the custom field in the **Name** field.
3. Enter a **Description** for the custom field. When you add the field to the form, the contents of the Description field appear in a pop-up box when any user hovers the mouse over the custom field in TRAVERSE.
4. Choose a **Field Type**. Depending on the field type, parameter fields will appear which allow you to further define the custom field. See “Field Types” below for a description of each of the field types.
5. Click the **Save** button. You can create another custom field by clicking the **New Record** button.

Field Types

Here are brief descriptions for each of the field types, including parameter fields required to set up the field:

Text

Use the Text field type to create a custom field that allows users to enter numbers, letters, and other characters in a form. This field type does not provide verification or have any interactive capability beyond text entry.

Parameters:

- Text Length-Choose the number of characters users may enter in this custom text field.

Number

Use the Number field type to create a field that allows users to enter only numbers within a range you specify.

Parameters:

- Maximum Value-Choose the highest allowable number for this custom field.
- Minimum Value-Choose the lowest allowable number for this custom field.

Date

Use the Date field type to create a custom field that allows users to enter or choose a date. Date fields will include a standard calendar control to help users select a date.

There are no other parameters to select for the date field type.

YesNo

Use the YesNo field type to create a check box that users can check or clear.

There are no other parameters to select for the YesNo field type.

List

Use the List field type to create a field with a list of values from which users can choose. You can enter the possible values, or use a preset Lookup ID to use values from an established TRAVERSE list field.

Parameters:

- **Lookup ID**-Select a Lookup ID if you would like your field to list values found in a table. Otherwise, leave the Lookup ID blank.
- **Display Values**-Use the Display Values field to enter specific values for the list field, separating each value with a semicolon.
- **Limit to List**-Check the Limit to List box to allow users to select only the values you have specified in the Display Values field or defined by the Lookup ID field. Leave the Limit to List box unchecked to allow users to enter their own value into the list field.

Color

Use the Color field to assign a color to the field.

Entity

Use the Entity field type to create an entity field with multiple values to choose from. You can enter the possible values for the user to choose, or use a preset Lookup ID to use values from an established TRAVERSE list field.

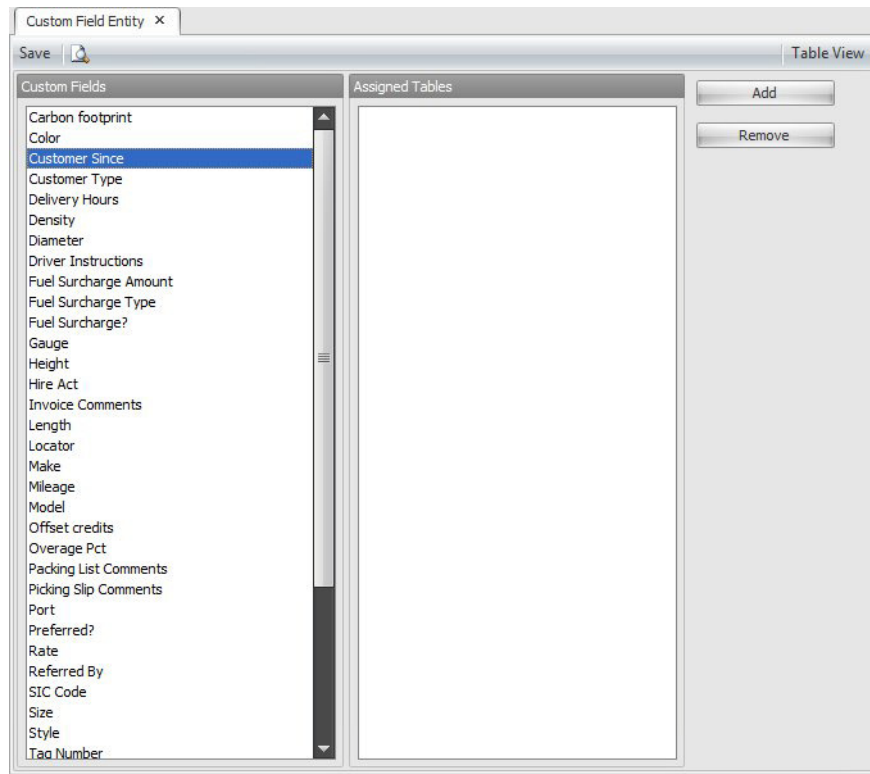
Parameters:

- **Lookup ID**-Select a Lookup ID if you would like your list field to offer the same possible values as a TRAVERSE list field. Otherwise, leave the Lookup ID blank.
- **Display Values**-Use the Display Values field to enter your own possible values for the list field, separating each value with a semicolon (;).
- **Limit to List**-Check the Limit to List box to only allow the user to select the values you have chosen in the Display Values field or defined by the Lookup ID field. Leave the Limit to List box unchecked to allow the user to enter their own value into the list field.

Assign Custom Fields to Application Tables

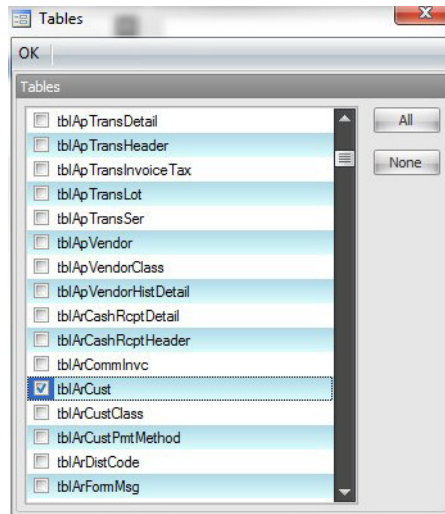
To use your custom fields in TRAVERSE functions, you must first assign them to the TRAVERSE application tables. Follow these steps to assign custom fields:

1. Click **Assign** in the Task window of TRAVERSE Design Studio.



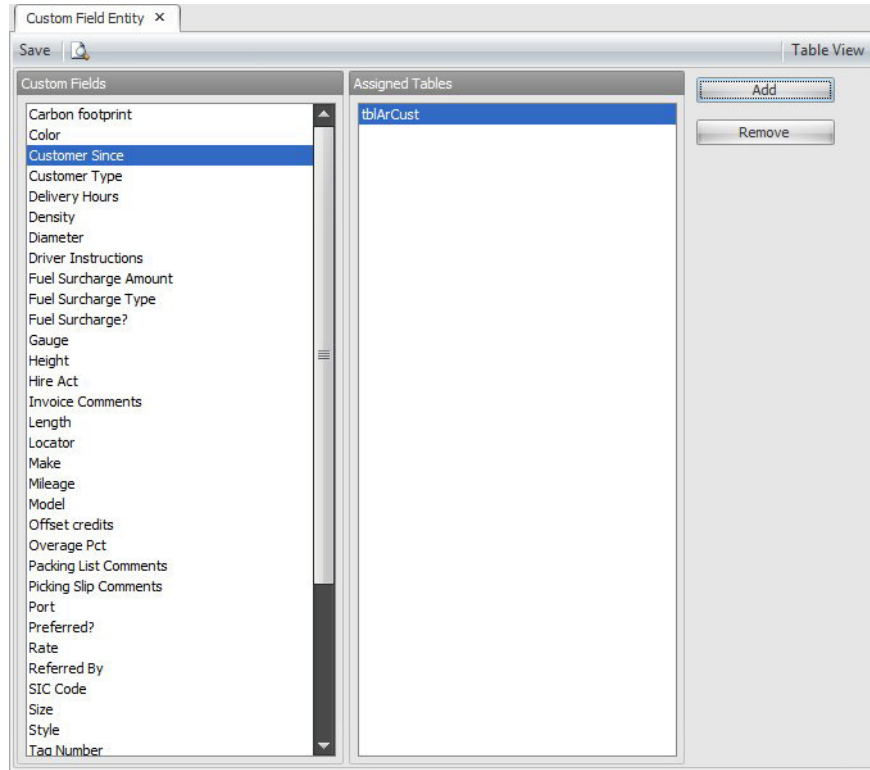
2. In the **Custom Fields** column, choose the custom field you want to assign. Click the **Add** button.

3. The **Tables** dialog box appears.



4. Check the table or tables to which you would like to assign the highlighted custom field, or click **All** to assign the custom field to all tables, and **None** to clear the check marks you've made.

5. When you finish assigning the field, click **OK** to close the Tables dialog box. In the Custom Field Entity window, the table to which you've chosen to assign the custom field appears in the **Assigned Tables** column.

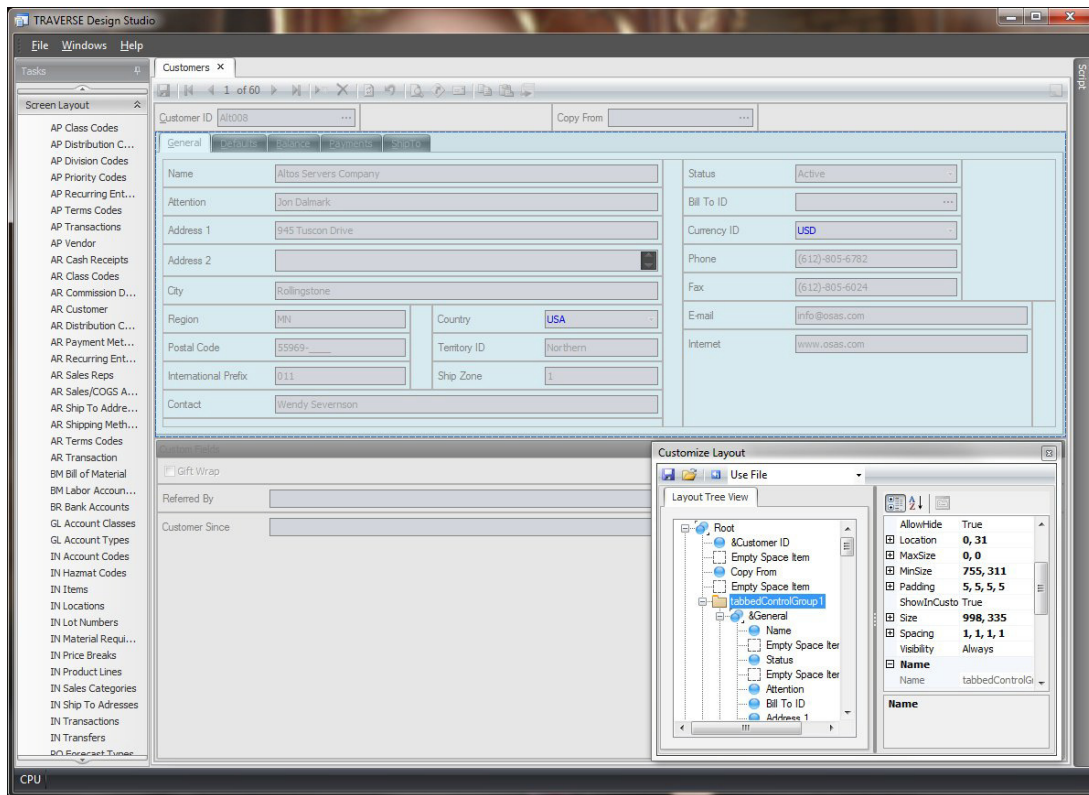


6. Click **Save** to assign the field to the tables you chose.
7. To remove a custom field from a table, highlight the table and click **Remove**.

After you assign a custom field to an application table, it appears in the associated TRAVERSE function in a section called **Custom Fields**.

Custom fields will automatically become available in the Lookup controls for key fields on forms and in the Column Chooser or Field List of any related Interactive Views and Lookup boxes, allowing you to include that field in the views and lookups as if it were one of the standard TRAVERSE fields.

For example, here is the customer form with custom fields appearing at the bottom of the screen:



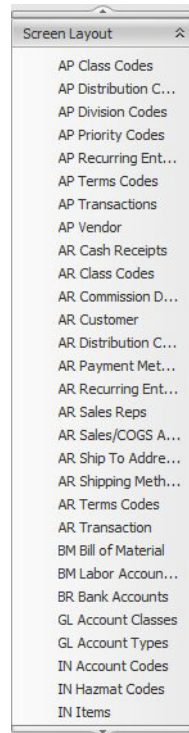
You can move the custom field to a different place on the screen, as well as implement the custom fields in a specific workstation. See the “Screen Layout” section for instructions.

Screen Layout

The TRAVERSE Design Studio Screen Layout functions allow you to relabel, reorganize, or hide fields on forms to best suit your workflow. If you add custom fields, you can also move those on the screen.

Screen Layout Menu

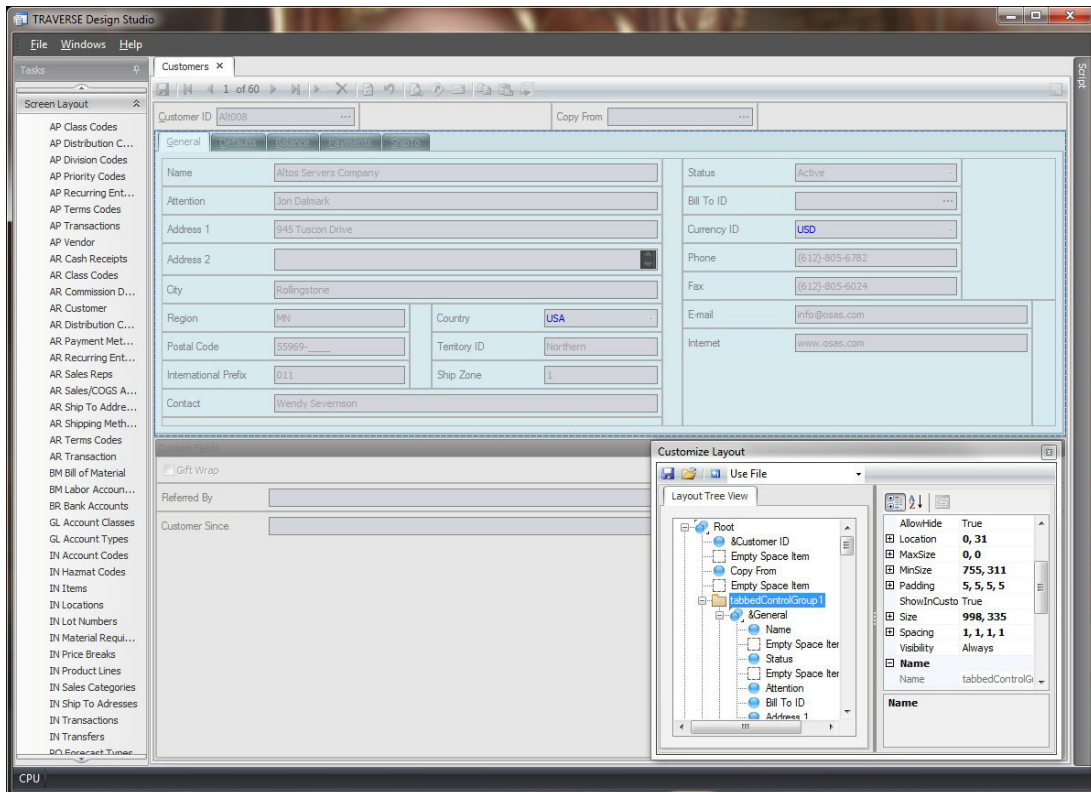
The **Screen Layout** menu in the Tasks window of TRAVERSE Design Studio lists all of the available functions for which you can adjust the field layout.



To edit a function's screen, choose that function from the list.

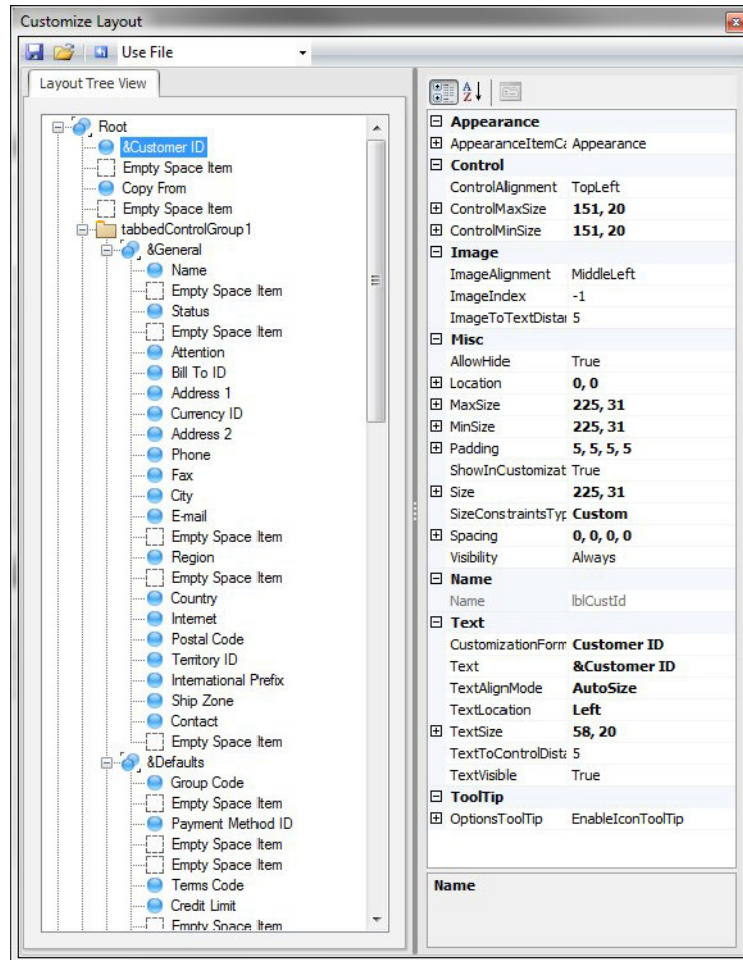
Screen Layout Editing Screen

The function you chose to edit appears in the TRAVERSE Design Studio window, and the fields will appear to be partially greyed-out. The Customize Layout dialog box appears over your chosen function.



The Customize Layout dialog box

Use the Customize Layout dialog box to perform several screen editing tasks in the Screen Layout function.

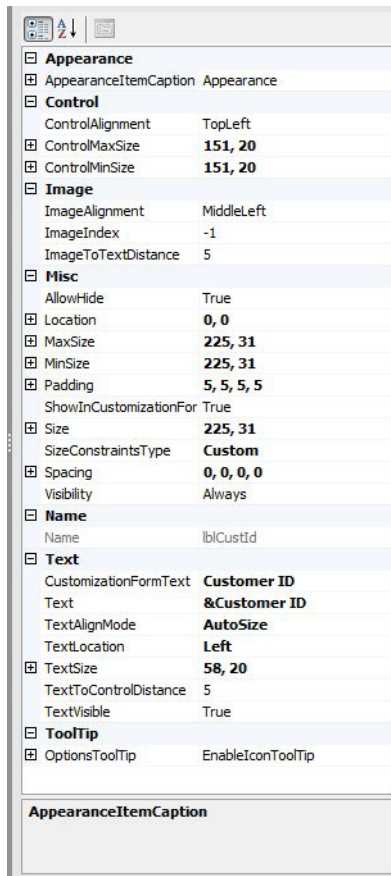


The properties of fields can be viewed and edited using the Customize Layout window.

When a Screen Layout is open with the customize layout window displayed the properties of the selected field will be displayed on the right side of the window.

A field can be selected in multiple manners: You can select the field from the layout tree view and the properties will be displayed in the properties window, or you can select the field in the design view of the form and in the customize layout window the field will be selected in the layout tree with the properties displayed.

The Properties box contains several sections to control different values within the field selected. The options for what can be changed will vary depending on the type of field that is selected.



Some of the common properties available to edit are:

Appearance: The appearance area will control things like the backcolor of the field, the font used, font size, font color, attributes of the font such as bold, italics and underline, options used for the fonts and colors and text options.

Control: The control area will control things like the alignment of the field and the minimum and maximum size of the field.

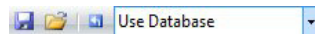
Image: The image area will control any image alignment, indexes and distance from the label to the text field.

Misc: The miscellaneous area will control things like allowing to hide the field, location, maximum and minimum field size, padding used around the field, the field size spacing and visibility.

Text: The text area will control things like customized form text, where the text is coming from, the text align mode, text location, text size and text visibility.

ToolTip: The tool tip area will control things like the tool tip options to control things like displaying the tool tip text entered when the field was created.

Use the **Save Layout** button and **Open Layout** button to save and open the screens you customize. Use the **Reset Layout** button  to return the screen to the original settings. Because you can return the screen to the original state at any time by clicking the **Reset Layout** button, you can experiment with the screen layout safely. The **Use** field allows you to save your layout to the Database or a File. To open an existing screen layout saved to the database select Use Database from the selection box and then click on the Open button. The layout saved with the default description will be opened.



