

---

Training Manual for

# **Advanced Report Writer**

**By Parish Data System**

## **Advanced Report Writer for PDS Church Office, Formation Office, School Office, Ledger, Ledger/Payroll, and Facility Scheduler**

Copyright © 2006 ACS Technologies Group, Inc.® All rights reserved.

Information in this document is subject to change without notice. Names and data used in examples herein are fictitious unless otherwise noted.

Parish Data System®, PDS®, and the PDS logo are registered trademarks of ACS Technologies Group, Inc. PDS Formation Office Management™, PDS Church Office Management™, and PDS School Office Management™ are trademarks of ACS Technologies Group, Inc.

Microsoft® Windows®, Windows NT®, Windows XP, Windows ME, Windows 2000, Windows 2003, and MS-DOS® are either registered trademarks or trademarks of Microsoft Corporation. Pentium® is a registered trademark of Intel® Corporation. All other brand and product names mentioned in this document may be trademarks or registered trademarks of their respective companies and are hereby acknowledged.

### **Parish Data System**

14425 North 19th Avenue

Phoenix, Arizona 85023-6702

Orders and Information: (800) 892-5202

Technical Support: (602) 789-0595

Fax: (602) 789-0597

E-mail: [solutions@parishdata.com](mailto:solutions@parishdata.com)

Web Site: [www.parishdata.com](http://www.parishdata.com)

---

# Contents

<b>1: Introduction .....</b>	<b>1-1</b>
<b>Overview.....</b>	<b>1-1</b>
Copy vs. Add.....	1-1
UDR files.....	1-2
INI files .....	1-4
<b>2: Report Basics.....</b>	<b>2-1</b>
<b>Overview.....</b>	<b>2-1</b>
<b>The Work Space.....</b>	<b>2-1</b>
<b>Toolbars.....</b>	<b>2-2</b>
Objects.....	2-2
Adding an Object.....	2-3
Labels and Memos.....	2-3
Images .....	2-4
System Variables .....	2-4
Variables.....	2-5
Shapes and Lines .....	2-5
Data Fields.....	2-6
Data Calculations .....	2-6
Data Images.....	2-7
Selecting Objects .....	2-7
Right Click on Objects .....	2-8
Moving Objects .....	2-9
<b>Report Tree and Data Tree.....</b>	<b>2-10</b>
Report Tree.....	2-10
Data Tree .....	2-11
<b>Templates .....</b>	<b>2-12</b>
PDS Easy List Template.....	2-12
PDS Header/ Footer Template.....	2-13
Fonts and Page Style .....	2-14
Preview.....	2-15
<b>Exercise.....</b>	<b>2-15</b>
<b>Notes.....</b>	<b>2-18</b>
<b>3: Sub-reports, Groups and Regions.....</b>	<b>3-1</b>
<b>Overview.....</b>	<b>3-1</b>
<b>Sub-reports.....</b>	<b>3-1</b>
Adding a Sub-report .....	3-1
Data Pipeline .....	3-2
Modifying a Sub-report .....	3-2
Exercise .....	3-3
Right Click on Sub-reports .....	3-4
<b>Groups .....</b>	<b>3-5</b>
Add a Group .....	3-5
Other Group Options .....	3-6
Group Header and Footer .....	3-6
<b>Regions.....</b>	<b>3-7</b>
Add a Region.....	3-7

Right Click on a Region .....	3-7
<b>Exercise.....</b>	<b>3-8</b>
<b>Project: Building the Family Directory .....</b>	<b>3-9</b>
Start the Report.....	3-9
The Header .....	3-9
The Detail .....	3-10
Add the Sub-Report.....	3-12
Setup the Sub-Report.....	3-12
Adding the Group.....	3-14
 <b>4: Embedded Code.....</b>	 <b>4-1</b>
<b>Overview.....</b>	<b>4-1</b>
Code Editor.....	4-1
<b>Data .....</b>	<b>4-3</b>
Data Fields.....	4-3
Objects.....	4-3
Variables.....	4-5
<b>Code .....</b>	<b>4-6</b>
BEGIN-END statements .....	4-6
Local Variable declarations.....	4-6
Assignment Statements .....	4-7
IF-THEN statements.....	4-7
CASE statements.....	4-9
WHILE loops .....	4-9
REPEAT-UNTIL loops .....	4-9
FOR loops .....	4-10
Built-in Procedure statements.....	4-10
Conditions .....	4-11
Calculations.....	4-11
Comments.....	4-12
Test the Code.....	4-12
View the Code.....	4-13
Print the Code.....	4-13
<b>Ongoing Project: Building the Family Directory.....</b>	<b>4-14</b>
Open the Report.....	4-14
Create Global Variables .....	4-14
Initialize Global Variables.....	4-15
Inactive Families .....	4-15
Create the new Label.....	4-17
Run the Report.....	4-18
 <b>5: Expanding Letters and Statements.....</b>	 <b>5-1</b>
<b>Overview.....</b>	<b>5-1</b>
IF statements.....	5-1
IF statement format.....	5-2
Using ELSE.....	5-3
Using AND and OR .....	5-4
<b>Multiple Languages .....</b>	<b>5-4</b>
Inside Address Style.....	5-5
Modify the Text of the Letter .....	5-6
Closing Style .....	5-6
<b>Embedded Lists.....</b>	<b>5-8</b>
<b>Exercise.....</b>	<b>5-10</b>

<b>6: Scripting Language .....</b>	<b>6-1</b>
<b>Overview.....</b>	<b>6-1</b>
Accessing the Script .....	6-1
Events.....	6-2
Objects.....	6-3
<b>Additional Code Statements .....</b>	<b>6-3</b>
With Statements .....	6-4
Try-Finally and Try-Except statements .....	6-4
Exit, Break, Continue and Goto statements.....	6-5
More Comments Options .....	6-6
Detecting Errors .....	6-6
Common Problems and Things to Help.....	6-7
<b>Notes.....</b>	<b>6-8</b>
 <b>7: Adding and Accessing Screens .....</b>	 <b>7-1</b>
<b>Overview.....</b>	<b>7-1</b>
Simple Messages and Questions.....	7-1
Screen routines .....	7-2
Screen Objects.....	7-5
<b>Create the New Screen .....</b>	<b>7-6</b>
<b>Saving/Loading the Screen Information .....</b>	<b>7-7</b>
<b>Communicating with the Report .....</b>	<b>7-8</b>
<b>Other Objects.....</b>	<b>7-11</b>
<b>Positioning objects .....</b>	<b>7-12</b>
 <b>8: Special Sorts .....</b>	 <b>8-1</b>
<b>Overview.....</b>	<b>8-1</b>
SelectTbl.....	8-1
MailTbl.....	8-1
<b>Memory Tables .....</b>	<b>8-3</b>
Declare the Variables .....	8-3
Initialize the Variables.....	8-3
Assign the Variables.....	8-4
Attach the Tables.....	8-5
Build the Table .....	8-5
<b>Database Tables .....</b>	<b>8-8</b>
 <b>9: Exporting and Importing Data.....</b>	 <b>9-1</b>
<b>Overview.....</b>	<b>9-1</b>
<b>Export Events.....</b>	<b>9-1</b>
Simple Export.....	9-1
Improving the Export .....	9-2
Other File Formats.....	9-3
Adding a Summary.....	9-5
Exporting to OLE .....	9-6
<b>Importing Data.....</b>	<b>9-7</b>
<b>Data Manipulation.....</b>	<b>9-9</b>
<b>Notes.....</b>	<b>9-10</b>
 <b>10: Appendix .....</b>	 <b>10-1</b>
<b>Overview.....</b>	<b>10-1</b>
UDR Options .....	10-1