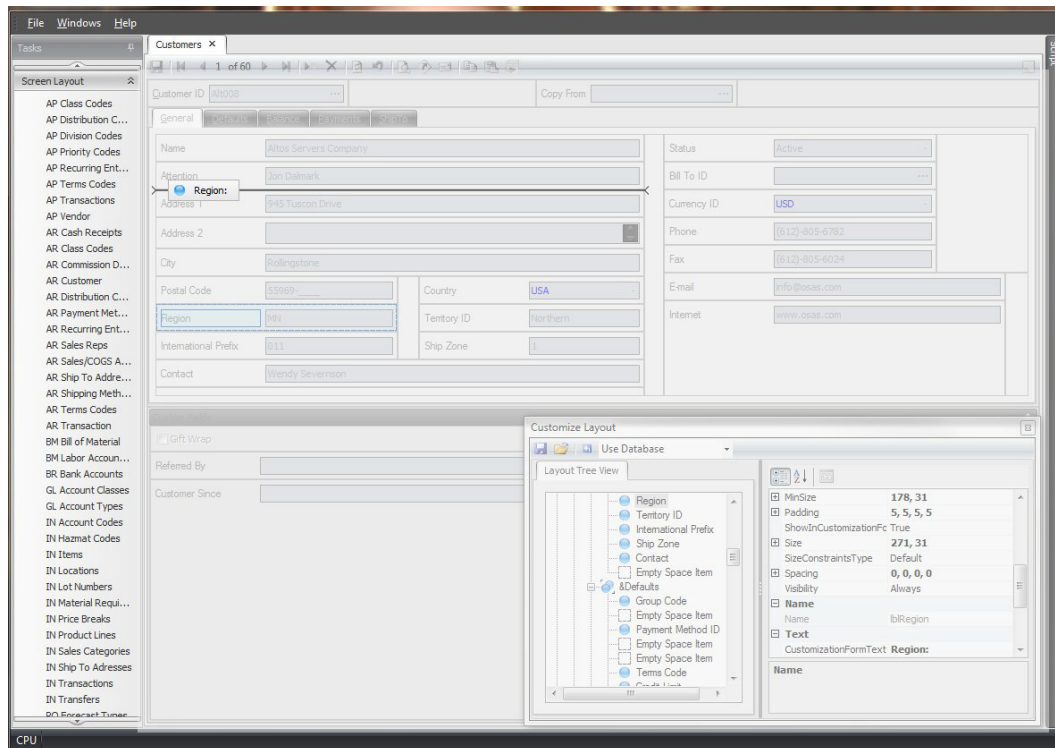
When saving to the database, by selecting Use Database, you are saving the changes you made and you can assign the new screen layout to users with the Custom Layout function in the System Data menu.

When saving to a file, by selecting Use File, you are saving your changes to a file to allow you to open the file to make further changes and then save to the database when satisfied with the layout.

The Customize Layout dialog box also contains a Layout Tree View of the fields on the screen. In addition to the drag-and-drop method of moving fields described below, you can grab and move fields within the Tree View to alter the screen.

Move a Field Within a Screen

To move a field within a screen, simply point to the field, hold down the mouse button, and drag it to the desired location on the screen. In the illustration below, the **Region** field will be positioned between the **Attention** and **Address 1** fields.

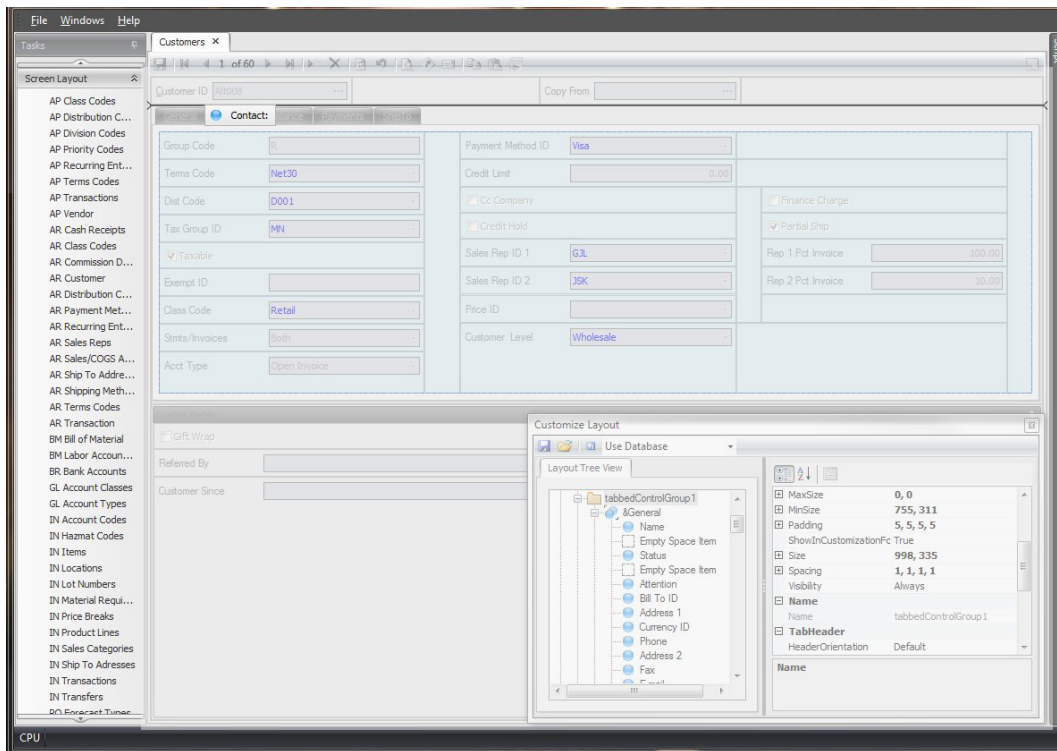


When you release the mouse button, the field is placed in the selected position, and the other fields are adjusted accordingly.

NOTE: If there is no data in the company database you will not be allowed to move the fields by dragging and dropping on the screen layout screen. Once there is data in the database and table the field is assigned to you will then be allowed to move the fields around on the screen.

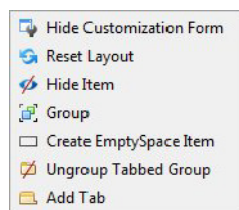
Move a Field from One Tab to Another

To move a field from one screen tab to another, point to the field you want to move, hold down the mouse button, and drag it to the tab heading on the screen to which you want to move the field. The associated tab layout appears, on which you can then place the field.



Test a Screen Layout Within TRAVERSE Design Studio

Once you have finished making field changes to the screen, you can test the functionality of the rearranged screen by closing the Customize Layout dialog box. You also can get to the test mode by right clicking in the screen layout and selecting Hide Customization Form.



The screen appears to function as it will within TRAVERSE, displaying your company data.

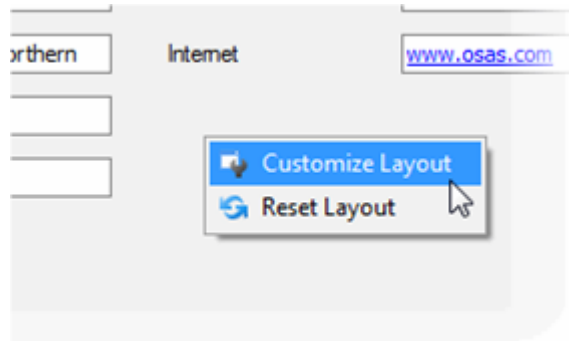
The screenshot displays the 'Customers' window in the TRAVERSE Design Studio. The window has a title bar 'Customers x' and a standard toolbar. Below the toolbar, there is a 'Customer ID' field with the value 'Alt008' and a 'Copy From' field. The main area is divided into two tabs: 'General' (selected) and 'Ship To'. The 'General' tab contains various fields for company information, organized into two columns. The 'Custom Fields' section at the bottom includes a 'Customer Since' date field, a 'Gift Wrap' checkbox, and a 'Referred By' field.

General	
Name	Altos Servers Company
Attention	Jon Dalmark
Address 1	945 Tuscon Drive
Address 2	
City	Rollingstone
Region	MN
Country	USA
Postal Code	55969-_____
International Prefix	011
Contact	Wendy Severson
Status	Active
Bill To ID	
Currency ID	USD
Phone	(612)-805-6782
Fax	(612)-805-6024
E-mail	info@osas.com
Internet	www.osas.com

Custom Fields	
Customer Since	12/1/2010
Gift Wrap	☐
Referred By	

NOTE: Changes made to your company data while testing screens within TRAVERSE Design Studio are not written to the database, and therefore will not affect the company data.

To return to the Customize Layout dialog box, right-click anywhere within the testing screen and choose **Customize Layout**.

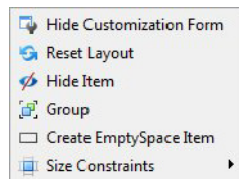


The Customize Layout dialog box reappears, and you can continue making changes to the screen. You can also right-click anywhere in the screen and choose Reset Layout to return the form to its original state.

Right Click Menus

Changes can be made to the design of the form or some of the properties of a selected field using right click menus. The right click menu will vary depending on the area of the form you right click or the type of field you have selected when right clicking.

Right Click Blank Area



The menu selections available are:

Hide Customization Form: This selection will hide the design view of the form and put you to the test mode of the screen layout.

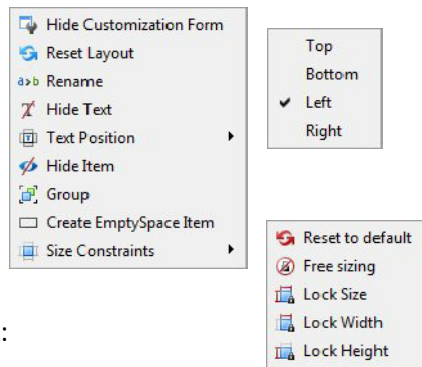
Reset Layout: This selection will reset the screen layout back to the default layout with no fields moved or properties modified.

Hide Item: This selection will hide the selected blank area from the screen layout.

Group: This selection will group multiple fields selected.

Create Empty Space Item: This selection will add an empty space item to the form to place where you want it.

Right Click Field Selected



The menu selections available are:

Hide Customization Form: This selection will hide the design view of the form and put you to the test mode of the screen layout.

Reset Layout: This selection will reset the screen layout back to the default layout with no fields moved or properties modified.

Rename: This selection will allow you to rename the field label for the selected field.

Hide Text: This selection will allow you to hide the field label to display the field only.

Show Text: This selection will be visible if you have selected Hide Text to hide the field label. Selecting Show Text will display the field label again.

Text Position: This selection will allow you to change the text position within the field. You can place the text on the Top, Bottom, Left or Right.

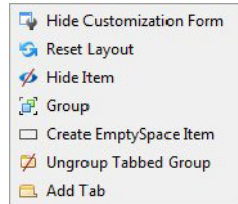
Hide Item: This selection will hide the selected field from the screen layout.

Group: This selection will group multiple fields selected.

Create Empty Space Item: This selection will add an empty space item to the form to place where you want it.

Size Constraints: This selection will allow you to set the size constraints on the selected field. The size constraints that can be selected are: Reset to Default, Free Sizing, Lock Size, Lock Width and Lock Height.

Right Click Tab Form Section



The menu selections available are:

Hide Customization Form: This selection will hide the design view of the form and put you to the test mode of the screen layout.

Reset Layout: This selection will reset the screen layout back to the default layout with no fields moved or properties modified.

Hide Item: This selection will hide the selected tab from the screen layout.

Group: This selection will group multiple fields selected.

Create Empty Space Item: This selection will add an empty space item to the form to place where you want it.

Ungroup Tabbed Group: This selection will ungroup any tabs that have been grouped together within an area on the screen. For example: in the Customers screen layout the tabs are grouped. Selecting this will eliminate the tabs and form different sections on the screen with each field from the tabs in a section of the screen layout. Be careful when selecting this selection, the only way to get the tabs back is to reset the layout.

Add Tab: This selection will allow you to add a tab to the form. The new tab will automatically be grouped with the other tabs on the form, if they are grouped. You can then name the tab and place custom fields, or move current fields to this new tab.

Add Tabbed Group: This selection will allow you to group tabs that have been added to the screen layout, if they currently are not grouped. This selection will only be visible if you do not currently have a tab group.

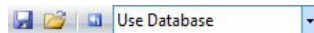
Save a Screen Layout

Once you are satisfied with the screen layout, you can save it for use within TRAVERSE on one or more workstations. You can save the layout as an.FRM file that can be sent and imported into different databases, or save the layout in your current database which will over-write the previous layout in the system.

NOTE: If you save the layout to the database, all users who use the database will use the same layout for that function, if you do not assign the layout to a user or group using the System Data, Custom Data function.

Save a screen layout to a file

If you choose to **Use File**, a dialog box will open when you click the **Save** button. This will allow you to save the form file. If you choose **Use Database**, the customizations will be saved into your database when you click the **Save** button.



1. Select **Use File** in the **Customize Layout** dialog box.

Saving a to a file is mainly used as a backup for your screen layout changes. You can save the file, present the new screen to the client to see if it needs modification. If the screen needs to be changed you can then open the saved file and modify it and save the file again.

2. Click the **Save** button on the Customize Layout dialog box.
3. Choose a location to save the **.FRM** file which contains the modified layout.
4. After you have saved the file, copy the **.FRM** file to the TRAVERSE root folder on each workstation you want to modify. When you subsequently open the function on the workstation, the modified screen will appear.

Save a screen layout to the database

1. Select **Use Database** in the **Customize Layout** dialog box.
2. Click the **Save** button on the Customize Layout dialog box. The layout will replace the current layout in your database, and will be used for that screen by all users who use the database.

NOTE: Use the Custom Layout function in the System Data to assign the Screen Layout to specific users you want to see the modified screen. If you don't assign to specific users, ALL users will see the new screen layout.

Deployment

You may want certain users to work with different screen layouts; perhaps hiding certain fields for some users or changing the layout to optimize it for a user's role.

There are multiple ways to deploy the changes to the workstation.

You can save the screen layout to file and copy the file to workstations as described below.

For example, the default file name for a customized Customer screen is CustomerControl.FRM. You could change the name of the file when you save it to CustomerControl_Acct.FRM for an accounting user, then make other changes to the form and save it again using the name CustomerControl_Sale.FRM for a user in sales.

After saving the files, simply copy the appropriate file to each user's workstation, renaming it to the standard CustomerControl.FRM name on that user's workstation.

NOTE: You must rename the .FRM form back to the original name after copying to the client directory, or the TRAVERSE client will not recognize the changes.

The second alternative to deploying the screen layout changes is to save the screen layout to a file and the database and assign the screen to users. Modify the screen for other users, save to a file with a different name (for backup purposes) and save the screen to the database again and assign that screen layout to users. Do this process for the different screen layouts you want for different users.

Customize the Lookups and Interactive Views

The TRAVERSE Design Studio includes functions to edit the Lookup and (Interactive) View definitions. However, if you just change one of the standard definitions, it is possible that those changes will be overwritten by the automated update process.

There is a specific method to use to customize Lookup or View definitions so that they will not be overwritten by our updates. The example below references a Lookup definition, but the process is the same for View definitions.

Copy the definition that you want to change to a **new** record using the Lookup function of Design Studio. **Create an ID to a unique value.** In the example below we copied the PaEmployee ID to a new record and called it PaEmployeeCustom. In the new record, put the ID of the original record into the “Replace ID” column. Now make your changes to this new definition. The lookup function(s) that use the PaEmployee ID will now use the custom lookup, and our update process will not overwrite the custom entry in the lookup table.

ID	Description	Data Source	Key Column	Default Columns	Enumerated Columns	Formats	Database Type	Volatile Data	Replace ID
PaDepartmentId	PA Department Id	SELECT Id AS DeptId, DepartmentName FROM [dbo].[trav_tbPaDept_view] ORDER BY DeptId	DeptId	DeptId, DepartmentName			Payroll	☐	
PaDeptAllocation	PA Department Allocation	SELECT Id, Descr, ExpDate FROM dbo.tbPaAllocDeptHdr ORDER BY Id	Id, Descr, ExpDate				Payroll	☐	
PaEamCode	Earning codes	SELECT Id AS EarningCode, EarningType FROM dbo.tbPaEamCode ORDER BY EarningCode	EarningCode, EarningType				Payroll	☐	
PaEmployee	Pa Employees	SELECT s.EmployeeId, LastName, FirstName, AddressLine1, AddressLine2, ResidentCity, ResidentState, ZipCode, TerminationDate (CF) FROM dbo.trav_tbSmEmployee_view s INNER JOIN dbo.tbPaEmployee p on s.EmployeeId = p.EmployeeId	EmployeeId, LastName, FirstName, AddressLine1, AddressLine2, ResidentCity, ResidentState, ZipCode				Company	☐	
PaEmployeeCheck	PA Employees with ...	SELECT DISTINCT s.EmployeeId, LastName, FirstName, AddressLine1, AddressLine2, ResidentCity, ResidentState, ZipCode FROM dbo.trav_tbSmEmployee_view s INNER JOIN dbo.tbPaEmployee p on s.EmployeeId = p.EmployeeId	EmployeeId, LastName, FirstName, AddressLine1, AddressLine2, ResidentCity, ResidentState, ZipCode				Company	☐	
PaFormula	Payroll Formula Look...	select Id, [Description] from dbo.tbPaFormulaHeader ORDER BY Id	Id, Description				Payroll	☐	
PaLeaveCode	Leave codes	SELECT Id AS LeaveCode, [Description] FROM [dbo].[tbPaLeaveCodeHdr] ORDER BY LeaveCode	LeaveCode, Description				Payroll	☐	
PaEmployeeCustom	Custom Pa Employee...	SELECT s.EmployeeId, LastName, FirstName, AddressLine1, AddressLine2, ResidentCity, ResidentState, ZipCode, TerminationDate (CF) FROM dbo.trav_tbSmEmployee_view s INNER JOIN dbo.tbPaEmployee p on s.EmployeeId = p.EmployeeId	EmployeeId, LastName, FirstName, AddressLine1, AddressLine2, ResidentCity, ResidentState, ZipCode				Company	☐	PaEmployee

System Data

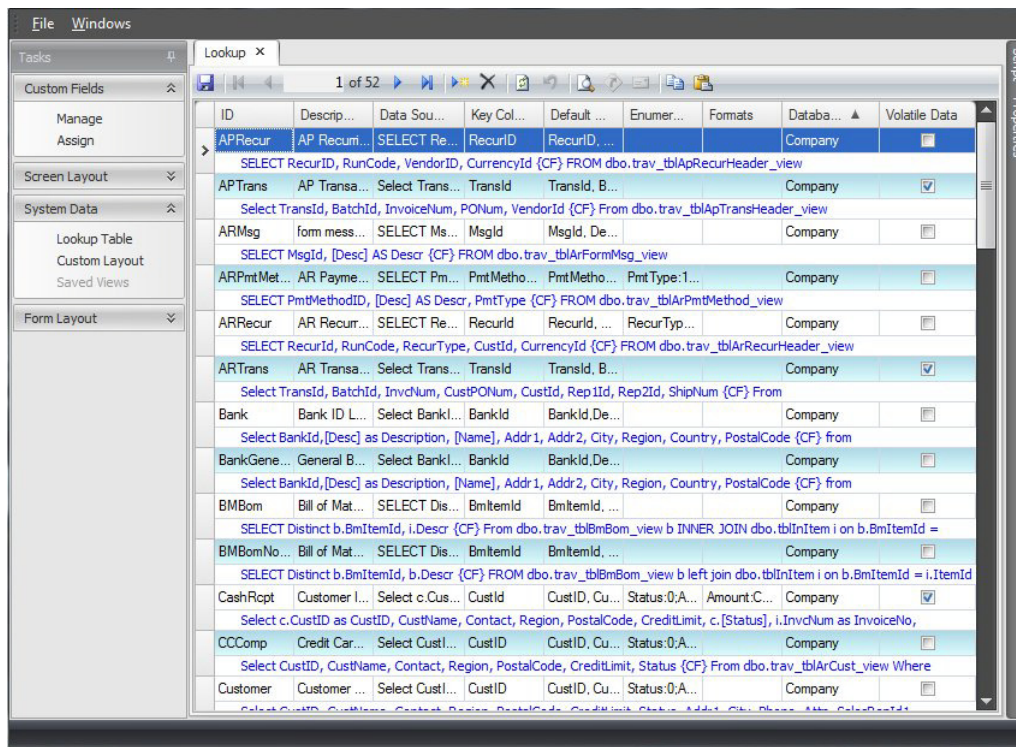
Lookup Table

Use the System Data, Lookup Table function to configure the information shown when you use the Lookup command (F4) within TRAVERSE without making changes directly to SQL Server tables.

Use the Lookup Table function to define column heading names, whether you want a field to appear as search criteria, the order in which items appear in the Look By field, the order in which fields appear in the results area of the Search screen, and so on.

NOTE: Do not modify these fields unless you have knowledge of database design and administration.

To define what appears when you use the Lookup command, select Lookup Table from the System Data menu in TRAVERSE Design Studio. The Lookup screen appears.



Modify the lookup data as needed.

NOTE: Changes are not recommended to any columns except the Data Source and Default Columns. Changes to other columns may cause the lookup to not function properly.

The column definitions are:

ID

The Lookup ID used by the programs within TRAVERSE to open the lookup boxes.

Description

The actual column heading name in the results section of the Search screen appears in the Description.

Data Source

The SQL query that the system uses to retrieve the data for this lookup ID.

Key Column

Defines the leading column in the lookup display, independent of user input or changes.

Default Columns

Names of the columns that you want displayed by default, independent of user input or changes.

Enumerated Columns

Provides an alternative description of the numeric values stored in a table. For example, TRAVERSE stores the inventory item status as 1, 2, 3 or 4. You may want to view these values as Active, Discontinued, Superseded, and Obsolete.

Formats

Use to format information that appears in the results section of the Search screen. For example, the social security format is &&&-&&&-&&&.

Database Type

Describes the database from which that lookup's information is taken (e.g. Company, System, Payroll).

Volatile Data

Indicates that the data is constantly changing. Used for the transactions screens, where transaction records are being added and taken away often.

Replace ID

Use the replace ID function to indicate the original Lookup ID you used when copying an existing ID to a new record. If you are going to make changes to one of the lookup ID's you should use this method to modify it so updates will not overwrite your changes. The replace ID will indicate which search box will be used by your new modified Lookup ID. See *Customize the Lookups and Interactive Views in TRAVERSE Edit* (page 3-29) for more details.