Formation Embedded Lists.....	10-2
<b>Built-in Routines:.....</b>	<b>10-2</b>
String Operations:.....	10-2
Simple Math Operations.....	10-2
Math Routines .....	10-3
Date / Time Operations .....	10-3
Conversions .....	10-3
Formatting .....	10-3
Boolean Operations .....	10-4
PDS built-in Routines.....	10-4
PDS Scripting Routines.....	10-4
Member Type Values: .....	10-4
Fund History Activity Types: .....	10-5
Field Listings.....	10-5
Relational Tables .....	10-5
<b>Church Office Field Names.....</b>	<b>10-6</b>
<b>Formation Office Field Names.....</b>	<b>10-22</b>
<b>Ledger, Ledger/Payroll Field Names .....</b>	<b>10-47</b>
<b>Facility Scheduler Field Names .....</b>	<b>10-75</b>
<b>PDS Office Table Relationships.....</b>	<b>10-78</b>

# 1: Introduction

## Overview

The Advanced Report Writer manual was written to help those users that wish to create their own reports, beyond the Simple Reports created with **Add | Simple Reports**.

The Advanced Report Writer utility allows the user to create reports from the ground up, using elements of report design and programming. You will be able to use programming commands imbedded in the report (“imbedded code”) to control the information to be printed, and the format of that information. You will also be able to use Pascal “scripting” language to create additional layout screens and control events outside of the report.

Therefore, this manual will assume that the user is familiar with the PDS Office Suite programs, the Ledger and Payroll programs for Windows, or the Facility Scheduler for Windows. It also assumes the user is familiar with basic mouse operation, such as double-click and right-click; and with standard Windows menus, toolbars and command buttons.

Although some programming knowledge is helpful, it is not required to create ARW reports.

This chapter shows how to create a new report, and what’s in the INI and UDR files.

### Copy vs. Add

An Advanced Report Writer report (ARW) can be created in two ways from the Reports screen:

- Select a report category, and click on **Add | Advanced Reports | Report**.
- Select a predefined report or ARW report and click on **Copy**.

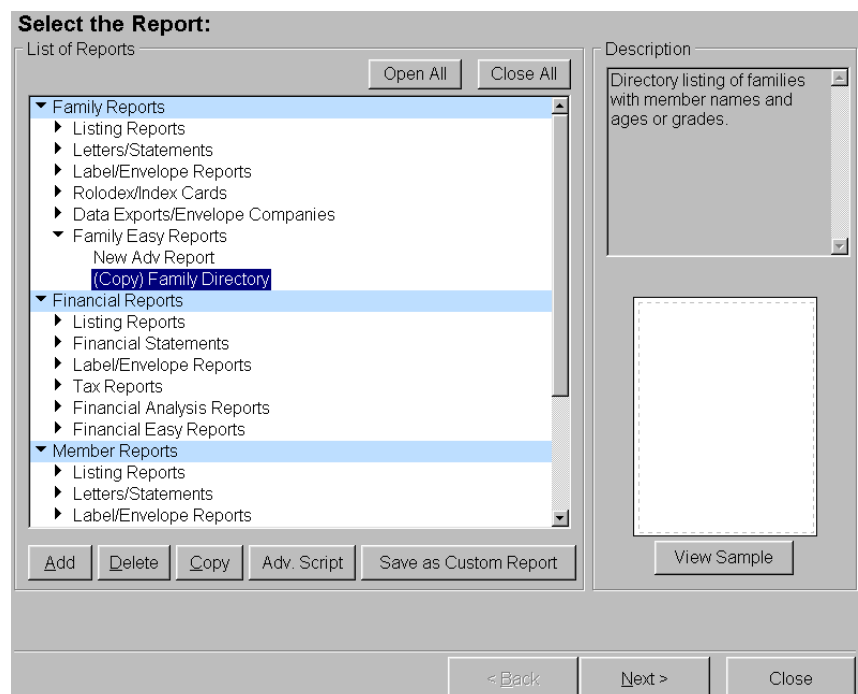
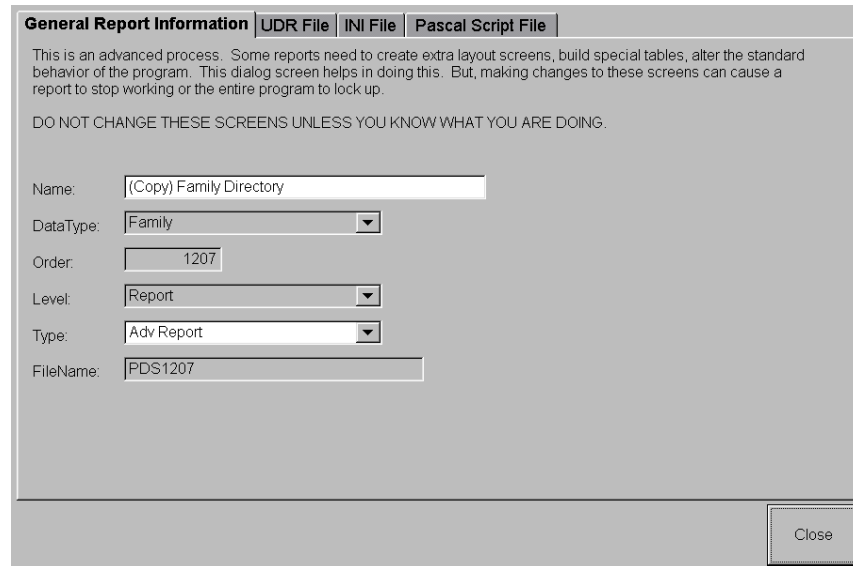


Fig. 1-1

Only some of the predefined reports were originally created as ARW reports, and only those will allow you into the ARW editor when copied. The simplest way to tell is to make a copy and click on **Adv. Script**. If the **Type** is Adv. Report, then it is an ARW report; if the **Type** is anything else, then it is an easy report.



**General Report Information** | UDR File | INI File | Pascal Script File

This is an advanced process. Some reports need to create extra layout screens, build special tables, alter the standard behavior of the program. This dialog screen helps in doing this. But, making changes to these screens can cause a report to stop working or the entire program to lock up.

DO NOT CHANGE THESE SCREENS UNLESS YOU KNOW WHAT YOU ARE DOING.