When you finish making changes to the Lookup table, click the Save button to save your changes. Close the Lookup screen without saving to abandon changes.

Views Table

Use the System Data, Views Table function to configure the information shown in the interactive views within TRAVERSE without making changes directly to SQL Server tables.

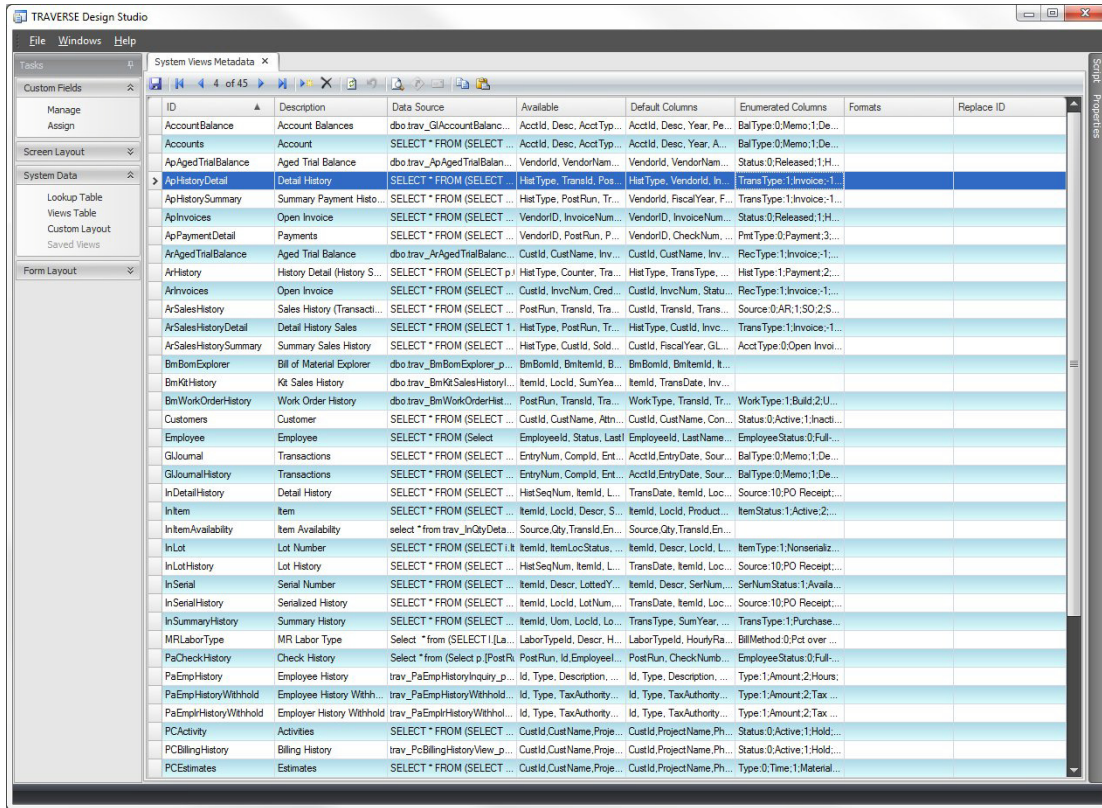
Use the Views Table function to define columns available for the view, whether you want a field to appear as filter criteria, any joins and grouping that need to be made when pulling data from multiple tables, the default fields to display when opening the view and resetting the view once it has been changed within TRAVERSE, the order in which fields appear in the results area, any enumerated values that need to be translated, and so on.

NOTE: Do not modify these fields unless you have knowledge of database design and administration.

DESIGN STUDIO

Using Design Studio

To define what appears and is available when interactive views are opened, select Views Table from the System Data menu in TRAVERSE Design Studio. The Views Table screen appears.



Modify the lookup data as needed.

NOTE: Changes are not recommended to any columns except the Data Source, Available, and Default Columns. Changes to other columns may cause the view to not function properly.

The column definitions are:

ID

The name assigned to the view.

Description

The description of what the view shows.

Data Source

The SQL query that the system uses to retrieve the data for this view ID.

```
SELECT * FROM (SELECT CASE p.[EntryNum] WHEN -1 THEN 4 WHEN -2 THEN 5 WHEN -3 THEN 6 ELSE 3 END AS [HistType], h.[PostRun], h.[TransId], h.[InvoiceNum], h.[BatchId], h.[VendorId], h.[InvoiceDate], h.[TransType], h.[PONum], h.[DistCode], h.[TermsCode], h.[DueDate1], h.[DueDate2], h.[DueDate3], SIGN(h.[TransType]) * h.[PmtAmt1] AS [PmtAmt1], SIGN(h.[TransType]) * h.[PmtAmt2] AS [PmtAmt2], SIGN(h.[TransType]) * h.[PmtAmt3] AS [PmtAmt3], SIGN(h.[TransType]) * h.[Subtotal] AS [Subtotal], SIGN(h.[TransType]) * h.[SalesTax] AS [SalesTax], SIGN(h.[TransType]) * h.[Freight] AS [Freight], SIGN(h.[TransType]) * h.[Misc] AS [Misc], SIGN(h.[TransType]) * h.[CashDisc] AS [CashDisc], SIGN(h.[TransType]) * h.[PrepaidAmt] AS [PrepaidAmt], h.[CurrencyId], h.[ExchRate], h.[PmtCurrencyId], h.[PmtExchRate], SIGN(h.[TransType]) * h.[PmtAmt1Fgn] AS [PmtAmt1Fgn], SIGN(h.[TransType]) * h.[PmtAmt2Fgn] AS [PmtAmt2Fgn], SIGN(h.[TransType]) * h.[PmtAmt3Fgn] AS [PmtAmt3Fgn], SIGN(h.[TransType]) * h.[SubtotalFgn] AS [SubtotalFgn], SIGN(h.[TransType]) * h.[SalesTaxFgn] AS [SalesTaxFgn], SIGN(h.[TransType]) * h.[FreightFgn] AS [FreightFgn], SIGN(h.[TransType]) * h.[MiscFgn] AS [MiscFgn], SIGN(h.[TransType]) * h.[CashDiscFgn] AS [CashDiscFgn], SIGN(h.[TransType]) * h.[PrepaidAmtFgn] AS [PrepaidAmtFgn], h.[CheckNum], h.[CheckDate], h.[PostDate], h.[GLPeriod], h.[FiscalYear], h.[Ten99InvoiceYN], h.[Status], h.[Notes], h.[TaxGripID], h.[TaxableYN], h.[TaxClassFreight], h.[TaxClassMisc], h.[TaxAdjClass], h.[TaxAdjLocID], h.[Source], h.[GLAcctAP], h.[GLAcctFreight], h.[GLAcctTaxAdj], h.[GLAcctMisc], h.[DiscDueDate], SIGN(h.[TransType]) * h.[Taxable] AS [Taxable], SIGN(h.[TransType]) * h.[NonTaxable] AS [NonTaxable], SIGN(h.[TransType]) * h.[TaxableFgn] AS [TaxableFgn], SIGN(h.[TransType]) * h.[NonTaxableFgn] AS [NonTaxableFgn], SIGN(h.[TransType]) * h.[TaxAdjAmt] AS [TaxAdjAmt], SIGN(h.[TransType]) * h.[TaxAdjAmtFgn] AS [TaxAdjAmtFgn], p.[EntryNum], p.[PartId], p.[PartType], p.[WhseId], p.[Desc], SIGN(h.[TransType]) * p.[Qty] AS [Qty], SIGN(h.[TransType]) * p.[QtyBase] AS [QtyBase], p.[Units], p.[UnitsBase], p.[UnitCost], p.[UnitCostFgn], SIGN(h.[TransType]) * p.[ExtCost] AS [ExtCost], SIGN(h.[TransType]) * p.[ExtCostFgn] AS [ExtCostFgn], p.[GLDesc], p.[AddnlDesc], p.[TaxClass], p.[ConversionFactor], p.[LottedYN], p.[GLAcct], p.[GLAcctWIP], p.[GLAcctSales], p.[TransHistId], p.[CustomerId], p.[JobId], p.[ProjName], p.[PhaseId], p.[PhaseName], p.[TaskId], p.[TaskName], p.[UnitInc], SIGN(h.[TransType]) * p.[ExtInc] AS [ExtInc], p.[ProjItemYN], p.[CostType], p.[LineSeq], s.[Name], s.[PriorityCode], s.[VendorHoldYN], s.[VendorClass], s.[DivisionCode], s.[Status] VendorStatus (CF) FROM [(COMP)]..dbo.tblApHistHeader h LEFT JOIN [(COMP)]..dbo.tblApHistDetail_v p ON h.[PostRun] = p.[PostRun] AND h.[TransId] = p.[TransId] AND h.[InvoiceNum] = p.[InvoiceNum] LEFT JOIN [(COMP)]..dbo.tblApVendor s ON h.[VendorId] = s.[VendorId] ds
```

The script displayed in this window may be edited to include new columns, columns from tables not currently in the view, or other programming that may be changed for the view.

Available

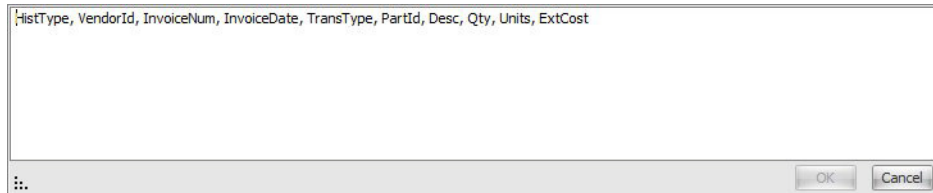
Defines columns available when doing server filters and also when the Column Chooser is selected.

```
[HistType, TransId, PostRun, VendorId, Notes, InvoiceNum, InvoiceDate, TransType, EntryNum, PartId, Desc, AddnlDesc, PartType, WhseId, Qty, Units, UnitCost, UnitCostFgn, ExtCost, ExtCostFgn, GLAcct, JobId, DistCode, PONum, GLPeriod, FiscalYear, Name, PriorityCode, VendorHoldYN, VendorClass, DivisionCode, VendorStatus, ConversionFactor, LottedYN, CustomerId, ProjName, PhaseId, PhaseName, TaskId, TaskName, UnitInc, ExtInc]
```

From the columns selected in the Data Source script you can list the columns that will be available for server filtering and the column chooser.

Default Columns

Names of the columns that you want displayed by default, independent of user input or changes. When you click the Reset button in the Interactive View these columns will be on the screen when the view refreshes.



HistType, VendorId, InvoiceNum, InvoiceDate, TransType, PartId, Desc, Qty, Units, ExtCost

OK Cancel

Enumerated Columns

Provides an alternative description of the numeric values stored in a table. For example, TRAVERSE stores the inventory item status as 1, 2, 3 or 4. You may want to view these values as Active, Discontinued, Superseded, and Obsolete.

Formats

Use to format information that appears in the results section of the Search screen. For example, the social security format is &&&-&&&-&&&.

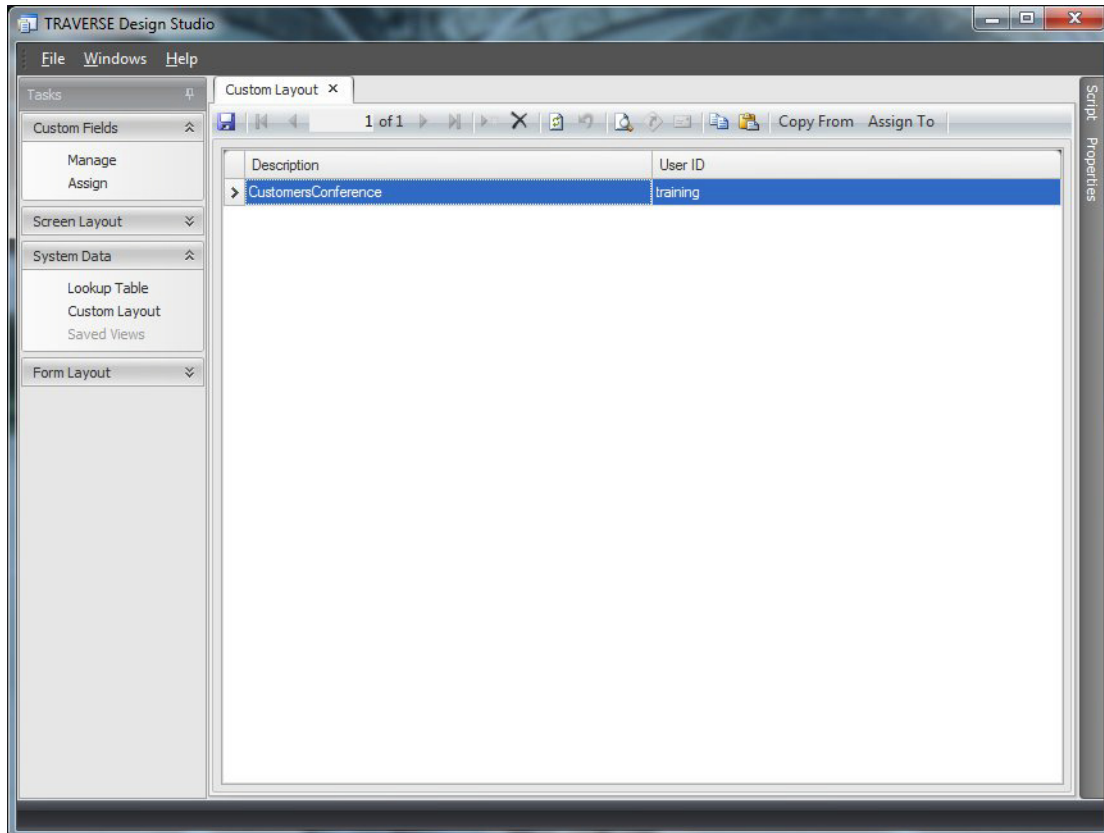
Replace ID

Use the replace ID function to indicate the original View ID you used when copying an existing ID to a new record. If you are going to make changes to one of the view ID's you should use this method to modify it so updates will not overwrite your changes. The replace ID will indicate which view will be used by your new modified View ID. See Customize the Lookups and Interactive Views in TRAVERSE Edit (page 3-29) for more details.

When you finish making changes to the Views table, click the Save button to save your changes. Close the Views screen without saving to abandon changes.

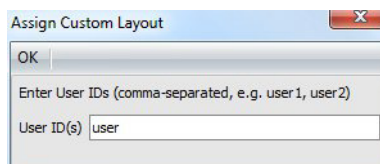
Custom Layout

Use the System Layout, Custom Layout to assign custom screen layouts to users, when the screen layout has been saved to database.

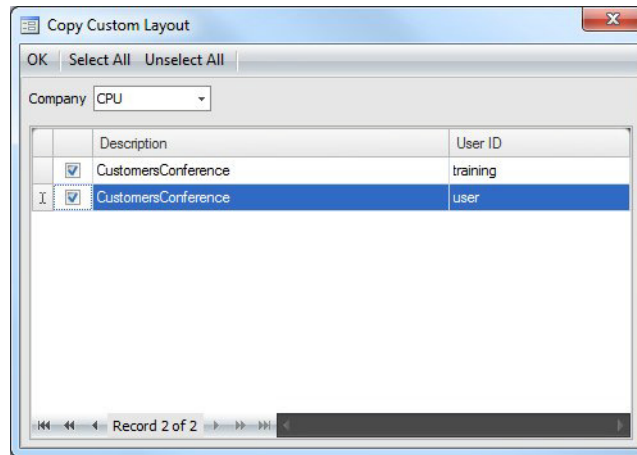


You will see any screen layouts that have been saved using the Use Database function in the screen layout function.

1. Edit the description to describe the user or group of users you want to assign the screen to.
2. Click **Assign To** to assign the screen to users or groups(roles).



3. Enter the user ID or role name you want the selected custom screen layout assigned to. Separate the users or roles with a comma (,).
4. Click **OK** once you are finished with the user or role assignment.
5. Click **Copy From** to copy the custom screen assignment from one company to another.



6. Select the **Company** to copy from and the **Description** and **User ID** list will appear, for screen layouts saved to that company.
7. Check the box next to the screens and users you want to copy to the company you are currently attached to in design studio.
8. Select **Select All** or **Unselect All** to check or uncheck all boxes.
9. Click **OK** to save the copy from one company to your current company.

Business Rules

The business rules function will display all the business rules available in TRAVERSE. The main purpose of this function is to make it easier for developers to add their own new business rules.

The business rules you see listed are coming from the tblSmConfig and tblSmConfigDep tables in the SYS database. This list of business rules will be available for all companies.

NOTE: ***Warning***** We do not recommend editing existing rules!!!**

Ref	Config ID	Da...	App ID	Pa...	Gr...	Display S...	Caption	Lo...	Default V...	Max Char...	Value List	Visible	Role Con...	Required	Min Value	Max Value
4	AppInterface	2048	AP	356		10	AppInterface			0		☒	☐	☒		
6	PeriodicSetup	2048	AP	356		30	PeriodicSetup			0		☒	☐	☒		
7	Transaction...	2048	AP	356		50	Transaction...			0		☒	☐	☒		
8	GLDefault	2048	AP	356		60	GLDefault			0		☒	☐	☒		
9	GIYn	2048	GL	357		10	InterfaceGL			0		☒	☐	☒		
10	INYn	2048	IN	357		20	InterfaceIn			0		☒	☐	☒		
11	BrYn	2048	BR	357		30	InterfaceBr			0		☒	☐	☒		
12	JcYn	2048	JC	357		40	InterfaceJc			0		☒	☐	☒		
13	PaYn	2048	PA	357		50	InterfacePa			0		☒	☐	☒		
14	ApYn	2048	AP	357		60	InterfaceAp			0		☒	☐	☒		
15	SoYn	2048	SO	357		70	InterfaceSo			0		☒	☐	☒		
16	PoYn	2048	PO	357		80	InterfacePo			0		☒	☐	☒		
17	MpYn	2048	MP	357		90	InterfaceMp			0		☒	☐	☒		
18	SdYn	2048		357		100	InterfaceSd			0		☒	☐	☒		
19	ArYn	2048	AR	357		110	InterfaceAr			0		☒	☐	☒		
20	BmYn	2048	BM	357		120	InterfaceBm			0		☒	☐	☒		
21	MrYn	2048	MR	357		130	InterfaceMr			0		☒	☐	☒		
22	BatchYn	2048		357		140	UseBatch			0		☒	☐	☒		
23	PostWoPrint...	2048		357		150	PostWoPrint...			0		☒	☐	☒		
24	PostDtlYn	2048		357		160	PostDtl			0		☒	☐	☒		
25	MiscOpt	2048	AP	356		0	MiscOpt			0		☒	☐	☒		
27	GIYn	1024	AP	4 9		10	GL		0	0	Ikp TrueFalse	☒	☐	☒		
28	INYn	1024	AP	4 10		20	IN		1	0	Ikp TrueFalse	☒	☐	☒		
29	BrYn	1024	AP	4 11		30	BR		1	0	Ikp TrueFalse	☒	☐	☒		
30	JcYn	1024	AP	4 12		40	JC		1	0	Ikp TrueFalse	☐	☐	☒		
35	PostWoPrint...	1024	AP	25 23		10	PostWoPrint...		0	0	Ikp TrueFalse	☒	☒	☒		
36	BatchYn	1024	AP	25 22		20	UseBatch		1	0	Ikp TrueFalse	☒	☐	☒		
37	PostDtlYn	1029	AP	25 24		30	PostDtl		1	0	Ikp TrueFalse	☒	☐	☒		
38	ShowTen99...	1024	AP	25		40	Use1099		0	0	Ikp TrueFalse	☒	☐	☒		

The column definitions are:

Ref

The reference ID in the tblSmConfig table.

Config ID

The configuration ID for what the business rule does.

Data Type

The kind of data that will be input into the business rule. Select the data type from the list of available data types.

App ID

The application the business rule will appear in.

Parent Ref

The reference to the parent config ID value. For example, 4 is the parent reference for the AP interfaces to other apps.

Group Ref

The reference to the group of rule types. For example, 558 is the reference for the audit group.

Display Seq

The order it will appear in the list of business rules for the application.

Caption

The caption ID for what will appear on the screen for the business rule.

Lookup ID

The lookup Id to use if a selection needs to be made from a list of records. For example, for the default GL accounts the lookup ID would be GLAccounts.

Default Value

If there is a default value for the business rule, what should the default value be.

Max Characters

What is the maximum number of characters allowed in the field for the business rule.

Value List

The function that should be used to retrieve a list of values.

Visible

Should the business rule be displayed on the screen.

Role Configurable

Is the business rule used in the role specific application of the business rule. Can it be applied to specific groups to act differently for different groups within TRAVERSE.

Required

Is it required to have a value in the business rule when saving and exiting the business rules.

Min Value

If the business rule requires a number to be put in, what is the minimum value accepted.

Max Value

If the business rule requires a number to be put in, what is the maximum value accepted.

NOTE: Only experienced programmers familiar with TRAVERSE programming and with knowledge of how the Business Rule configuration tables are related should use this function.