Name: (Copy) Family Directory

DataType: Family

Order: 1207

Level: Report

Type: Adv Report

FileName: PDS1207

Close

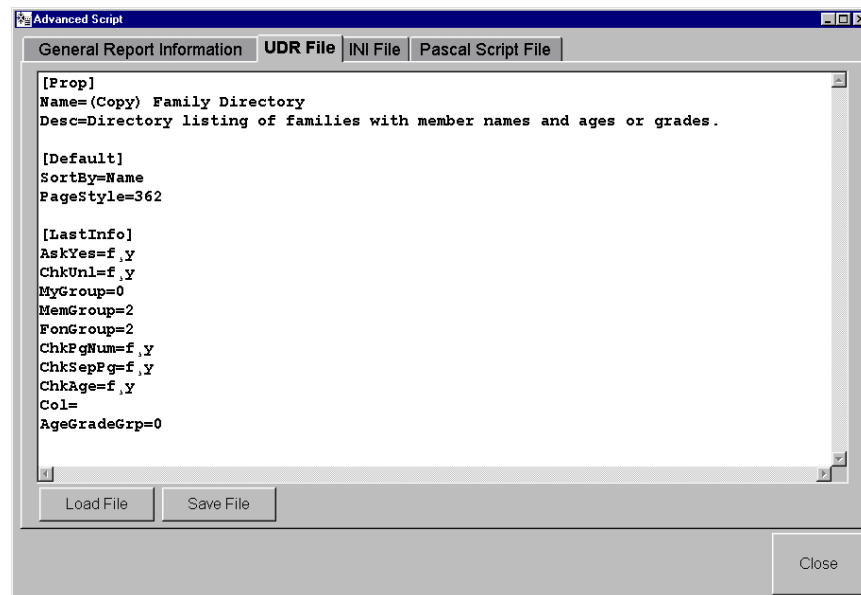
Fig. 1-2

## UDR files

Each ARW report has a UDR file that gives information about the report. To see the UDR file, highlight the report and click on **Adv. Script** | **UDR File** | **Load File**.

Each section of information is preceded by the type in square brackets, such as [Prop] for the Properties section.

Each line is a different setting, with the format of **Object=setting**; for example, Name=Family Directory.



**Advanced Script**

**General Report Information** | UDR File | INI File | Pascal Script File

```
[Prop]
Name=(Copy) Family Directory
Desc=Directory listing of families with member names and ages or grades.

[Default]
SortBy=Name
PageStyle=362

[LastInfo]
AskYes=f,y
ChkUnl=f,y
MyGroup=0
MemGroup=2
FonGroup=2
ChkPgNum=f,y
ChkSepPg=f,y
ChkAge=f,y
Col=
AgeGradeGrp=0
```

Load File Save File

Close

Fig. 1-3



### Properties [Prop]

- Name = report name
- Desc = report description
- SkipMergeUp = 0

### Default [Default]

In the Default Section, you can assign values that you want the report to use the first time it is run.

- SortBy = Name, ID Number, or Second ID Number.
- ForceSortBy = Name - the default sort is by Name and only name can be chosen.
- Landscape = 1 - turn on landscape.
- DisableScaling = 1 - The program will not scale the report. It works best for forms and certificates.
- PageStyle = xxx
- MarginStyle = xxx
- LHStyle = xxx – Letterhead style
- DateFmtStyle = xxx – Date Format style
- InsideAddrStyle = xxx – Inside Address style
- ClosingStyle = xxx
- LabelStyle = xxx

After the report is run, the INI file holds the values used. Those values can in turn be used to set XXX, which the report will then use as the default for that style. For example, PageStyle=362 will default the report to Compress Print.

### Last Info [LastInfo]

This section holds initialization variables and values used by the Pascal Script File. These parameters are discussed in chapter 4.

### Fields [Fields]

Listing reports contain a line for each field and its settings.

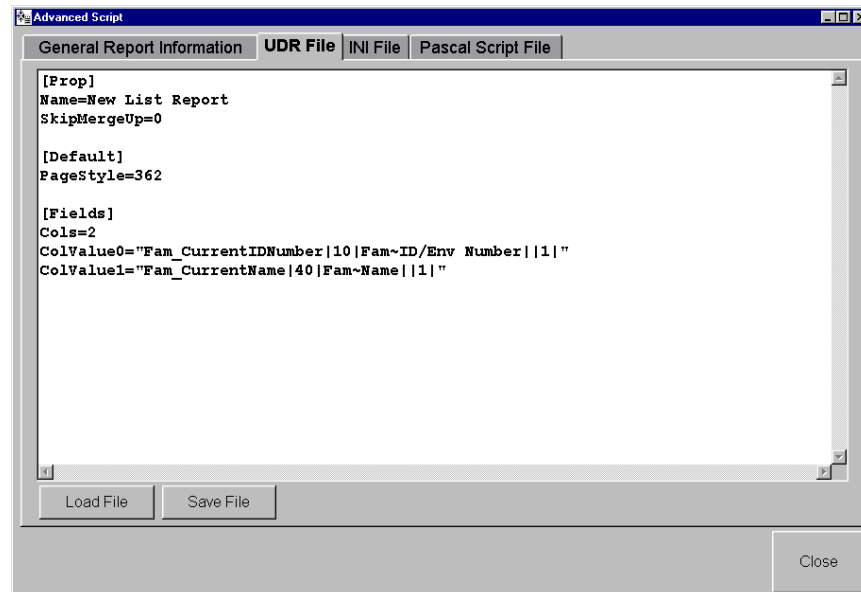


Fig. 1-4

Cols= number of columns

ColValue# = the values for each column in the following format:

field | width/indent | heading | | indent total or "1" | label

For columns that are "New Column", "Indent total" is equal to 1 (one). For columns that are "Same Column and Line", "Indent total" is the total width of the column, including widths from the main New Column and indent values from Same Column and Line.

For example, column 1 is a New Column, width=10. Column 2 is a Same Column and Line, width 3. The ColValues would be:

ColValue1=field | 10 | heading | | 1 | label

ColValue2=field | 3 | heading | | 13 | label

The tilde (~) represents a soft carriage return in the Heading.

## INI files

Each report creates an INI file the first time it is run. It saves and updates the printer, report, style and other settings each time the report is run.

To access the INI file, highlight the report and click on **Adv. Script | INI File | Load File**.

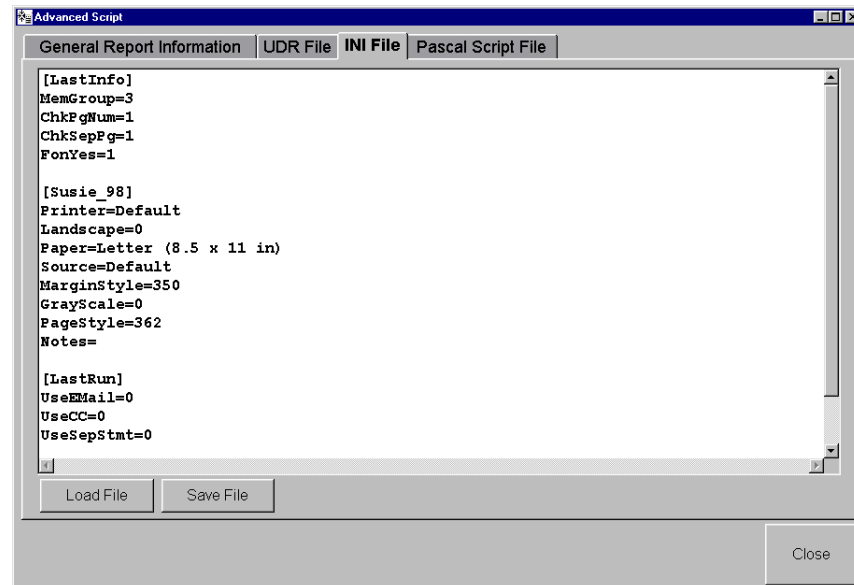


Fig. I-5

### Computer Name

One of the sections will be the computer name, such as [Susie\_98], which saves the last printer settings used by this computer; including which printer, orientation (landscape=0 means portrait), paper size, paper source, margin style, gray scale, page style and notes.

### Last Info [LastInfo]

Contains variables and values used in the Pascal Script File from the last time the report was run.

### Last Run [LastRun]

This section saves the built-in selection settings from the last run of the report, such as UseSepStmt (use separate statement) and UseEMail (prefer email).

## 2: Report Basics

### Overview

This chapter discusses the basics of the Advanced Report Writer (ARW) editor, the workspace, basic controls and tools, and how to use templates.

The ARW editor has two parts – the toolbars...

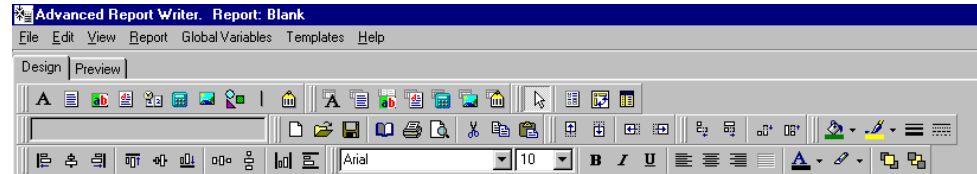


Fig. 2-1

...and the workspace.

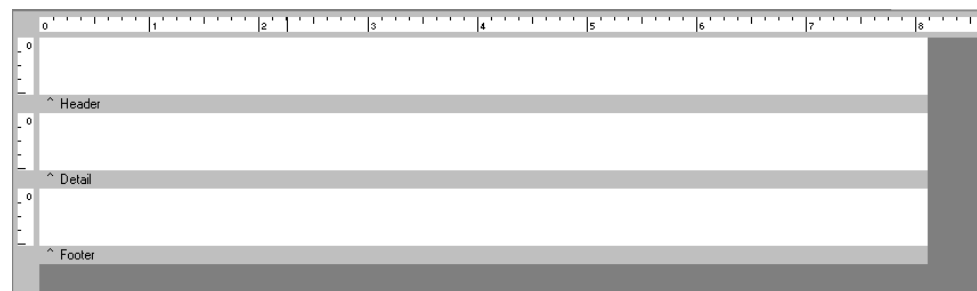


Fig. 2-2

### The Work Space

When a new report is opened, the workspace is divided into three bands: Header, Detail and Footer.

Objects in the Header will appear at the beginning of each page. Objects in the Footer will appear at the end of every page. Objects in the Detail band print once for each item inside this band.

Two other bands, Title and Summary, can be added by clicking on **Report | Title** and **Report | Summary**. The Title will appear at the very top of the report, before the Header on the first page. The Summary will appear at the very end of the report, before the Footer on the last page.

**Title** – prints at the top of the report.

**Header** – prints at the top of each page.

**Detail** – prints once for each item inside.

**Footer** – prints at the bottom of each page.

**Summary** – prints at the end of the report, before the footer on the last page.

Workspace bands can be made smaller or larger by clicking and pulling the gray bars up or down. The band's height can be set to re-adjust to match its contents or stay static. Right-click on the gray bar, and choose:

- **DynamicHeight:** When printing, the vertical size of the band fluctuates depending on the objects inside. This can be used to eliminate 'white space' between families printed.
- **StaticHeight:** When printing, the vertical size of the band remains constant, regardless of the size of the objects inside. This can be used to create extra 'white space' in the report between families printed.

## Toolbars

The toolbars consist of a mix of objects and formatting elements. An object is something that can be placed in the report, such as a label, data field, variable, or image. Formatting elements are ways to modify the objects, such as bold, italic, and font color; or change the object's position, such as nudge, left align, and center.

The toolbars can be turned off or on by clicking on **View | Toolbars** and select or unselect the toolbar.

## Objects

An object is a piece of information that you want to appear in the report. They are divided into two categories: generic and data-sensitive.

The following are generic objects. They are not connected to particular data fields.



Label: an individual piece of text, like a heading or title.



Memo: large blocks of text.



System Variables: such as date, time, and page numbers.



Variables: used to calculate numeric and string values.



Image: used to place images into the certificate, such as pictures, letterheads or signatures.



Shapes: used to create shapes (boxes, circles, etc.) in the report.



Lines: used to create horizontal lines of various thicknesses and styles.

These objects are data-sensitive. They are similar to the generic objects, except they will show actual program data.



DBText: an individual data field, such as Fam Name.



DBMemo: large data fields, such as phone listings.



DBCalc: specific calculations, such as Count or Sum.



DBImage: an image from the data, such as the Family or Member picture.

## Adding an Object

Each object is added the same way.

- Click on the appropriate button.
- Click in the workspace where you want the object to appear.

Objects are deleted by selecting them and pressing the **Del** key on your keyboard.

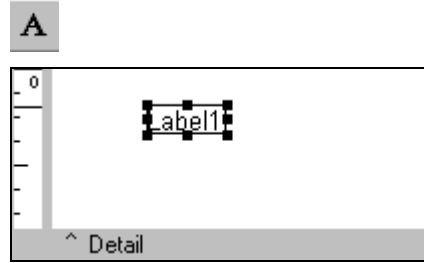


Fig. 2-3

## Labels and Memos



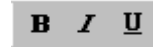
A label is an embedded piece of text that does not change from family to family when printing several families. An example would be a label describing a field.

Type the text of the label in the input box.



Fig. 2-4

Use these buttons to make the text **bold**, *italic*, or underlined.



Use these buttons to change the color of the text or the background.



Use these buttons to change the font style and size.



A memo is a larger box of text, used for paragraphs or other situations where you need multiple lines of text.

Once a memo object is added, right-click on the object and choose **Lines...**; this will open the Memo Editor for you to type in the lines of text.

**Load:** click here to read in text from an outside data source or TXT file.

**Save:** click here to save the memo in a TXT file.

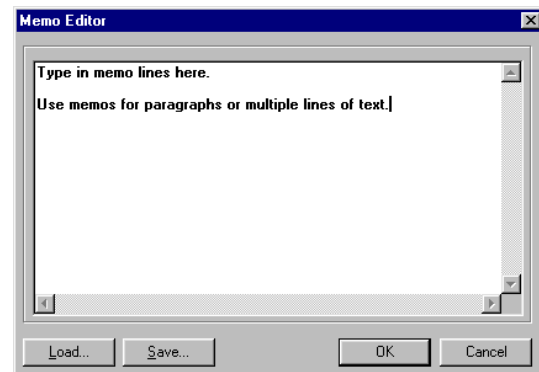


Fig. 2-5

## Images



You can insert images into your report with the Image button.