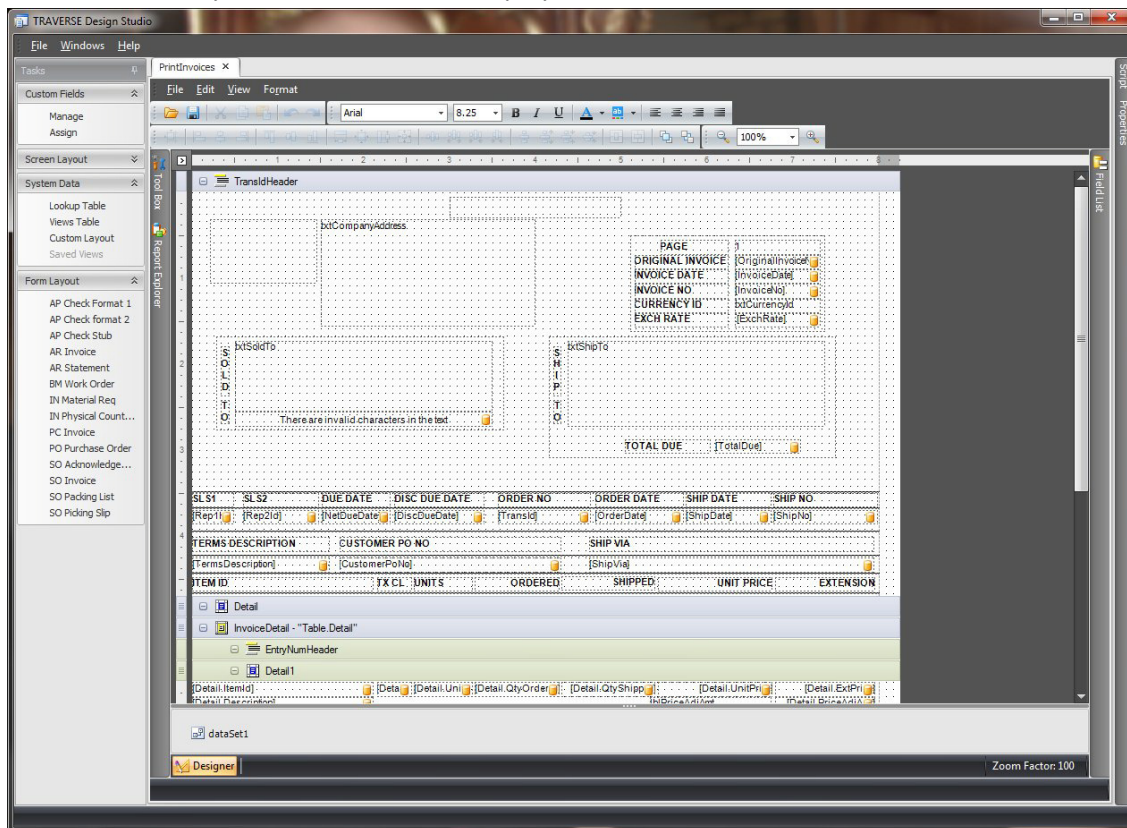
Form Layout

Use the Form Layout tool of TRAVERSE Design Studio to edit the appearance of TRAVERSE forms such as invoices, statements, checks, picking slips, and more.

Form Layout Menu

The Form Layout menu in the Tasks window of TRAVERSE Design Studio lists all of the available functions for which you can adjust the field layout.

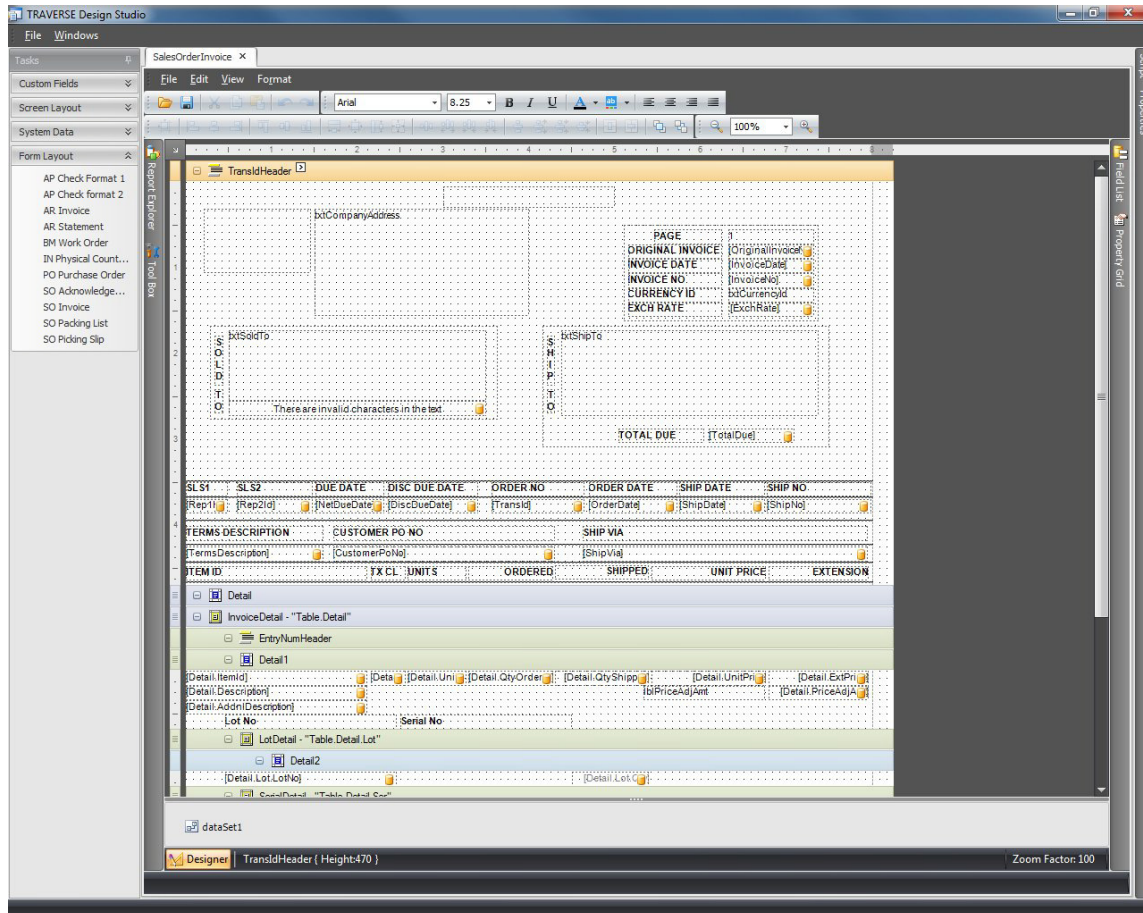
To edit the layout of a form, choose the form from the list. The Form Layout window opens with the form you've chosen to edit displayed in the window.



You can change the layout of the forms using the sections of the Form Layout window. You can drag, resize, or dock any section in or outside the main form at your discretion.

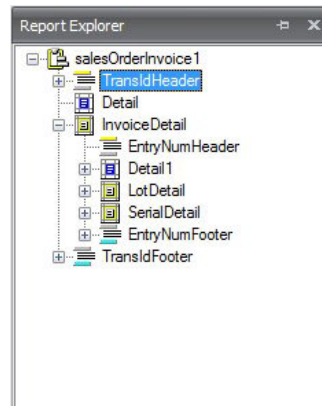
Designer

The designer section of the Form Layout function displays the elements of the form and their location within the form.



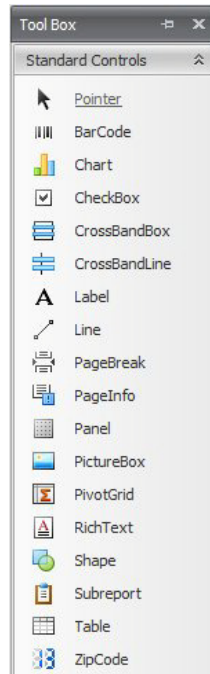
Report Explorer

The Report Explorer provides easy navigation through report elements. You can use the explorer when you build a form to quickly access all the elements of a report and their properties, and to see the whole form's structure.



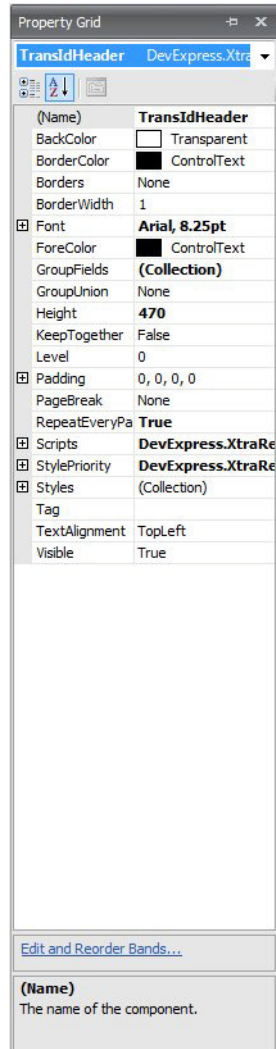
Tool Box

The Tool Box displays report elements that you may drop into the form. To do so, select a Tool Box item and drop it onto one of the report bands.



Property Grid

The Property Grid displays the property values of any element you highlight in the form.



Toolbar

Use the toolbar buttons to change the font properties of text elements within the form, or to zoom in or out on the form. You can also use any of a number of element alignment options when selecting one or more elements in the Designer section.



To save your form changes, click Save in the Form Layout toolbar.

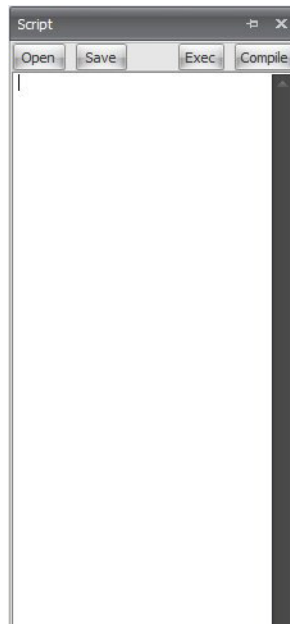
Scripting

The Script Window

Scripts are a specific type of programming that can be used to add functionality to the software without making changes to the source code of the program. Scripts can be used to add validations, restrictions, calculations and otherwise modify the behavior of the software.

In TRAVERSE, scripts are stored in files outside of the core software. This allows you to receive updates and enhancements to your software while generally leaving your customizations intact.

Click **Script** on the right side of the screen to open the Script window.



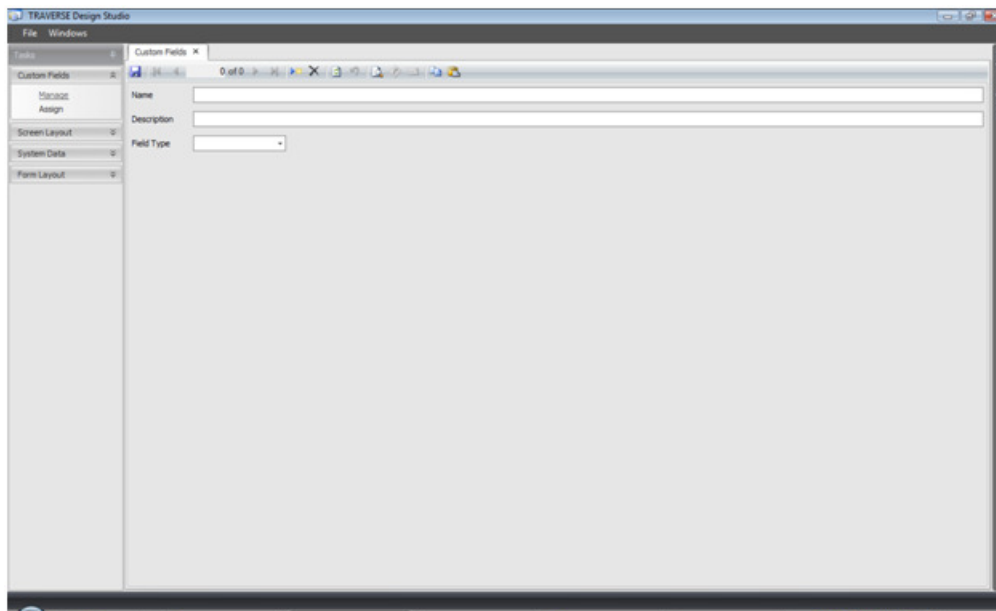
DESIGN STUDIO EXAMPLES

Custom Fields

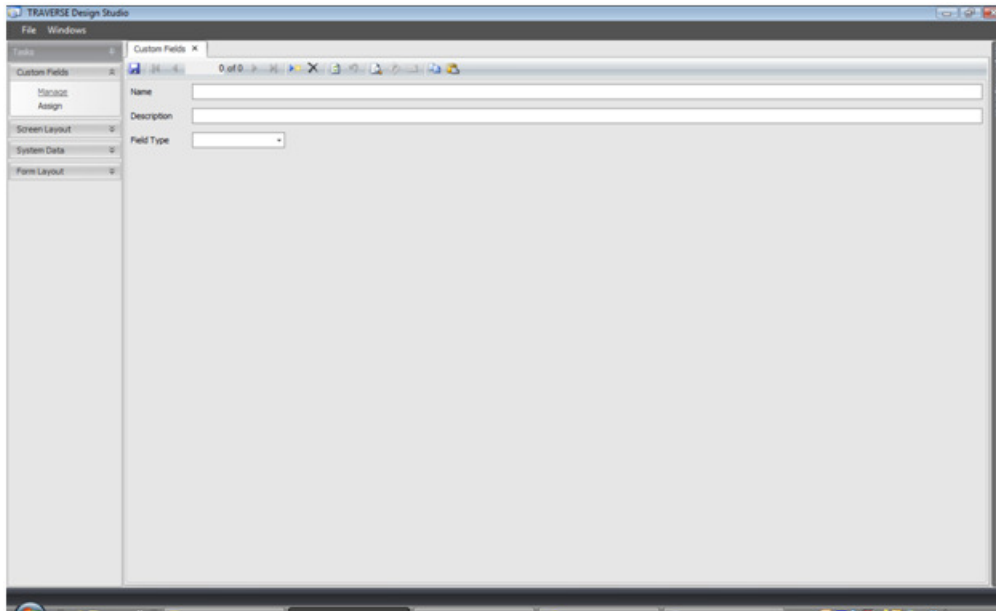
Manage - used to create/modify custom fields to be used within TRAVERSE.

*** Data is stored in tblSmCustomFields in the company database.

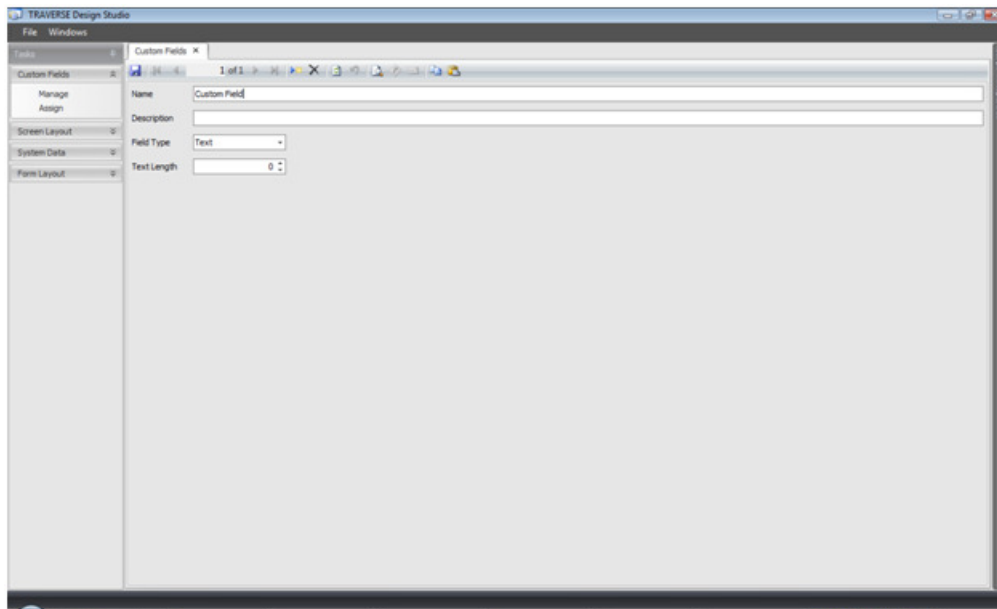
1. Click **Manage**.



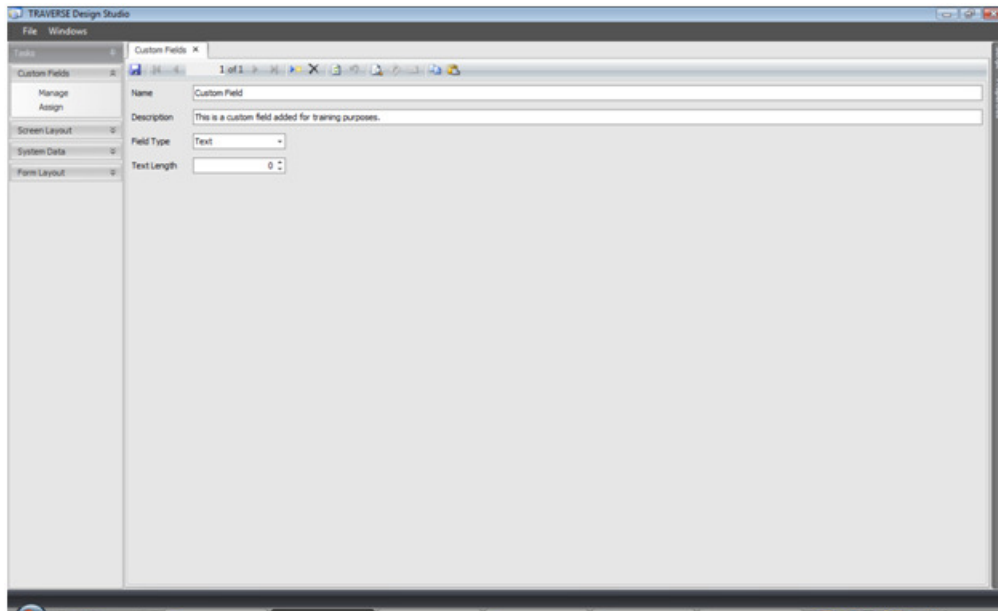
2. Click **Add New Record**.



3. Enter the **Name** - Name of the field as it appears on the TRAVERSE screen. We will call it **Custom Field**.



4. Enter a **Description** - This is text that will appear in TRAVERSE when the field is hovered over.



5. Select a **Field Type**

- **Text** - Field is alpha-numeric.
Assign a Text Length.
- **Number** - Field is numeric only.
Assign Maximum and Minimum Values.
- **Date** - Field is a date.
- **YesNo** - Field shows in TRAVERSE as checkbox for True/False or Yes/No.
- **List** - Field is a drop down containing a list of values.

Select a Lookup Id or enter Display Values.

If using Lookup Id, select a TRAVERSE Search Id to populate the drop down in TRAVERSE.

If using Display Values, enter values that will populate the drop down in TRAVERSE; you can also limit the values a user can select by checking the Limit to List checkbox.

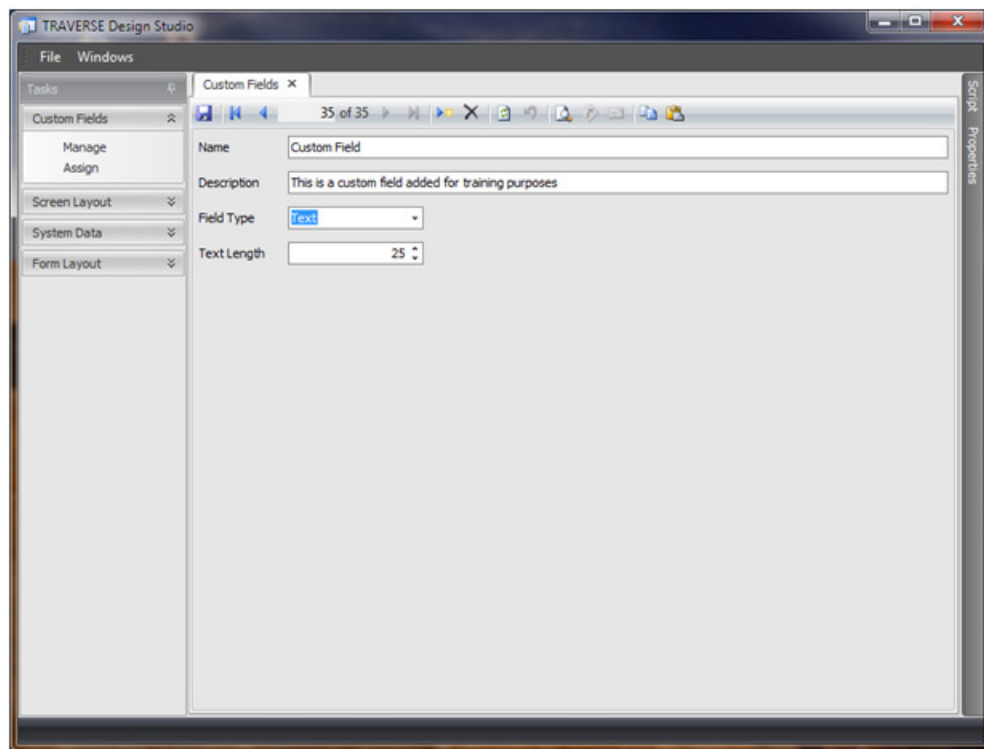
- **Color** - Field is a color (will show color pallet in TRAVERSE).
- **Entity List** - Field is a drop down containing a list of values.

Select a Lookup Id or enter Display Values.

If using Lookup Id, select a TRAVERSE Search Id to populate the drop down in TRAVERSE.