Once an image object is added, right-click on the object and choose **Picture...** to open the Windows browse dialog. From there, choose your picture image.

Images such as scanned letterheads, signatures, clipart and other picture files can be inserted this way.

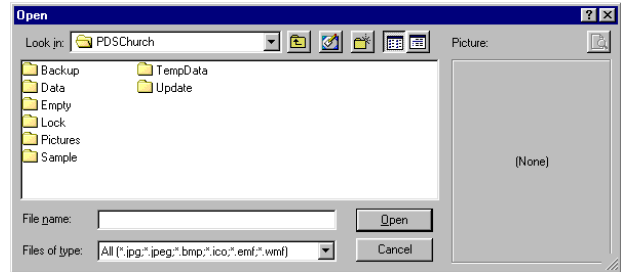


Fig. 2-6

## System Variables



System variables are fields such as date and time that can be added to your report with the System Variable button.

Once a system variable object is added, choose what kind of variable from the input box.

### System Variables:

Date  
DateTime  
DocumentName  
PrintDateTime  
PageCount  
PageSet  
PageSetDesc  
PageNo  
PageNoDesc  
Time

You can change the format of the variable by right-clicking on the object and choosing **DisplayFormat...**. This will provide a list of display options for that system variable.

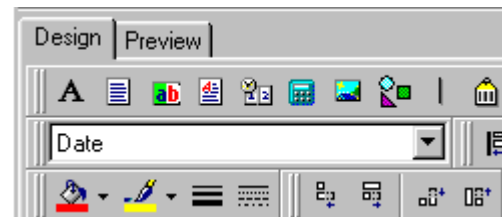


Fig. 2-7

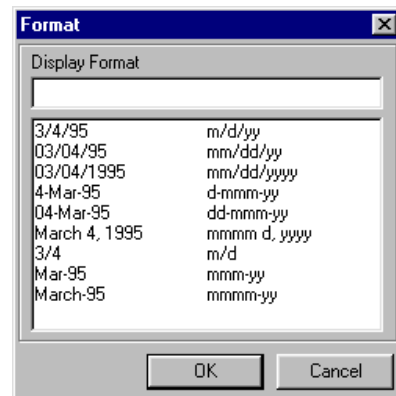
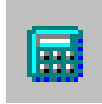


Fig. 2-8

## Variables



Variables are numeric or string calculations that do not exist elsewhere in the data or in the system variables.

Variables are used to create values from other values or fields. For example, a directory listing usually has the first letter of the last name at the top of the column. That letter is a variable; calculated from the name.

Once a variable is added, choose what type of variable from the input box.

### Variable types:

Boolean  
Date  
Time  
DateTime  
Integer  
Single  
Double  
Extended  
Currency  
Char  
String



Fig. 2-9

You can set the calculation of the variable by right-clicking on the object and choosing **Calculations...**. This will provide a Pascal programming dialog box. For more info about working with variables, refer to chapter 4.

## Shapes and Lines



You can insert shapes and lines into your report with the Shapes and Lines buttons.

Once a shape object is added, choose what kind of shape from the input box. Choices are rectangle, square, rounded rectangle or square, circle, and ellipse.



Fig. 2-10

Once a line is added, choose where the line appears from the input box. Choices are top, bottom, left, or right of the object.

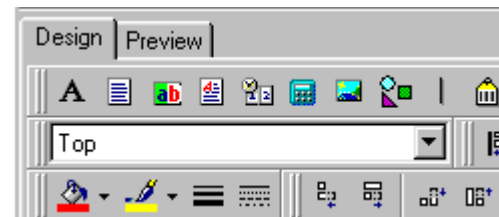


Fig. 2-11



Use these buttons to choose the color of the shape, the color of the line, the line thickness and the line style.

## Data Fields



The most common object to input into a report is a DBText, or data-sensitive, field. These fields are directly connected to your database, and will display the family or member information.

Once you add a data field, choose which field category (also called 'data pipeline') and field name from the input box.

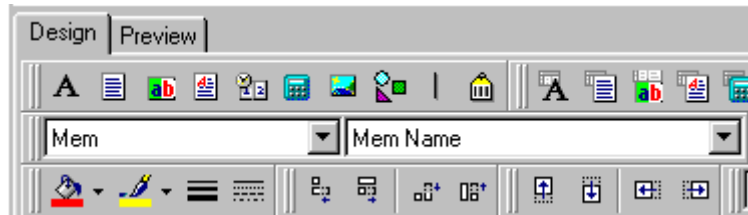


Fig. 2-12

The list of categories and fields is the same as that for Listing reports, and can be found the User's Manual.

Data fields are the objects that get information from the program and display it in the report. For example, if you wanted to show the member's name, baptism date, and sponsors – you would add a data field for each, choosing Mem Mem Name, Sac Baptism Date, and Sac Baptism Sponsor List.

The data itself is usually larger than the object size created. You can right-click on the object and choose **Autosize** to set the object size to be the same as whatever data is displayed.

If the data field normally has a special format, such as dates and phone numbers, right-click on the object and choose **DisplayFormat...** to choose the appropriate format for the data.



A DBMemo object is the same as a DBText field only larger, designed for fields that use multiple lines of text, such as phone listings, keywords and talent/ministry lists.

## Data Calculations



A DBCalc field is used to count or total a data-sensitive number field, such as age or financial fields. It is normally added to the Summary band of the workspace, so that the program can calculate based on all of the data looped through in the Detail band.

Once the DBCalc object is added, choose the field category and data field just like a DBText field. This is the field that the calculations will be based on.



Fig. 2-13



Then, right-click on the object and choose **Calculations...**

Choose the **Calculation Type**: Sum, Count, Maximum, Minimum, or Average.

When the report is run, the program will display the result of that calculation on the field selected.

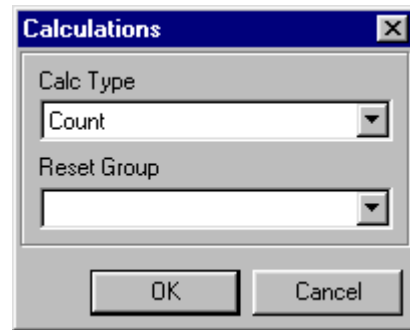


Fig. 2-14

## Data Images



A DBImage object is used to show a data-connected picture, such as a Member Picture or Family Picture.

Once the DBImage object is added, choose the field category and data field just like a DBText field. This should be a picture-based field, such as Mem Picture.