If using Display Values, enter values that will populate the drop down in TRAVERSE; you can also limit the values a user can select by checking the Limit to List checkbox.

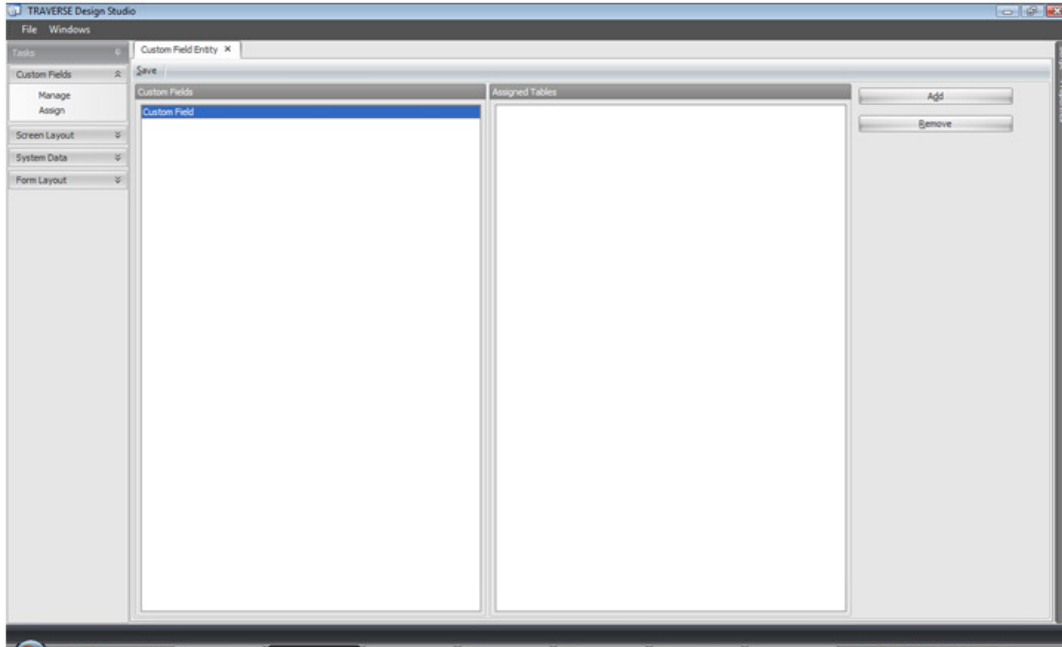
6. Click **save** to save the Custom Field.



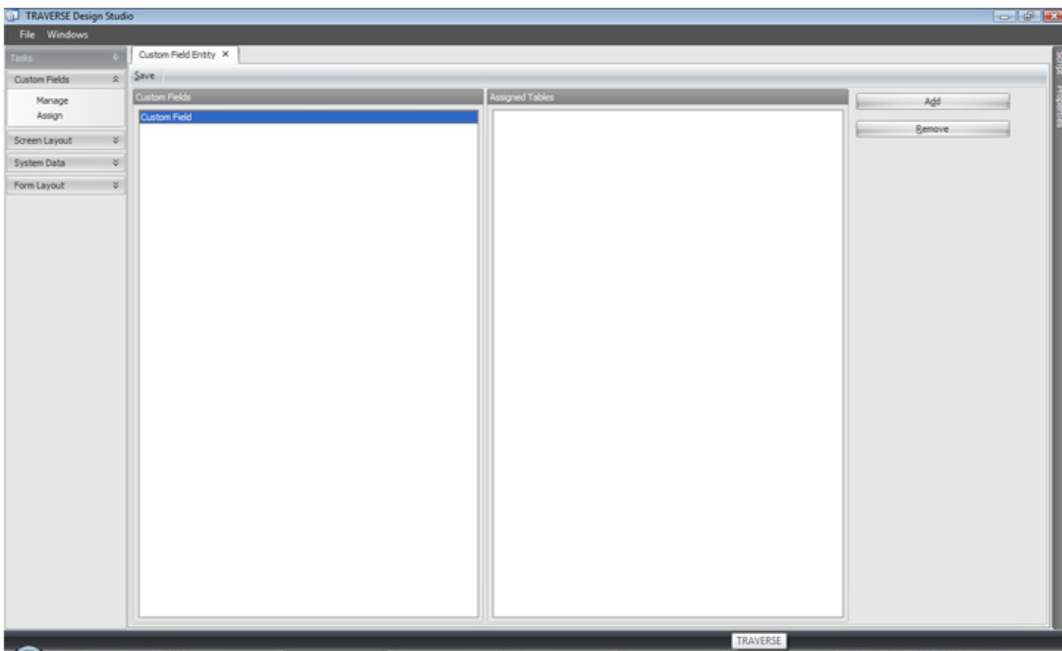
Assign - used to assign custom fields to TRAVERSE tables / screens.

*** Data is stored in tblSmCustomFieldsEntity in the company database.

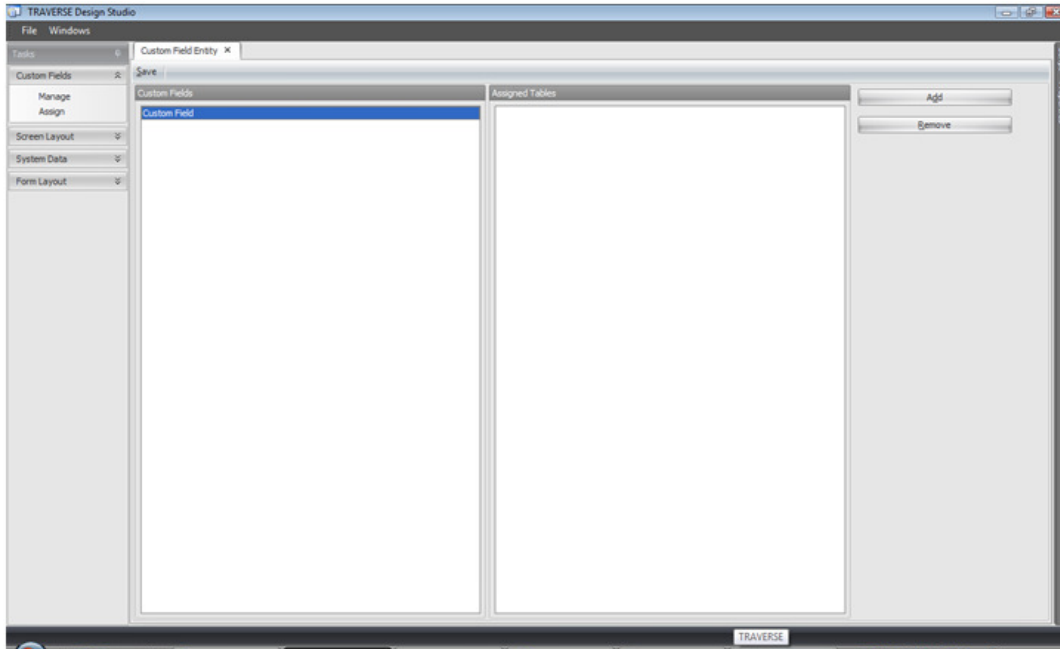
1. Click **Assign** to open the Custom Fields Entity screen.



2. Click on the field you wish to assign to a table. We should see **Custom Field**.

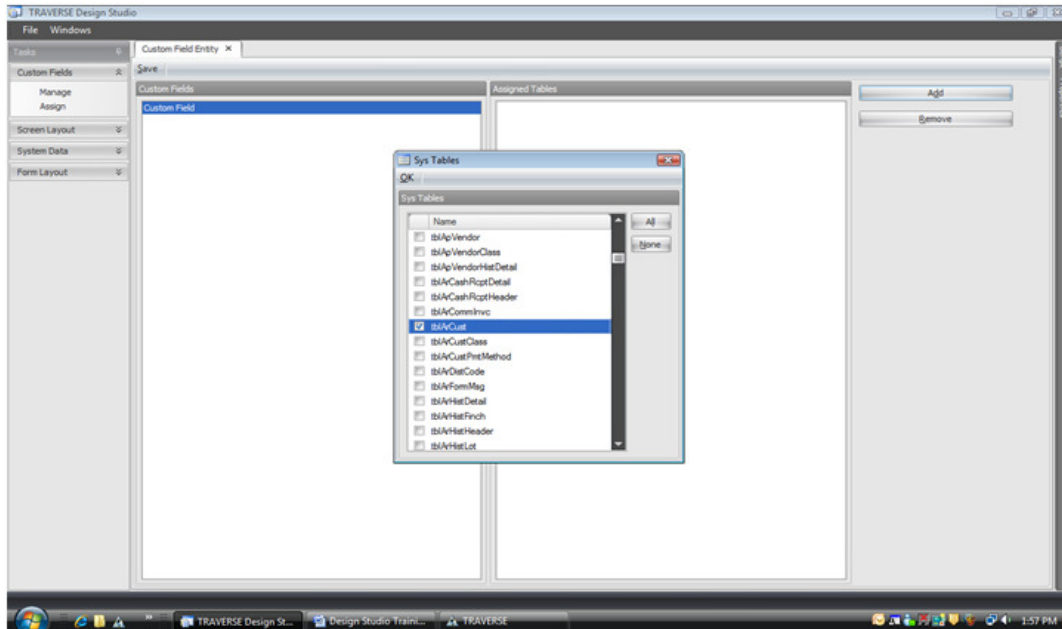


3. Click the **Add** button to select which tables to add the field to.

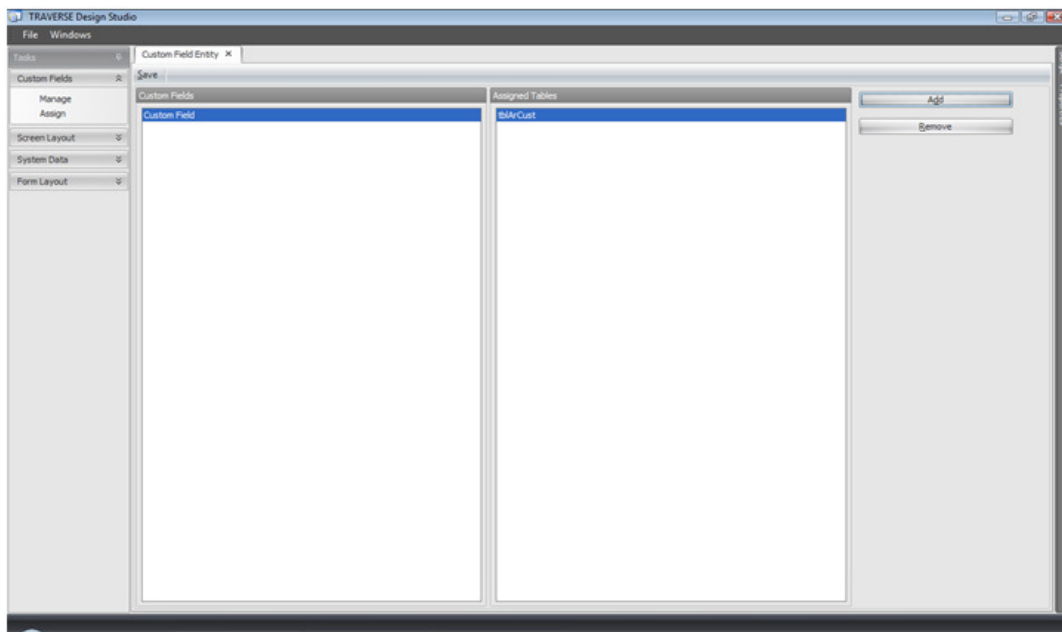


4. Select the table(s) to add field to "tblArCust" (selecting **All** will check all tables, **None** will uncheck all tables).

5. Click **OK** to close the table selection screen.



6. Click **Save** to assign the field to the selected table(s).



After the custom field has been created and assigned, you can open TRAVERSE, Accounts Receivable, Setup and Maintenance, Customers and the field should appear on the screen associated with the table you assigned the field to.

The screenshot displays the TRAVERSE Accounts Receivable Setup and Maintenance screen for Customers. The form is for Customer ID 1. The fields are organized as follows:

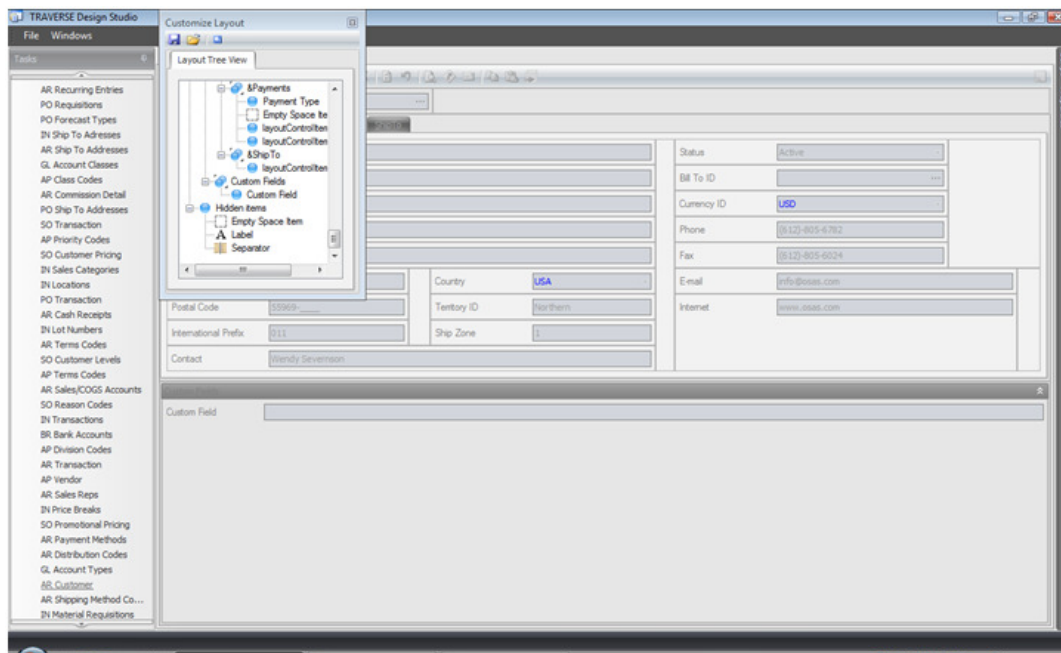
General		Defaults	Balance	Payments	Ship To
Name	customer 1	Status	Active		
Attention	Jon Dalmark	Bill To ID			
Address 1	945 Tuscon Drive	Currency ID	USD		
Address 2	No. 3	Phone	(612)-805-6782		
City	Rollingstone	Fax	(612)-805-6024		
Region	MN	Country	USA		
Postal Code	55969-_____	Territory ID	Northern		
International Prefix	011	Ship Zone	1		
Contact	Wendy Severson				

Below the main form, there is a section for Custom Fields. A custom field is listed with the text: "This is a custom field added for training purposes."

Screen Layout - used to modify TRAVERSE screens (TRAVERSE menus).

*** Contains a list of TRAVERSE screens available to modify via Design Studio.

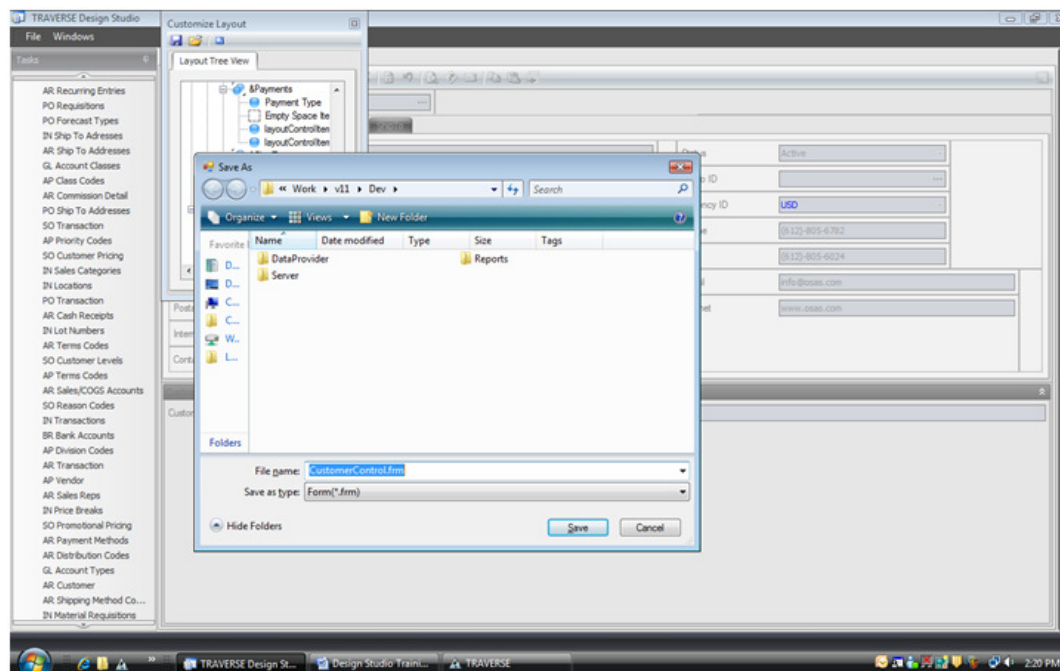
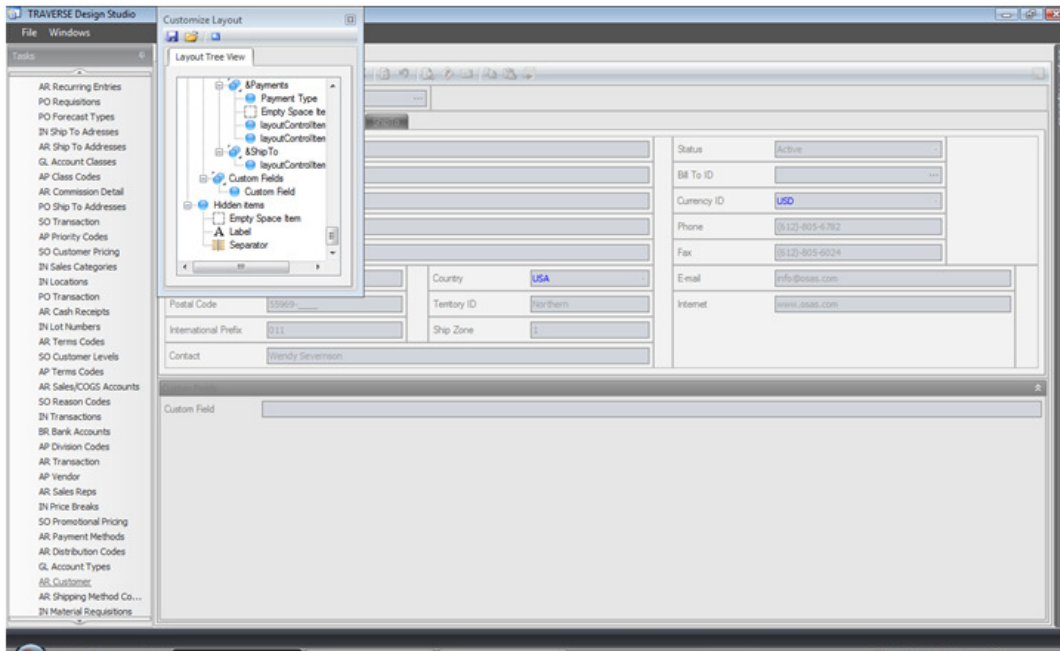
1. Click on a specific screen to modify it, AR Customer. The custom field will appear on screens that use the table(s) the custom field was assigned to.



2. Modify the screen as desired. Drag and drop the Custom Field under the Internet field.

NOTE: Custom fields must be created and assigned to get them to appear as fields available for the modified screen.

3. Click **Save** using the Use File, to save the modified screen to a build folder (you must accept the default naming convention).



After you have moved the field where you would like it and saved it to your build folder, copy it to your TRAVERSE client install folder. You can open TRAVERSE and you should see your modified screen.

The screenshot displays the TRAVERSE software interface. The main window is titled 'AR Customers' and shows a list of 71 records, with the first record selected. The record details are as follows:

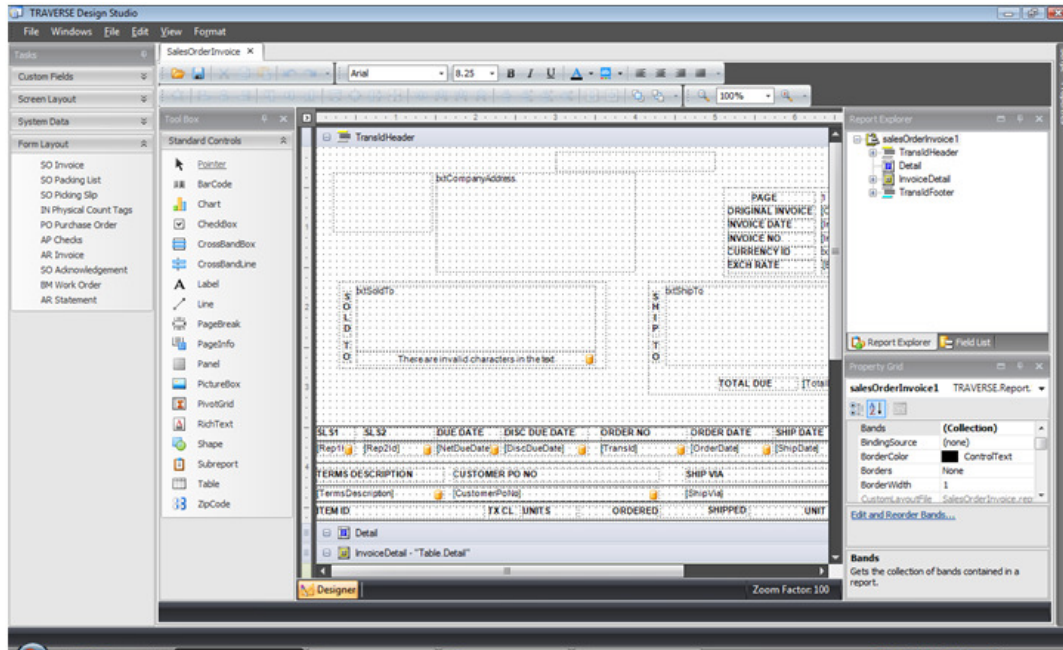
General		Defaults	Balance	Payments	Ship To
Name	customer 1				
Attention	Jon Dalmark				
Address 1	945 Tuscon Drive				
Address 2	No. 3				
City	Rollingstone				
Region	MN	Country	USA		
Postal Code	55969	Territory ID	Northern		
International Prefix	011	Ship Zone	1		
Contact	Wendy Severson				
Status	Active				
Bill To ID					
Currency ID	USD				
Phone	(612)-805-6782				
Fax	(612)-805-6024				
Email	info@posas.com				
Internet	www.posas.com				
Custom Field					

The left sidebar contains a tree view of the Accounts Receivable module, with 'Customers' selected. The bottom status bar shows 'test | X06'.