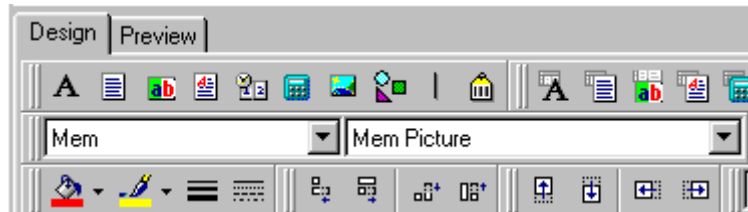


Fig. 2-15

## Selecting Objects

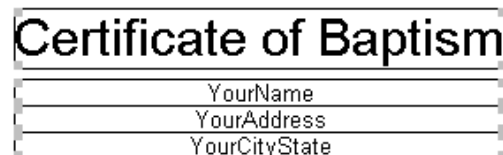
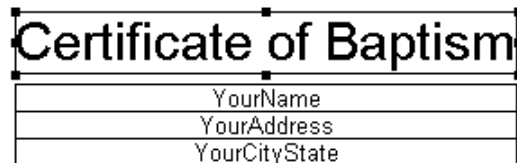
The simplest way to select an object that has already been added is to click on the object. This creates a dark border around it, with small black boxes in the corners and sides.

On occasion, you may need to select several objects at once.

This is done by holding the Shift key as you click on successive objects.

You can also click near the objects and create a 'selection box' around them. When you let go of the mouse, all of the items in the box are selected. The selected objects will now have small gray boxes in the corners.

To de-select an object or objects, click anywhere in the empty portion of the workspace.



## Right Click on Objects

### Embedded objects:

Some objects are embedded within others. If this is the case, when you try to select the embedded object, you are actually selecting the entire thing.

In order to select an object embedded within another, hold down the CTRL key before using the mouse. If you accidentally select the outside object, hold down the SHIFT key and click on that object to deselect it.

You can right click on any object to change options or properties of the object. An ellipsis [...] in the property name indicates additional controls.

**Bring to Front/Send to Back:** All objects. Move this object in front of or in back of other objects.

**Double/Single:** Lines only. Change the line to a single or double line.

**Autosize:** Automatically re-sizes this object horizontally to match its contents. See also Stretch.

**BlankWhenZero:** If the object's value is zero, print a blank instead of the digit "0".

**Calculations...:** For DB fields, choose the type of calculation (Sum, Count, Max, Min, Average). For Variables, modify the value or formula.

**CalcOrder:** Indicate what order groups are to be calculated.

**CharWrap:** Break words in the middle of the word when wrapping. See also WordWrap.

**DisplayFormat...:** For DBText fields, set the format of the field, i.e. Date format, number format, etc.

**ForceJustify:** For Memo fields, justify the text to the left or right.

**KeepTogether:** Do not split words when wrapping text inside of a memo box.

**Lines...:** In a Memo field, the text of the Memo.

**LookAhead:** For Variables, DBCalc fields and System Variables, when you need to display the result of a calculation or variable before the objects that are used in it. For example, displaying the DBCalc field Count in the Header, instead of the Footer or Summary. To use this ability, you must also set the report for two passes through the data. This is set under **Report | Pass Setting | Two Pass**.

**ParentHeight:** For Lines, make the object the same height as the workspace.

**ParentWidth:** Make this object the same width as the workspace.

**Position...:** Where is the object positioned in the workspace, including width and height.

**ReprintOnOverflow:** When set to Yes, if any Stretched object would go to a 2<sup>nd</sup> page, this object will be also be reprinted on that 2<sup>nd</sup> page.

**ReprintOnSubsequent:** Reprint this object the first time it occurs on subsequent pages.

**ResetGroup...:** Each time the value of the group's breaking field changes, the calculated value of this object is reset to zero, and the calculation begins again.

**ShiftRelativeTo...:** Keep the position of this object static with respect to another object. For memos and sub-reports.

**ShiftwithParent:** Keep the position of this object static with respect to the workspace band it is in. For memos and sub-reports.

**StretchwithParent:** Automatically re-size this object horizontally to match the workspace.

**Stretch:** Automatically re-size this object vertically to match its contents. See also Autosize.

**SuppressRepeatedValues:** If the value to be displayed for this variable or field is the same as the last value displayed for this variable/field, don't display it – unless the data has overflowed to the next page and ReprintOnSubsequent is set to Yes.

**Timing...:** Variables only. Used to determine when this variable will be calculated relative to other variables.

**Transparent:** Make this object transparent, so that objects behind it can be seen.

**Visible:** Make this object visible. You would turn this off if you are using a variable for calculations but don't want to see the variable.

**Wordwrap:** Do not break words in the middle of the word. Wrap whole words only. See also CharWrap.

## Moving Objects

Once you have added an object, and have selected it, the next step is to move it where it belongs. To move an object, simply click on it, hold the mouse-button down, and drag it to where you want it to go, then release the mouse button. The object is now repositioned.

In order to line up an object with those near it, the ARW editor has a series of minor adjustment buttons. To use these buttons, select the object(s), then click on the appropriate button.

**Nudge:** These four buttons are used to move object(s) exactly one pixel up, down, left or right.



Nudge Up



Nudge Down



Nudge Left



Nudge Right

Nudging can also be accomplished by selecting an object, holding down the CTRL key and using the up, down, left and right arrow keys on your keyboard.

**Grow/Shrink:** These four buttons are used to grow or shrink several objects to the same size.



Shrink Width to Smallest: reduce the width of each selected object to match the narrowest one.



Grow Width to Largest: increase the width of each selected object to match the widest one.



Shrink Height to Smallest: lower the height of each selected object to match the shortest one.



Grow Height to Largest: raise the height of each selected object to match the tallest one.

**Alignment/Spacing:** These ten buttons are for aligning or spacing several objects at once. Objects will align with the first object selected.



Align Left: This button aligns the selected objects along their left edge.



Align Right: This button aligns objects along their right edge.



Align Top: This button aligns objects along their top edge.



Align Bottom: This button aligns objects along their bottom edge.



Align Middle: This button aligns objects in a vertical stack around a common middle.



Align Center: This button aligns side-by-side objects along a common center.



Space Vertically: This button puts an equal amount of vertical space between each selected object.



Space Horizontally: This button puts an equal amount of horizontal space between each selected object.



Center Horizontally: This button centers objects between the left and right margins.



Center Vertically: This button centers objects between the top and bottom margins.

## Report Tree and Data Tree

The Report Tree and Data Tree are useful tools for finding your objects and associating fields with them.

### Report Tree

The Report Tree shows an outline of the report and an outline of the objects.

To see the Report Tree, click on **View | Toolbars | Report Tree**.

The top portion is the Report Outline, a breakdown of the different sub-reports in use.

The bottom portion is the Object Outline, a breakdown of each band and the objects within.

Objects and sub-reports can be renamed. Right-click on the name, ie. DBText1, and type in a more descriptive name, ie. FamilyName. You cannot use spaces in the name.

The Report Tree can be “docked” to the side of the screen by dragging the dialog box all the way to the left. Otherwise, the Report Tree “floats” over your work space.

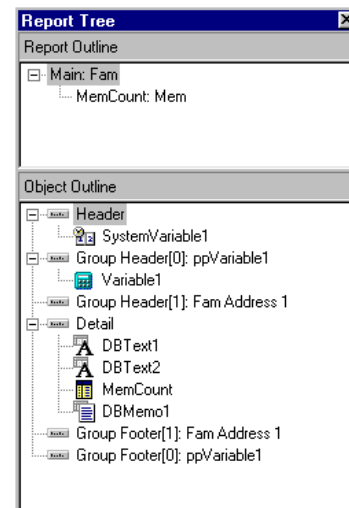


Fig. 2-16

## Data Tree

The Data Tree shows information about the fields in two tabs, Data and Layout.

The Data tab shows the field categories and fields available.

The top portion shows an outline of the data categories, ie. Fam, FamKeyword, etc.

The bottom portion shows a listing of the fields that belong to the chosen category. The field's Name, Type and Size are shown.

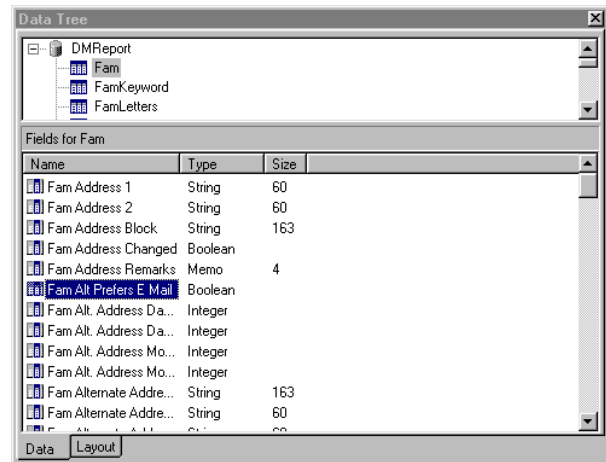


Fig. 2-17

You can drag and drop a data field directly from the Data Tree into your report. The program automatically creates a label for the field, which can be modified.

The **Layout** tab is used to indicate how and if a label is to be included when a data field is dropped from the data tree.

Choose **All** to create both a label and the field.

Choose **Vertical** or **Tabular** style.

Choose **Label** or **Field** to create one or the other. Use the **Font** and **A** buttons to change the font style and color.

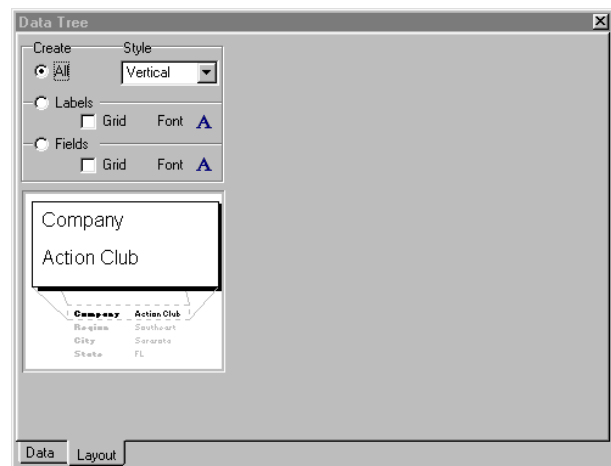


Fig. 2-18

The Data Tree can be “docked” in the same way as the Report Tree.

# Templates

The Advanced Report Writer has report format templates that you can use to quickly setup a report with predefined headers and footers.

To access the Templates, on the Title Bar, click on **Templates**.

## PDS Easy List Template

The PDS Easy List template sets up your ARW report with the same header and footer information that the predefined Listing reports have.

Click on **Templates | PDS Easy List**. The program opens up the same field selection dialog used by View Listings and Easy Lists reports.

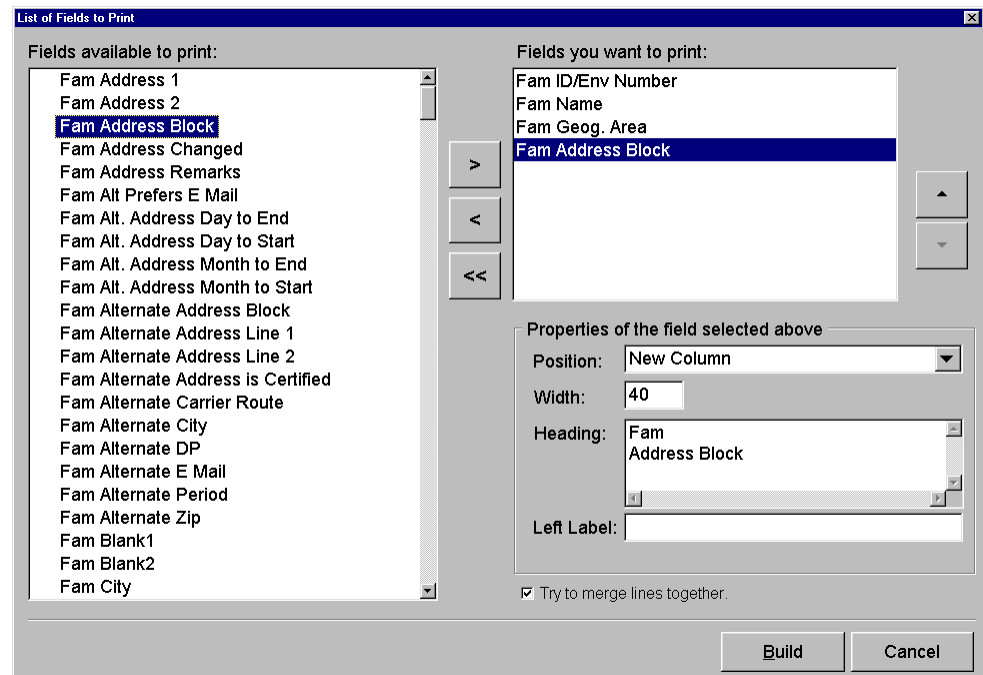


Fig. 2-19

Once you have chosen your fields, click on **Build**.

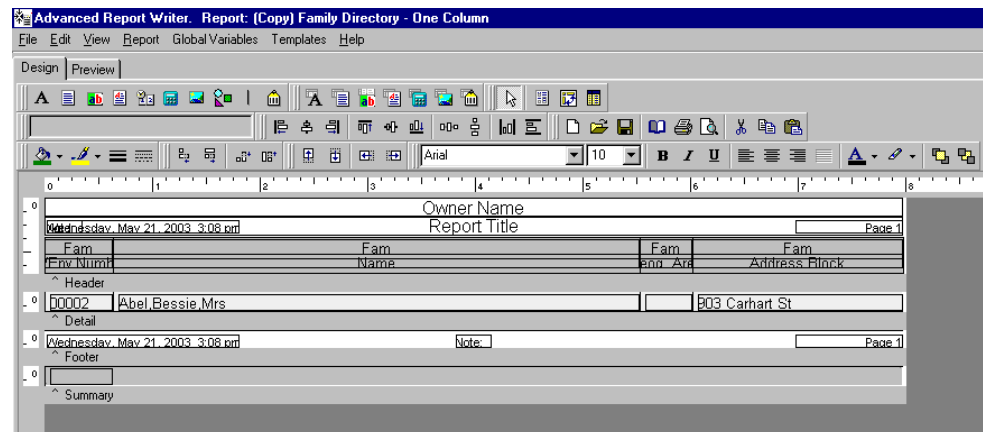


Fig. 2-20

The program creates the Header and Footer bands to match the predefined Listing reports. The fields chosen are placed in the Detail band in columns, matching the widths chosen. A Summary band is added to hold the Count system variable.