Form Layout - used to modify TRAVERSE forms (invoices, statements, etc).

*** Contains a list of TRAVERSE forms available to modify via Design Studio.

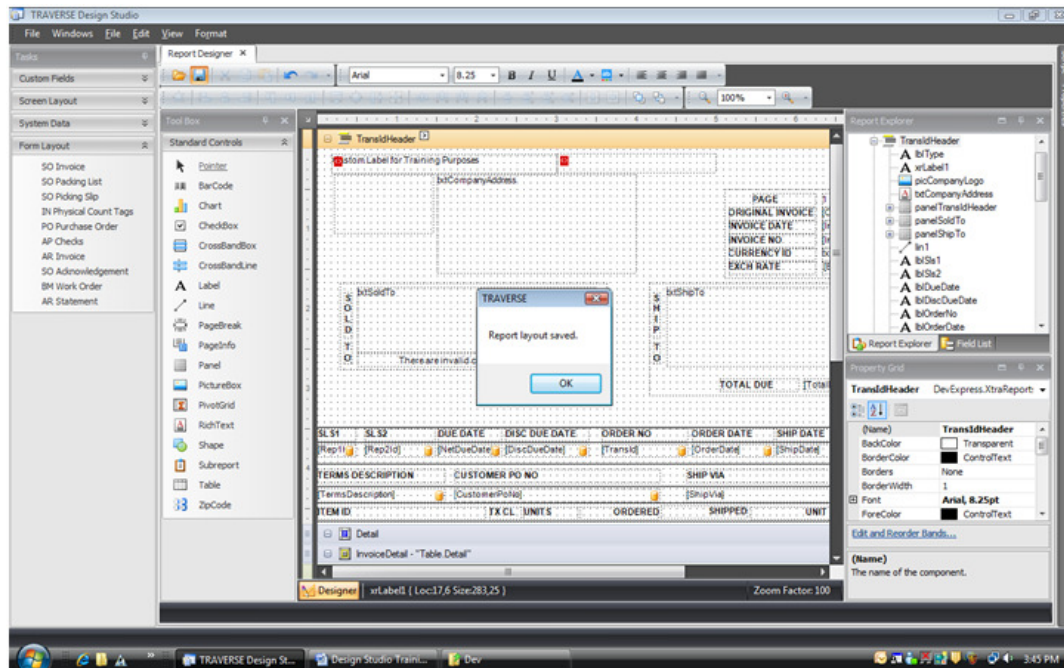
1. Click on a specific form to modify.



2. Modify the form as desired.

NOTE: At this time, custom fields are not available for forms.

3. Click **Save** to save the customized form.



4. After you have modified and saved your form, copy it to your TRAVERSE client install folder. You can open TRAVERSE and you should see your modified form when you print the form you modified using Design Studio.

Custom Label for Training Purposes

New Company
7525 Golden Triangle Drive
Eden Prairie, MN 55244-1203
UNITED STATES
(612)-629-0011

PAGE 1
INVOICE DATE 07/07/2009
INVOICE NO

S A0008
O Atlas Servers Company
L Jon Dahms
D 948 Tucson Drive
T No 3
O Rollingstone, MN 55969

S A0008
H Atlas Servers Company
I Jon Dahms
P 948 Tucson Drive
T No 3
O Rollingstone, MN 55969

TOTAL DUE 628.54

SL1	SL2	DUE DATE	DISC DUE DATE	ORDER NO	ORDER DATE	SHIP DATE	SHIP NO
GJL		06/07/2009	06/07/2009	00000002	07/07/2009	07/07/2009	

TERMS DESCRIPTION	CUSTOMER PO NO	SHIP VIA
0% Disc, Net 30		

ITEM ID	TX CL	UNITS	ORDERED	SHIPPED	UNIT PRICE	EXTENSION
100		3	PKG	1.00	1.00	628.54

Electrical Package
Includes Electrical Outlets and

Current Page: 1 Total Pages: 1 Zoom Factor: 100%

Add an image (logo) to a form in Design Studio

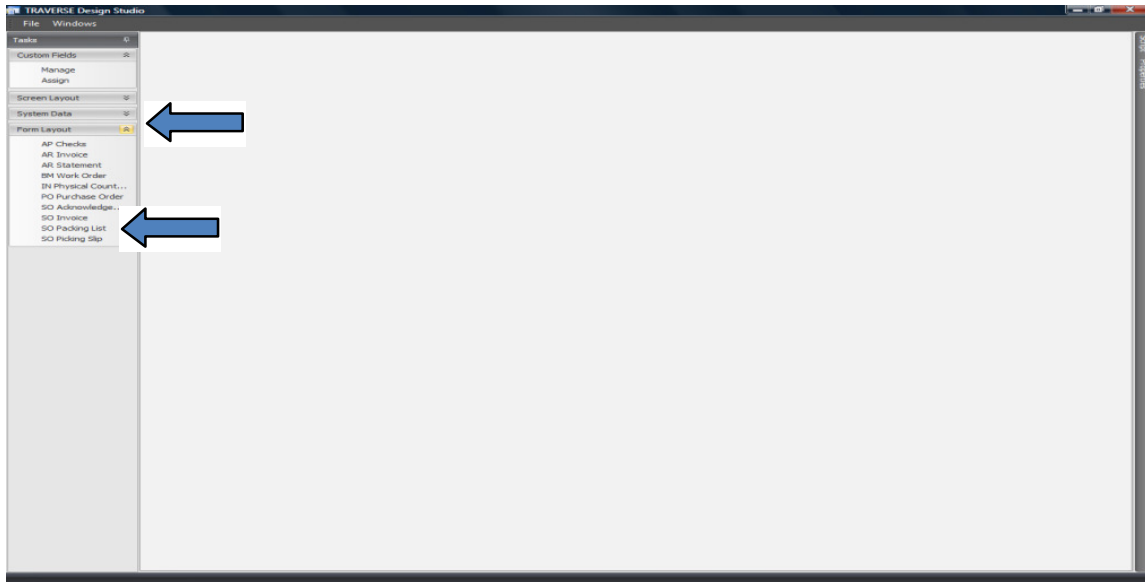
To add a logo to one of the printed forms using design studio follow these steps:

1. Created logo and saved as a jpg. Logo size 100x100 pixels.

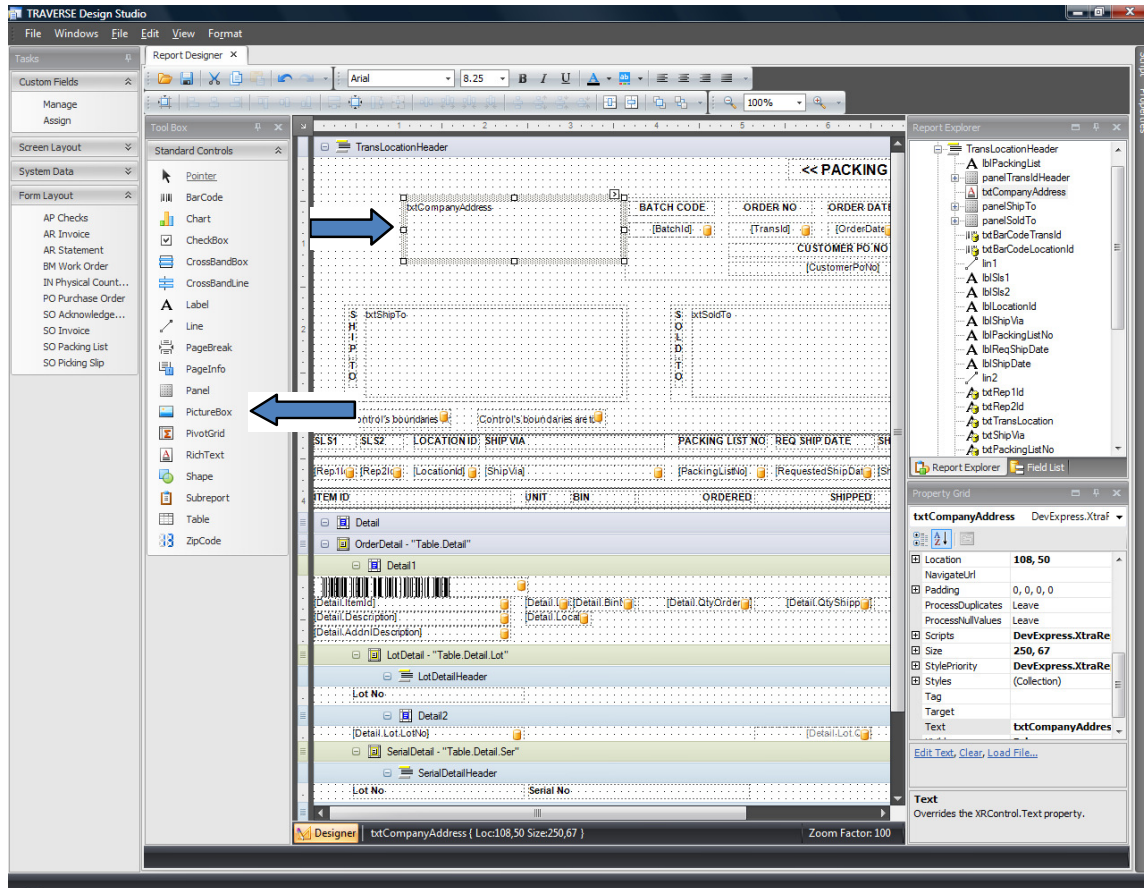


2. Open Design Studio.

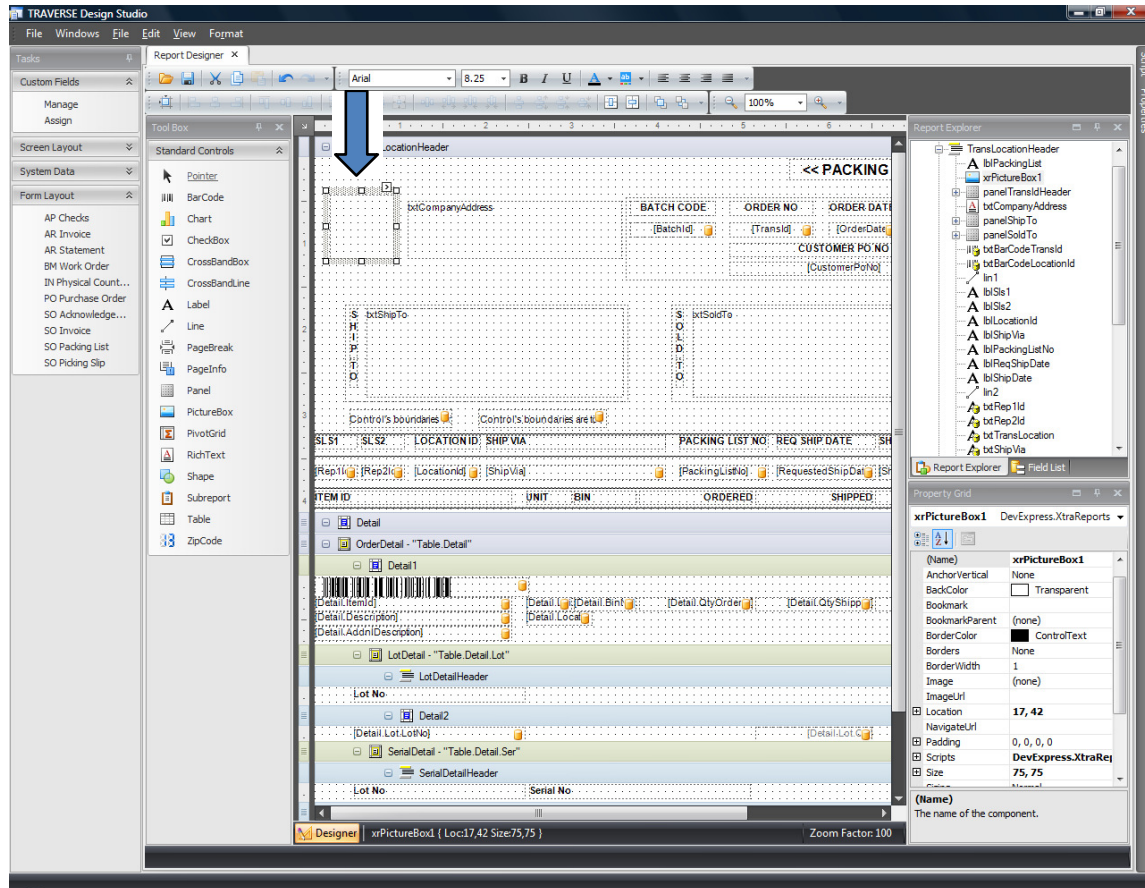
3. Select **Form Layout** drop down on the left and then select the form you would like to have logo added to, example **SO Packing List**.



4. Reposition the **txtCompanyAddress** box to the right to create space for the logo.



5. Select **PictureBox** under **Standard Controls** and position it where you would like logo to appear on the form.



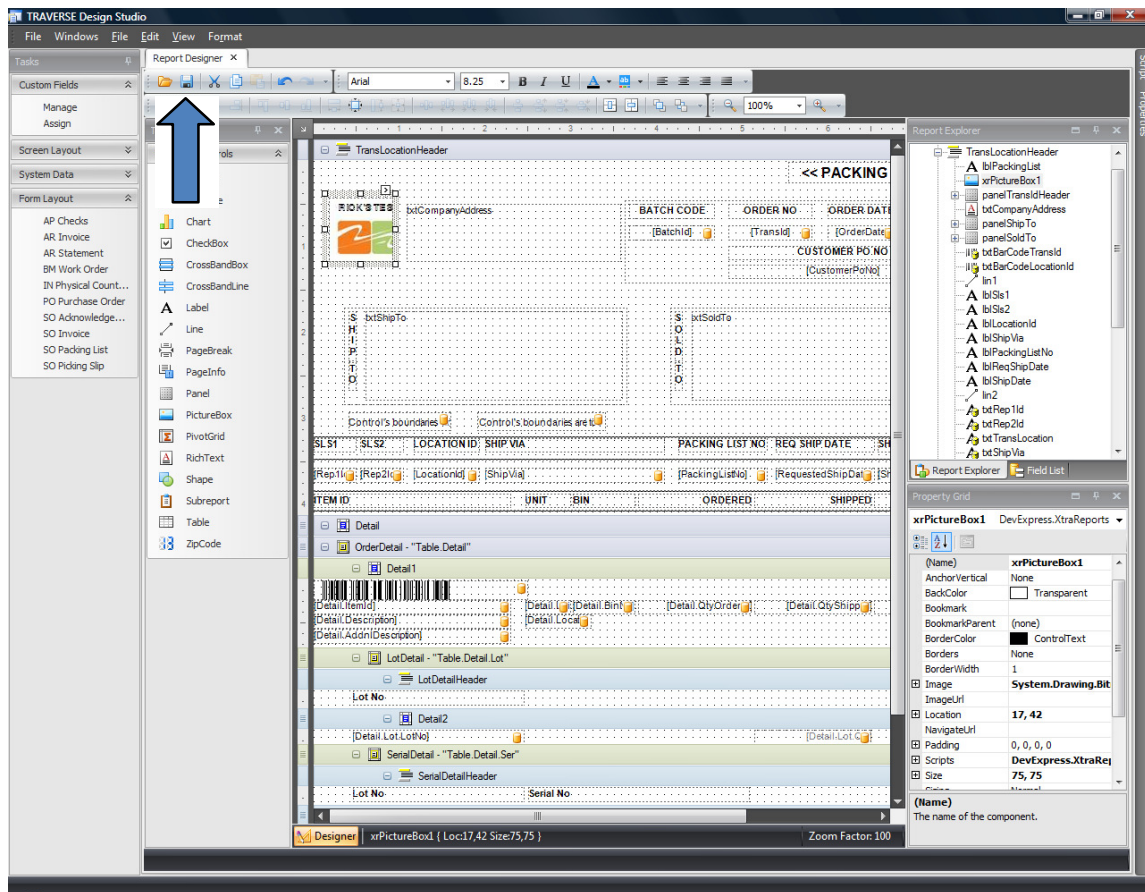
DESIGN STUDIO

Design Studio Examples

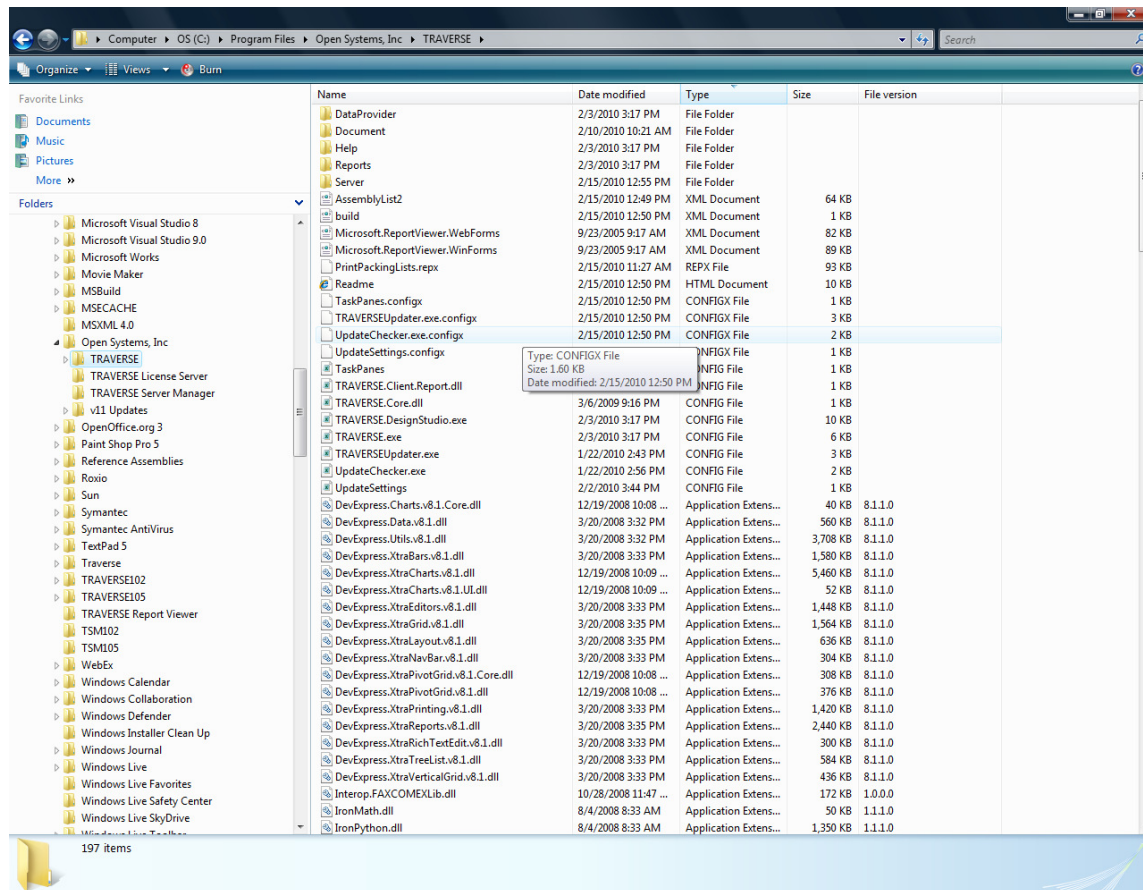
6. Click arrow at top of box to open the **XRPictureBox Tasks** and in the image field browse to the jpg that was saved in step 1.

XPictureBox Tasks

Image	System.Drawing.Bitmap ...
Data Binding	(None) ▾
Image Url	...
Data Binding	(None) ▾
Image Sizing	Normal ▾
Anchor Vertical	None ▾



7. Click on the **Save** button (blue floppy) to save the new form in the **Traverse Client** directory.



8. The next time you launch Traverse and print a **Packing List** the logo will appear on the form.

<< PACKING LIST >>

RIK'S TEST COMPANY
Rick's Test Company
9933 East Washington Ave
Suite 1500
Minneapolis, MN 55664-3333
UNITED STATES

BATCH CODE **ORDER NO** **ORDER DATE** **PAGE**
10000108 02/12/2010 1

CUSTOMER PO NO

S Alt008
H Altos Servers Company
I Jon Dalmark
P 945 Tuscon Drive
T No. 3
O Rollingstone, MN 55969

S Alt008
O Altos Servers Company
L Jon Dalmark
D 945 Tuscon Drive
T No. 3
O Rollingstone, MN 55969

SLS1	SLS2	LOCATION ID	SHIP VIA	PACKING LIST NO	REQ SHIP DATE	SHIP DATE
GJL	JSK	MN0001		10	02/12/2010	02/12/2010

ITEM ID	UNIT	BIN	ORDERED	SHIPPED	BACKORDERED
200100 Furnace This is a test of the additional description section printing. I pressed return so this should be on the next line.	EA		1.0000	1.0000	0.0000

Design Studio - System Data

NOTE: When editing the Lookup Table and Views Table you should not edit the existing records. Make a new record and copy from the existing. Then edit the new record and refer to the original using the Replace ID field.

Using Design Studio to format the information displayed using the Lookup Command.

There are many fields in Traverse that use the Lookup command, they can be distinguished by the use of the browse button  in the field. Some examples of these would be:

Customer ID Alt008 

***AR Customer**

Customer ID	Name	Contact	Region	Postal Code	Status
Alt008	Altos Servers Company	Wendy Severnson	MN	55969	Active
Arg026	ARG Systems Inc.	Sue Rogers		1600023	Active
Atm047	Asynchronous Networking Tech.		MN	56444	Active
Atm053	ATM Switches Inc.		SD	57442	Active
Axi040	Axis Electronics	David Johanson	ND	58079	Active
Bet023	Beta Dynamics Inc.		WI	53500	Active
Bit020	Bitstream Technology		IA	50300	Inactive

Transaction No

***SO Transaction No**

Trans ID	Batch Code	Invoice No.	PO No.	Customer ID	Sales Rep 1	Sales Rep 2	Ship To ID	Trans Type
00000018	*****	55		Pol014	GJL	MSL		RMA
00000021	Quotes			Mou027	GJL			Price Quote
00000022	Verify			Atm053	JSK			Picked
00000027	*****			Alt008	GJL	JSK		Verified
00000028	*****			Alt008	GJL	JSK		Invoice
00000029	*****		5455	Pol014	GJL	MSL		New

Design Studio allows you to change the information displayed when using these functions. There are several points you need to know before starting this process.

1. Do not modify these fields unless you have knowledge of database design and administration.
2. Make a **backup of your SYS database** before attempting any changes. When you save changes, made in this section of Design Studio, it will change your SYS database, and there is no UNDO button to back out these changes.

Information for these lookups are stored in the SYS database, TblSYSLookup, changes made in Design Studio, update the table when you use the SAVE button. Changes will affect all Users, all Companies.

NOTE: Changes are not recommended to any columns except the “DATA SOURCE” and “Default Columns”. Changes made to other columns may cause the lookup to not function properly.

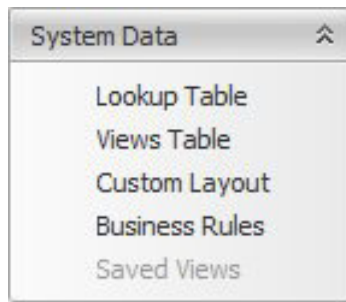
Example of a Lookup Table modification:

A user would like to see a field for “Terms Code” when looking up SO Transactions Numbers. The generic lookup for SO Transactions has these fields:

SO Transactions Number Lookup fields - generic

Trans ID	Batch Code	Invoice No.	PO No.	Customer ID	Sales Rep 1	Sales Rep 2	Ship To ID	Trans Type

1. Open Design Studio, and under **Tasks** - Select **System Data**
2. Select - **System data - Lookup Table**



3. Find the Lookup Table that you want to edit. In our example we need **SOTrans**.

SO Trans Lookup

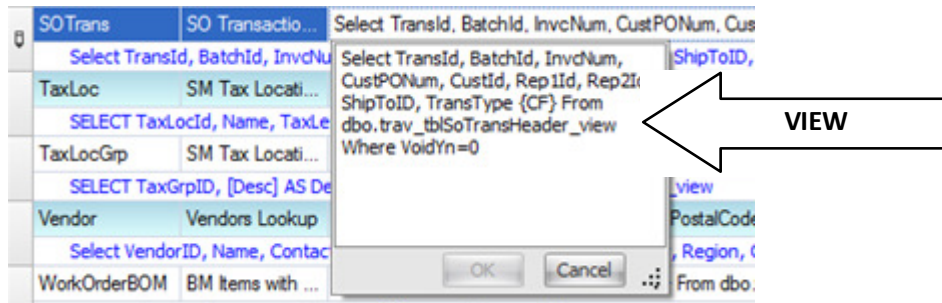
SOTrans	SO Transactions...	Select TransId, BatchId, InvcNum, CustPONum, ...
		Select TransId, BatchId, InvcNum, CustPONum, CustId, Rep1Id, Rep2Id, ShipToID, Tra

4. As noted above, we are only going to edit the “Data Source” and “Default Columns” fields. The first column to edit is the “Data Source”. The default information in that column is”

SO Trans Lookup - Data Source column - Select TransId, BatchId, InvcNum, CustPONum, CustId, Rep1Id, Rep2Id, ShipToID, TransType {CF} From dbo.trav_tblSoTransHeader_view Where VoidYn=0

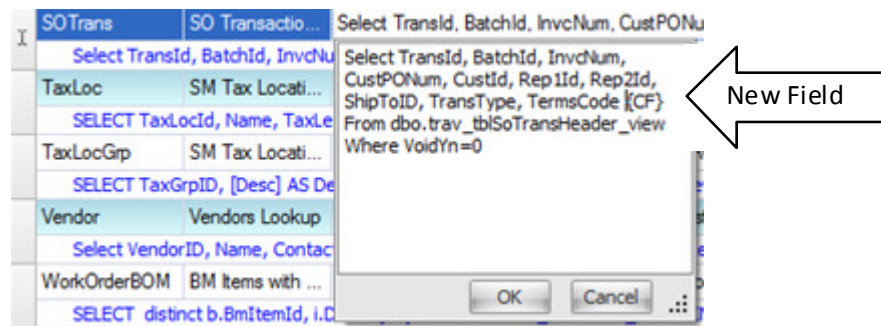
5. If you click in the field you can see this display:

SO Trans Lookup - Data Source column - dropdown



6. For our example, the first thing we need to do is verify that the field "Terms Code" is available in the VIEW used in this data source column. That VIEW is `dbo.trav_tblSoTransHeader_view`. Using SQL Management Studio, we find the View. We look under our v11 SQL instance - Databases - CPU (Company ID) - Views - `dbo.trav_tblSoTransHeader_view`. Expand the View and then expand the Columns - this will display a list of the columns available for our lookup. We will find a "TermCode" on the list, so we can add it to the lookup.

To add "Terms Code" to the lookup we will add to the "Data Source Column" just after the "TransType" field. Click in the "Data Source Column" (as displayed above) and put the cursor in the field just after TransType and insert our new field, just as it appears in the View - "TermsCode". The field names need to be separated by a comma. The column should now look like this:



(Select TransId, BatchId, InvNum, CustPONum, CustId, Rep1Id, Rep2Id, ShipToId, TransType, TermsCode {CF} From `dbo.trav_tblSoTransHeader_view` Where VoidYn=0)

7. Next we want to update the "Default Column", the generic column should be:

SO Trans Lookup - "Default Column" - generic

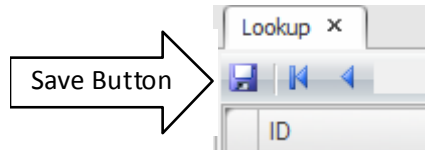
TransId, BatchId, InvNum, CustPONum, CustId, Rep1Id, Rep2Id, ShipToID, TransType

8. To add "**Terms Code**" to the lookup we will add it to the "Default Column" just after the "TransType" field. Click in the "Default Column" (as displayed above) and put the cursor in the field just after TransType and insert our new field, just as it appears in the View - "TermsCode". The field names need to be separated by a comma. The column should now look like this:

SO Trans Lookup - Default Column" column - After Edit

TransId, BatchId, InvNum, CustPONum, CustId, Rep1Id, Rep2Id, ShipToID, TransType, TermsCode

9. Save your changes using the **Save** button.



10. Now we want to open Traverse to see if our changes worked out correctly. Open SO - Transactions - and select the browse button for a Transaction number. The screen should display our changes.

SO Transactions Number Lookup fields - after changes

Trans ID	Batch Code	Invoice No.	PO No.	Customer ID	Sales Rep 1	Sales Rep 2	Ship To ID	Trans Type	TermsCode
00000018	#####	55		Pol014	GJL	MSL		RMA	1%Disc
00000021	Quotes			Mou027	GJL			Price Quote	Net30
00000022	Verify			Atm053	JSK			Picked	Net30
00000027	#####			Alt008	GJL	JSK		Verified	Net30

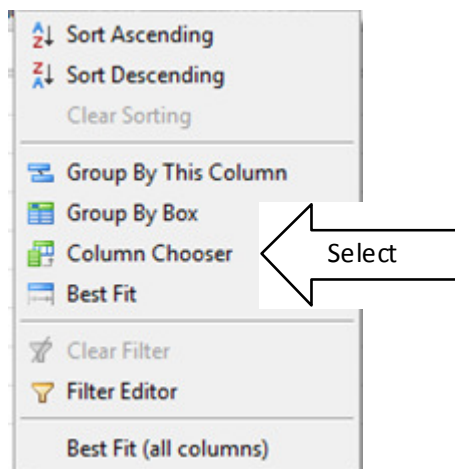
NOTE: If you do not see your changes when you open the dropdown, use the "Column Chooser" to see if they are available to add to the display.

Right click in a column header field

Trans ID	Batch Code	Invoice No.	PO No.	Customer ID	S
> 00000018	#####			Pol014	G

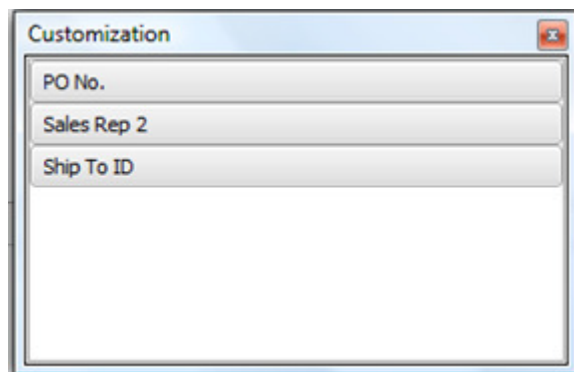
Right Click in Header

You should now see these options,



11. Select the “**Column Chooser**” to see the columns available to add to the display.

Available column fields display

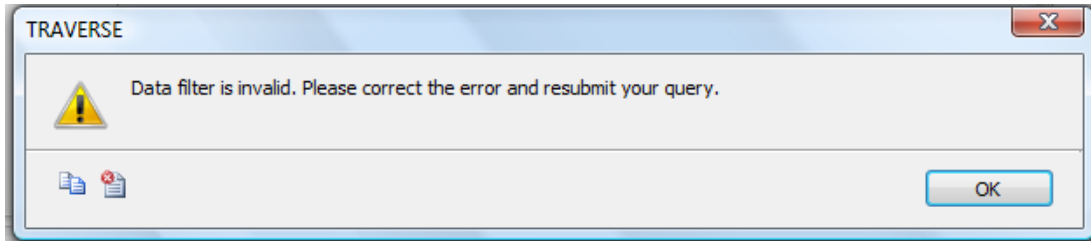


12. Move the fields To-From the column Chooser, by dragging them with your mouse.

Possible Errors:

If we had made a mistake with our modification, we could see a message like this opening the SO Transaction screen:

SO Transaction screen - Data Filter error



---I deliberately entered too many commas in the “Data Source” column to get this error

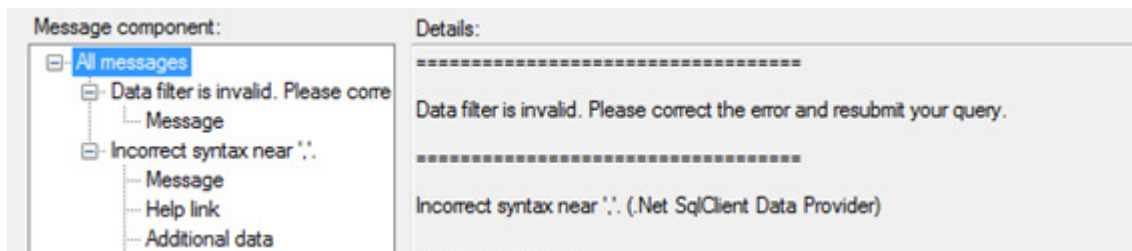
--- The incorrect text from Design Studio is:

Error in “Data Source” Column

`. TransType, . TermsCode {CF} From dbo.trav_tblSoTransHea...`

If I select the “Error Flag” above ...  ..., you can see the error detail.

Message displayed in Traverse



To fix this go back to Design Studio and take out the extra comma and save the lookup.

The next time you go into SO transactions the error should not appear.

Editing the System Views Table

Use the System Data, Views Table function to configure the information shown in the interactive views within TRAVERSE without making changes directly to SQL Server tables.

Use the Views Table function to define columns available for the view, whether you want a field to appear as filter criteria, any joins and grouping that need to be made when pulling data from multiple tables, the default fields to display when opening the view and resetting the view once it has been changed within TRAVERSE, the order in which fields appear in the results area, any enumerated values that need to be translated, and so on.

NOTE: Do not modify these fields unless you have knowledge of database design and administration.

1. To define what appears and is available when interactive views are opened, select Views Table from the System Data menu in TRAVERSE Design Studio. The Views Table screen appears.

ID	Description	Data Source	Available	Default Columns	Enumerated Columns	Formats	Replace ID
CustomARHistory	Custom AR Detail History	SELECT * FROM (SELECT p	Hist Type, Counter, Trans T...	Hist Type, Trans Type, Tran...	HistType:1;Payment:2;Fina...		ARHistory
Customers	Customer	SELECT * FROM (SELEC	CustId, CustName, Attn, C...	CustId, CustName, Contact...	Status:0;Active:1;Inactive...		
DComponentPegging	DR Component Pegging Vi...	dbo trav_DComponentPe...	MRPDate, SeqNum, DocC...	MRPDate, Source, DocID...	Source:1;Purchase Orders...		
DRDailyAvailability	Daily Availability View	dbo trav_DRDailyAvailab...	Itemid, Locid, Date, OnHand...	Date, Demand, Supply, NetA...			
DRItemActivityView	Item Activity View	dbo trav_DRItemActivityVe...	Source, Date, Locid, OrderN...	Source, Date, Locid, OrderN...	Source:1;Purchase Orders...		
Employee	Employee	SELECT * FROM (Select	Employeeid, Status, LastNam...	Employeeid, LastName, Fir...	EmployeeStatus:0;Full-Tim...		
FALeaseVsBuyAnalysis	FA Lease Vs Buy Analysis I...	LeaseVsBuyAnalysisData	RecType, Year, Type, Col...	RecType, Year, Type, Col...	RecType:0;Lease Analysis...		
GIJournal	Transactions	SELECT * FROM (SELEC	EntryNum, CompId, EntryD...	AcctId, EntryDate, SourceC...	BalType:0;Memo:1;Debit...		
GIJournalHistory	Transactions	SELECT * FROM (SELEC	EntryNum, CompId, EntryD...	AcctId, EntryDate, SourceC...	BalType:0;Memo:1;Debit...		
InDetailHistory	Detail History	SELECT * FROM (SELEC	HistSeqNum, Itemid, Locid...	TransDate, Itemid, Locid...	Source:10;PO Receipt:11...		
InItem	Item	SELECT * FROM (SELEC	Itemid, Locid, Descr, Add...	Itemid, Locid, ProductLine...	ItemStatus:1;Active:2;Dis...		
InItemAvailability	Item Availability	select * from trav_InQtyDet	Source, Qty, TransID, Entry...	Source, Qty, TransID, Entry...	TransType:0;Committed:1...		
InLot	Lot Number	SELECT * FROM (SELEC	Itemid, ItemLocStatus, Uo...	Itemid, Descr, Locid, LotN...	ItemType:1;Nonserialized:2...		
InLotHistory	Lot History	SELECT * FROM (SELEC	HistSeqNum, Itemid, Locid...	TransDate, Itemid, Locid...	Source:10;PO Receipt:11...		
InSerial	Serial Number	SELECT * FROM (SELEC	Itemid, Descr, LottedYN, It...	Itemid, Descr, SerNum, Lo...	SerNumStatus:1;Available...		
InSerializedHistory	Serialized History	SELECT * FROM (SELEC	Itemid, Locid, LotNum, Ap...	TransDate, Itemid, Locid...	Source:10;PO Receipt:11...		
InSummaryHistory	Summary History	SELECT * FROM (SELEC	Itemid, Uom, Locid, LotNu...	TransType, SumYear, GLP...	TransType:1;Purchase:2;P...		
MBMediaGroup	Media Group	SELECT * FROM (SELEC	MGID, Descr, PrimaryLink	MGID, Descr, PrimaryLink			
MPDispatchedProduction	MP Dispatched production	dbo trav_MpDispatch_Proc	SeqNo, Priority, OrderNo, Re...	SeqNo, Priority, OrderNo, Re...			
MPPinishedGoodsLotNum...	Finished Goods Lot Number	SELECT * FROM (SELEC	FgLotNum, AssemblyId, Ord...	FgLotNum, AssemblyId, Ord...			
MPPinishedGoodsSerialNu...	Finished Goods Serial Num...	SELECT * FROM (SELEC	FgLotNum, FgSerNum, Asse...	FgSerNum, OrderNo, Relea...			
MpGIWpTrans	MP GL WIP Transactions	SELECT (all) *	EntryNum, EntryDate, Orde...	EntryDate, OrderNo, Relea...	TransType:1;Adjustment:2...		
MpLotNumberWhereUsed	Lot Number Where Used	SELECT * FROM (SELEC	FgLotNum, AssemblyId, Ord...	RmLotNum, ComponentId...			
MPOrderActivityView	MP Order Activity View	Select * from (SELECT r O	OrderNo, ReleaseNo, CustI...	OrderNo, ReleaseNo, CustI...	Status:0;New:1;Planned:2...		
MPOrderStatusView	Order Status View	SELECT * FROM (Select o	OrderNo, ReleaseNo, CustI...	OrderNo, ReleaseNo, CustI...	Status:0;New:1;Planned:2...		
MPPrioritizeOrder	Prioritize Order	SELECT * FROM (SELEC	Priority, OrderNo, ReleaseN...	Priority, OrderNo, ReleaseN...			
MpProductionHistoryView	MP Production History View	Select * from (SELECT r	OrderNo, ReleaseNo, CustI...	OrderNo, ReleaseNo, CustI...	OrderSource:0;Manual:1...		
MPProductionSchedule	Production Schedule	SELECT * FROM (SELEC	OrderNo, ReleaseNo, Asse...	Priority, OrderNo, ReleaseN...	Status:0;New:1;Planned:2...		
MPSalesOrderAnalysis	MP Sales Order Analysis	dbo trav_MpSalesOrderAn...	OrderDate, ProdDate, SOTr...	AssemblyId, Locid, Descripti...			

2. Select the new record button to add a new record to the list of view tables.
3. We are going to edit the ARHistory view data.
4. Enter CustomARHistory into the ID field, Custom AR Detail History into the Description field and ARHistory into the ReplaceID field.

5. Browse up to the ARHistory view record and click in the Data Source field. The selection script will appear. You can pull the lower right corner of the box to extend the field to view more of the script.

```
SELECT * FROM (SELECT p.Counter, 1 AS [HistType], 2 AS [TransType], p.[PostRun], h.[CustPONum], p.[TransId],
p.[CustId], p.[CustId] AS [SoldToId], p.[CheckNum] AS [DocumentId], p.[PmtDate] AS [TransDate],
p.[PmtDate] AS [DocumentDate], p.[FiscalYear], p.[GLPeriod] AS [FiscalPeriod],
NULL AS [Description], -p.[PmtAmt] - p.[CalcGainLoss] AS [Amount], -p.[PmtAmtFgn] AS [AmountFgn],
0 AS [Cost], 0 AS [CostFgn], p.[CF], h.[BatchId], h.[ShipToId], h.[ShipToName], h.[ShipVia],
p.[InvNum], h.[CredMemNum], h.[OrderDate], p.[CurrencyID], ISNULL(p.[VoidYn], 0) [VoidYn],
p.[Counter] [EntryNum], p.[RepId], p.[RepZId], h.[ShipNum], h.[ShipDate], h.[InvDate], h.[ExchRate],
h.[Source], h.[Notes], NULL as [ItemJob], NULL as [WhseId], NULL as [PartId], NULL as [JobId], NULL as [PhaseId],
NULL as [PartType], NULL as [AddnDesc], NULL as [LottedYn], NULL as [ReqShipDate], NULL as [ActShipDate],
0 as [Qty], NULL as [UnitsSel], 0 as [UnitPriceSel], 0 as [UnitPriceSelfgn], 0 as [UnitCostSel],
0 as [UnitCostSelfgn], 0 as [QtyOrdSel], 0 as [QtyShipSel], 0 as [ExtFinalInc], 0 as [ExtOrigInc],
0 as [ExtFinalIncFgn], NULL as [TaskId], NULL as [PhaseName], NULL as [ProjName], NULL as [TaskName],
0 as [PriceAdjType], 0 as [PriceAdjAmt], 0 as [UnitPriceSelfgn], 0 as [UnitPriceSelfgnFgn],
0 as [PriceAdjAmt], 0 as [PriceAdjAmtFgn], 0 as [LineSeq], 0 as [Status], NULL as [ResCode],
s.[CustName], s.[ClassId], s.[GroupCode], s.[AcctType], s.[PriceCode], s.[CreditLimit], s.[TerrId],
s.[CustLevel], s.[Status] CustomerStatus, NULL [CatId], h.GLAcctReceivables, h.GLAcctSalesTax,
h.GLAcctFreight, h.GLAcctMisc, h.GLAcctGainLoss, NULL AS GLAcctSales, NULL AS GLAcctCOGS, NULL AS GLAcctInv (CF)
FROM [[COMP]].dbo.tblARHistmt p
LEFT JOIN [[COMP]].dbo.tblARHistHeader h ON p.[PostRun] = h.[PostRun] AND p.[TransId] = h.[TransId]
left join [[COMP]].dbo.tblARCust_View s on p.CustId = s.CustId WHERE p.[VoidYn] = 0
UNION ALL
SELECT 1 as Counter, 2 AS [HistType], 1 AS [TransType], f.[PostRun], NULL AS CustPONum, NULL AS [TransId],
f.[CustId] AS [SoldToId], NULL AS [DocumentId], f.[FinchDate] AS [TransDate], f.[FinchDate] AS [DocumentDate],
f.[FiscalYear], f.[GLPeriod] AS [FiscalPeriod], NULL AS [Description], f.[FinchAmt] AS [Amount],
f.[FinchAmtFgn] AS [AmountFgn], 0 AS [Cost], 0 AS [CostFgn], f.[CF], NULL as [BatchId], NULL as [ShipToId],
NULL as [ShipToName], NULL as [ShipVia], NULL as [InvNum], NULL as [CredMemNum], NULL as [OrderDate],
f.[CurrencyID], 0 AS [VoidYn], NULL [EntryNum], NULL AS [RepId], NULL AS [RepZId], NULL as [ShipNum],
NULL as [ShipDate], NULL as [InvDate], 0 as [ExchRate], NULL as [Source], NULL as [Notes],
NULL as [ItemJob], NULL as [WhseId], NULL as [PartId], NULL as [JobId], NULL as [PhaseId],
NULL as [PartType], NULL as [AddnDesc], NULL as [LottedYn], NULL as [ReqShipDate],
NULL as [ActShipDate], 0 as [Qty], NULL as [UnitsSel], 0 as [UnitPriceSel], 0 as [UnitPriceSelfgn],
0 as [UnitCostSel], 0 as [UnitCostSelfgn], 0 as [QtyOrdSel], 0 as [QtyShipSel], 0 as [ExtFinalInc],
0 as [ExtOrigInc], 0 as [ExtFinalIncFgn], NULL as [TaskId], NULL as [PhaseName], NULL as [ProjName],
NULL as [TaskName], 0 as [PriceAdjType], 0 as [PriceAdjAmt], 0 as [UnitPriceSelfgn],
0 as [UnitPriceSelfgnFgn], 0 as [PriceAdjAmt], 0 as [PriceAdjAmtFgn], 0 as [LineSeq],
0 as [Status], NULL as [ResCode], s.[CustName], s.[ClassId], s.[GroupCode], s.[AcctType], s.[PriceCode],
s.[CreditLimit], s.[TerrId], s.[CustLevel], s.[Status] CustomerStatus, NULL [CatId],
GLAcctReceivables, f.GLAcctFinch AS GLAcctSalesTax, NULL GLAcctFreight, NULL GLAcctMisc, NULL GLAcctGainLoss, NULL AS GLAcctSales, NULL AS GLAcctCOGS, NULL AS GLAcctInv (CF)
FROM [[COMP]].dbo.tblARHistFinch f
left join [[COMP]].dbo.tblARCust_View s on f.CustId = s.CustId
UNION ALL
SELECT 1 as Counter, CASE d.[EntryNum]
WHEN -1 THEN 4 WHEN -2 THEN 5 WHEN -3 THEN 6 WHEN -4 THEN 7 ELSE 3 END AS [HistType], h.[TransType],
h.[PostRun], h.[CustPONum], h.[TransId], h.[CustId], h.[SoldToId],
CASE WHEN h.[TransType] = 1 THEN h.[InvNum] ELSE h.[CredMemNum] END AS [DocumentId],
h.[OrderDate] AS [TransDate], h.[InvDate] AS [DocumentDate], h.[FiscalYear], h.[GLPeriod] AS [FiscalPeriod],
d.[Desc] AS [Description], SIGN(h.[TransType]) * d.[PriceExt] AS [Amount],
SIGN(h.[TransType]) * d.[PriceExtFgn] AS [AmountFgn], SIGN(h.[TransType]) * d.[CostExt] AS [Cost],
SIGN(h.[TransType]) * d.[CostExtFgn] AS [CostFgn], h.[CF], h.[BatchId], h.[ShipToId], h.[ShipToName],
h.[ShipVia], h.[InvNum], h.[CredMemNum], h.[OrderDate], h.[CurrencyID], h.[VoidYn], d.[EntryNum],
d.[RepId], d.[RepZId], h.[ShipNum], h.[ShipDate], h.[InvDate], h.[ExchRate], h.[Source], h.[Notes],
d.[ItemJob], d.[WhseId], d.[PartId], d.[JobId], d.[PhaseId], d.[PartType], d.[AddnDesc], d.[LottedYn],
d.[ReqShipDate], d.[ActShipDate], d.[Qty], d.[UnitsSel], SIGN(h.[TransType]) * d.[UnitPriceSel] UnitPriceSel,
SIGN(h.[TransType]) * d.[UnitPriceSelfgn] UnitPriceSelfgn, SIGN(h.[TransType]) * d.[UnitCostSel] UnitCostSel,
SIGN(h.[TransType]) * d.[UnitCostSelfgn] UnitCostSelfgn, d.[QtyOrdSel], d.[QtyShipSel], d.[ExtFinalInc],
d.[ExtOrigInc], d.[ExtFinalIncFgn], d.[TaskId], d.[PhaseName], d.[ProjName], d.[PriceAdjType],
d.[PriceAdjAmt], d.[PriceAdjAmtFgn], SIGN(h.[TransType]) * d.[UnitPriceSelfgnFgn] UnitPriceSelfgnFgn,
d.[PriceAdjAmt], d.[PriceAdjAmtFgn], d.[LineSeq], d.[Status], d.[ResCode], s.[CustName], s.[ClassId],
s.[GroupCode], s.[AcctType], s.[PriceCode], s.[CreditLimit], s.[TerrId], s.[CustLevel],
s.[Status] CustomerStatus, d.[CatId], h.GLAcctReceivables, h.GLAcctSalesTax,
h.GLAcctFreight, h.GLAcctMisc, h.GLAcctGainLoss, d.GLAcctSales, d.GLAcctCOGS, d.GLAcctInv (CF)
FROM [[COMP]].dbo.tblARHistHeader h
LEFT JOIN [[COMP]].dbo.tblARHistDetail d ON h.[PostRun] = d.[PostRun] AND h.[TransId] = d.[TransId]
left join [[COMP]].dbo.tblARCust_View s on h.CustId = s.CustId WHERE d.[OrpId] IS NULL and h.VoidYn = 0
) ds
```

6. Highlight the whole script in the box and press Ctrl C to copy the script.
7. Find your CustomARHistory record and paste the script into the Data Source field.
8. Do the same copy and paste for the Available, Default Columns and Enumerated Columns fields.

We want to add the custom fields assigned to the Customer maintenance table as available column chooser fields.

The custom fields can be added using a couple of different methods.

One method is this:

You can use the {CF} after the last field listed in the select statement. Don't put any commas before or after the {CF}.

Change the table the {CF} is coming from to the view (dbo.trav_tblxxxxxx_view).

The fields will be automatically brought into the view as available fields.

If you use the {CF} you can only have one view in the select statement. It will bring in the {CF} fields only from the first view it finds.

If you have multiple select statements with UNION ALL in between them you must put the same thing at the end of the select statement before the FROM.

Another method is this:

For more than one table to bring in the custom fields use s.[cf_custom field 1 name], s.[cf_custom field 2 name] as the custom fields at the end of the select statement.

NOTE: The s. refers to the tblArCust table in this view script. Find the letter indicating the table you want the custom field to be brought in from within the script.

Then in the available field you would add [cf_custom field 1 name], [cf_custom field 2 name] at the end of the list.

Or you can alias them like: s.[cf_custom field 1 name] AS Custom Field 1, s.[cf_custom field 2 name] AS Custom Field 2

Then in the available field you would add Custom Field 1, Custom Field 2 with the alias names as the field 1 and field 2.

9. We will use the first method described on the previous page to add the custom fields.

10. In the select script for our custom view table find the first occurrence of FROM and go up one line and add {CF} to the end of that line.

, h.GIAcctGainLoss, NULL AS GLAcctSales, NULL AS GLAcctCOGS, NULL AS GLAcctInv {CF}

FROM [{COMP}].dbo.tblArHistPmt p

11. Find the next occurrence of FROM in the next section of the select script and do the same thing with the {CF} and so on until you have the {CF} in each of the select sections.

12. In the sections starting with the FROM statement, find the listing for tblArCust and change it to dbo.trav_tblArCust_View.

```
FROM [{COMP}].dbo.tblArHistPmt p

LEFT JOIN [{COMP}].dbo.tblArHistHeader h ON p.[PostRun] = h.[PostRun] AND
p.[TransId] = h.[TransId]

left Join [{COMP}].dbo. trav_tblArCust_view s on p.CustId = s.CustId WHERE
p.[VoidYn] = 0

UNION ALL
```

13. Edit each of the From and Join sections to rename the table to a view as above.
14. You should now have the custom fields available as column chooser fields in the Accounts Receivable, Interactive Views, Detail History View within the TRAVERSE 11 client software.

Custom Scripting

The Script Window

Scripts are a specific type of programming that can be used to add functionality to the software without making changes to the source code of the program. Scripts can be used to add validations, restrictions, calculations and otherwise modify the behavior of the software.

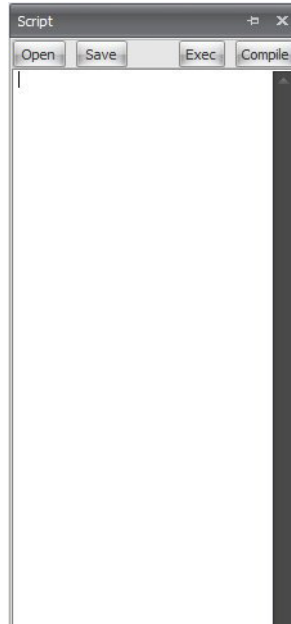
In TRAVERSE, scripts are stored in files outside of the core software. This allows you to receive updates and enhancements to your software while generally leaving your customizations intact.

NOTE: Most scripting languages will work with TRAVERSE Design Studio. Open Systems recommends using Iron Python scripting to enter your custom scripts.

Scripting can be used in the Screen Layout function.

The scripting tab is displayed in the upper right corner of these screens.

1. Click **Script** on the right side of the screen to open the Script window.



2. Once the script window is open you can either type your script directly into the window, or click the Open button and browse to a file that contains the script you want to use.
3. Once you have finished your script you will need to click on the Compile button to compile the script and check for any syntax errors.
4. You can execute the script to see that it is performing the functionality you want it to perform.
5. Once you have compiled and executed the script you will need to Save it.
6. When the script is saved it will have a specific name associating the script with the screen layout you are working with. You must use the naming convention of the default saved file for TRAVERSE to recognize the script to be executed for the screen layout you are using it for.

An example of the naming convention for a script is:

TRAVERSE.Client.Contacts.CustomerControl.py this example will be associated with the customer maintenance screen.

This saved file must be copied to each workstation that will have the script active. The file must be placed in the TRAVERSE folder where the client programs have been installed.

Below are some examples of scripts that have been used in programmer training classes.

- Changing the background color for the Customer ID, depending on the status of the customer:

```
def RecordChanged(o, e):
    # Check the customer status and then change the background color
    if main.CurrentEntity.Status == 0:
        main.lkpCustId.BackColor = Color.LightGreen
    else:
        main.lkpCustId.BackColor = Color.Blue
```

```
main.BindingSource.PositionChanged += RecordChanged
```

For the color to change using this script you must select a different customer, or save the record before the color will change.

- Changing the background color for the Customer ID, depending on the status of the customer as soon as the customer id is selected:

```
def RecordChanged(o, e):
    ChangeStatusBackColor()

def CustomerPropertyChanged(o, e):
    if e.PropertyName == "Status":
        ChangeStatusBackColor()

def CustomerCurrentChanged(o, e):
    if main.CurrentEntity != None:
        main.CurrentEntity.PropertyChanged -= CustomerPropertyChanged
        main.CurrentEntity.PropertyChanged += CustomerPropertyChanged

def ChangeStatusBackColor():
    if main.CurrentEntity.Status == 0:
        main.GetControls()["lkpCustId"].BackColor = Color.LightGreen
    else:
        main.GetControls()["lkpCustId"].BackColor = Color.Blue

main.BindingSource.PositionChanged += RecordChanged
main.BindingSource.CurrentChanged += CustomerCurrentChanged
```

With this script the color of the Customer ID field will change as soon as the customer id is selected.

- Displaying a message that a field is required.
- In this example we have added a custom field called Customer Type and we want it to be a required field. We added this script to pop up a message if the Customer Type field has been left blank:

```
def Req (o, e):
    if main.CurrentEntity.CustomValueCollection.Contains("Customer Type"):
        v=main.CurrentEntity.CustomValueCollection["Customer Type"].Value
        if (v == ""):
            e.Cancel=True
    else:
        MessageBox.Show("Value required in Customer Type field.")
        e.Cancel=True
```

```
main.SavingRow+=Req
```

- Change the color of the font for the Customer ID field when the Customer Lookup box has been expanded, if the customer is over their credit limit:

```
def HeaderPropertyChanged(o, e):
    if (e.PropertyName == "SoldToId"):
        SoldToColorChange()
```

```
def SoldToColorChange():
    header = main.ViewComponents.HeaderControl.CurrentEntity
    headerCtl = main.ViewComponents.HeaderControl
```

```
    if header.SoldToCustomer.CreditLimit != 0:
        if header.SoldToCustomer.TotalDue > header.SoldToCustomer.CreditLimit:
            headerCtl.GetControls()["lkpCustomerId"].ForeColor = Color.Red
        else:
            headerCtl.GetControls()["lkpCustomerId"].ForeColor = Color.Blue
```

```
def HeaderCurrentChanged(o, e):
    if main.ViewComponents.HeaderControl.CurrentEntity != None:
        main.ViewComponents.HeaderControl.CurrentEntity.PropertyChanged-
        =HeaderPropertyChanged
        main.ViewComponents.HeaderControl.CurrentEntity.PropertyChanged +=
        HeaderPropertyChanged
```

```
main.ViewComponents.HeaderControl.BindingSource.CurrentChanged+=HeaderCurrentChanged
```

- Default a fixed message to be put into the Notes field for either the SO transactions or PO transactions. In this example the words “IronPython Training” will be put into the notes field for the transaction.

```
def SetNotes (o, e):
```

```
    main.ViewComponents.HeaderControl.CurrentEntity.Notes = "IronPython Training"
```

```
main.ViewComponents.HeaderControl.SavingRow+=SetNotes
```

- Add an additional description for items in a specific product line. This script will add the additional description of “Freight Surcharge Applies” to transactions where the items selected have a product line of “A1” set up in Inventory.

```
def SetFreightDesc (o, e):
```

```
    header = main.ViewComponents.HeaderControl.CurrentEntity
```

```
    for linelItem in header.LinelItemList:
```

```
        if linelItem.CatId == "A1":
```

```
            linelItem.AddnldDescr = "Freight Surcharge Applies"
```

```
main.ViewComponents.HeaderControl.SavingRow+=SetFreightDesc
```

- Show the minimum price set up for line items. This script will display the minimum price set up for items in Inventory that have a product line of “A1”.

```
def ShowMinPrice (o, e):
```

```
    header = main.ViewComponents.HeaderControl.CurrentEntity
```

```
    for linelItem in header.LinelItemList:
```

```
        if linelItem.CatId == "A1":
```

```
            MessageBox.Show(linelItem.InItem.GetPriceInfo(linelItem.InItemSellingUnit).PriceMin.ToString())
```

```
main.ViewComponents.ItemSummaryControl.SavingRow+=ShowMinPrice
```

- Show a message if the price calculated or entered for an item in a Sales Order is less than the minimum price of the item in Inventory.

```
def ShowMinPrice (o, e):
```

```
    detail = main.ViewComponents.ItemSummaryControl.CurrentEntity
```

```
    minPrice = detail.InItem.GetPriceInfo(detail.InItemSellingUnit).PriceMin
```

```
    if detail.UnitPriceSellFgn < minPrice:
```

```
        MessageBox.Show("Price cannot be less than the minimum price.", "Minimum Price")
```

```
detail.UnitPriceSellFgn = detail.InItem.GetPriceInfo(detail.InItemSellingUnit).PriceBase
```

```
main.ViewComponents.ItemSummaryControl.SavingRow+=ShowMinPrice
main.ViewComponents.ItemDetailControl.SavingRow+=ShowMinPrice
```

- A script that requires a purchase order number for the sales order transaction if the PO Required custom field is set to true in Customer setup.
- First we would have to set up a custom field called PO Required and assign it to the tblArCust table.
- This script contains comments that describe what the different sections of the script are doing. The comments are preceeded by the # symbol.

```
# Add reference to enumerate Sales Order Transaction Header columns for use in Property
Changed Event
import clr
clr.AddReference("TRAVERSE.Business.SalesOrder")
from TRAVERSE.Business.SalesOrder import TransactionHeaderBase
from System import String
```

```
#Method to cancel the save if customer requires a purchase
#order number (PO Required custom field assigned to
#customer table
def HeaderSaved(o, e):
    if ValidatePO():
        e.Cancel = True
```

```
#If customer purchase order or Sold-To changes call the ValidatePO method
def HeaderPropertyChanged(o, e):
    if (e.PropertyName == TransactionHeaderBase.Columns.CustPONum.ToString()) or
    (e.PropertyName == TransactionHeaderBase.Columns.SoldToId.ToString()):
        ValidatePO()
```

```
#Check the PO Required custom field, if true, require the a purchase order number
def ValidatePO():
    # Set variable to Sales Order Transaction Header Control
    headerCtrl = main.ViewComponents.HeaderControl

    if
headerCtrl.CurrentTransactionHeader.SoldToCustomer.CustomValueCollection.Contains("PO
Required"):
    po = headerCtrl.CurrentTransactionHeader.SoldToCustomer.CustomValueCollection["PO
Required"].Value.ToString()
```

```

if String.IsNullOrEmpty(headerCtrl.CurrentTransactionHeader.CustPONum) == True:
    if po == "True":
        TravMessageBox.Show(None,"Purchase Order required.")
    return 1

```

```

#Add property changed event handler
def HeaderCurrentChanged(o, e):
    if main.ViewComponents.HeaderControl.CurrentEntity != None:
        main.ViewComponents.HeaderControl.CurrentEntity.PropertyChanged -=
        HeaderPropertyChanged
        main.ViewComponents.HeaderControl.CurrentEntity.PropertyChanged +=
        HeaderPropertyChanged

```

```

#Add event handlers for when the sales order transaction is saved and when the current
#transaction changes
main.ViewComponents.HeaderControl.SavingRow+=HeaderSaved
main.ViewComponents.HeaderControl.BindingSource.CurrentChanged+=HeaderCurrentChang
ed

```

- Adding a digital signature to a check when the amount is below a certain amount, or leave the digital signature off when the amount is above the limit.
- This will add the digital signature when the check amount is less than \$1,000 and print the blank signature line when the amount is above \$1,000.

1. Open the AP Check Format 1 Form Layout
2. Add a Picture Box, from the Tool Box, at the following location: 525,328. Set the name property to picSignature. Use the properties windows to enter the Location and Name.
3. Click the arrow at the top right corner of the picture box and click the Browse button next to the Image field. Browse to a signature file. Format the picture within the control added above to the .
4. Next add a line with location of 525, 267. The size should be 250,8.
5. Add the following code to the BeforePrint event of the form itself:

```

private void OnBeforePrint(object sender, System.Drawing.Printing.PrintEventArgs e)
{
    // Retrieve the TotalDue column from the data set, convert to double
    double dblCheckAmt = Convert.ToDouble(GetCurrentColumnValue("CheckAmt"));

    // Display signature if total due exceeds 1000

```

```
if (dblCheckAmt < 1000)
this.picSignature.Visible = true;
else
this.picSignature.Visible = false;
}
```

Scripting notes:

- The # symbol is the remark symbol to be able to enter comments into the formula and have them not be included in the calculation of the formula.
- When using the conditional if: else: make sure to put a colon after the if statement and after the else command. When defining the then, the line after the if, and when defining the else, the line after the else, be consistent with the number of spaces used to indent the conditional values.
- TRAVERSE uses Iron Python scripting as the scripting language to write the scripts. There are many books available as references to use when writing scripts.

- Recommended Reading:

IronPython In Action by Michael J Foord

Professional IronPython: Design and Develop IronPython Techniques by John Paul Mueller

- Web Resources:

www.ironpython.net

<http://ironpython.codeplex.com/>

- Tools

Object Browser in Visual Studio

RedGate Reflector Decompiler