Any of these items can be changed, rearranged, or removed. The template is simply providing a familiar starting point.

## PDS Header/ Footer Template

The PDS Header/Footer template simply places the Listing style Header and Footer around whatever Detail bands you already have created.

Click on **Templates | PDS Header/Footer**.

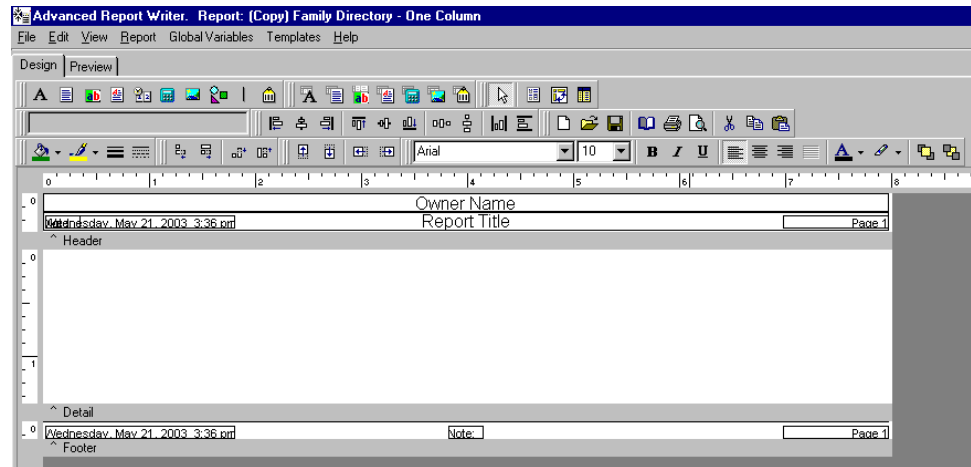


Fig. 2-21

The template Header and Footer consists of these fields:

**OwnerTitle label:** the church name from the License Information screen.

**ReportTitle label:** the title of the report, taken from the Name and Description step of the Report Wizard.

**THLine:** a Horizontal line positioned at the Top.

The **Date**, **Page Number** (system variables) and **Note** (a label) can be positioned either in the Header or Footer, depending on the Page Style. The template allots a field for either.

**Line1:** an unformatted horizontal line.

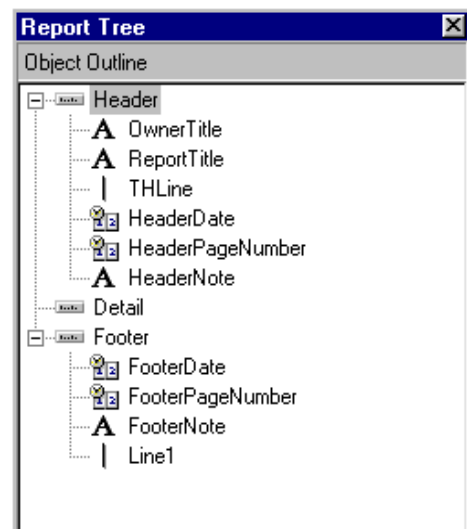


Fig. 2-22

## Fonts and Page Style

All regular objects (labels, memos, system variables, calculated variables) used in an ARW report, whether DB connected or not, can have whatever font you choose. However, sometimes you want those fields to have the same font as dictated in the **Page Style**.

This can be set by right-clicking on an object and choosing **Font Changes...**

You can choose:

**No – Don't Change the font.** Use the font chosen in the ARW.

**Yes – Change the font.** Use the Title, Heading, Detail, or Date/Page font as chosen in the Page Style.



Fig. 2-23

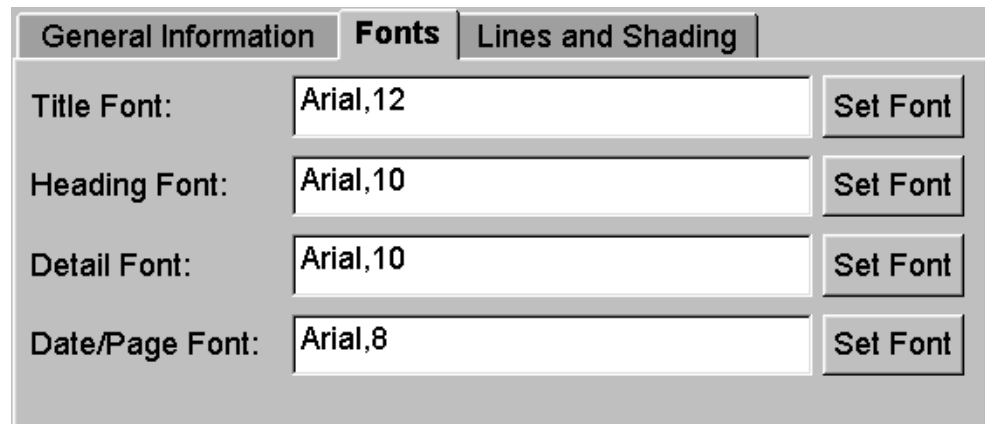


Fig. 2-24



## Preview

Click on the **Preview** tab to see how your report will look, using your data.

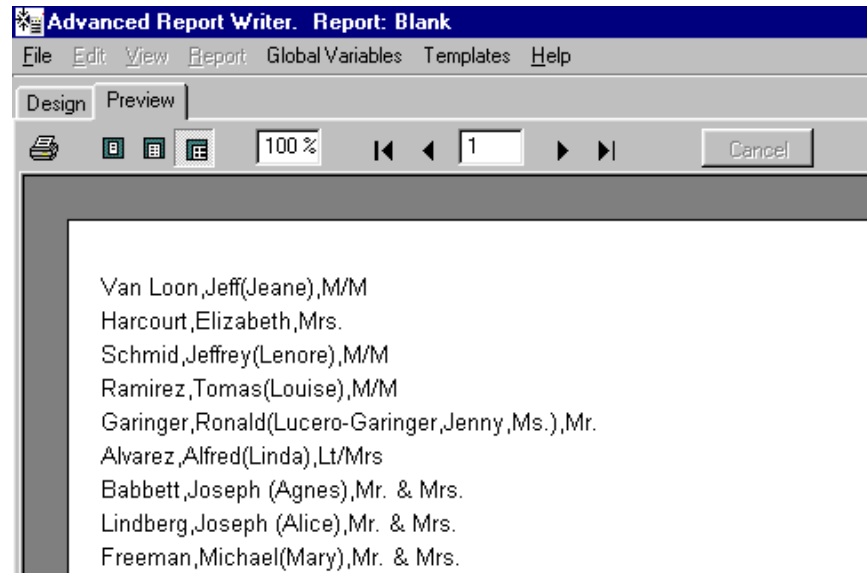


Fig. 2-25

Click on the Printer icon to print the report.

Click on the three sizing buttons to see the report Full Page, Full Width, or Zoom In.

Use the arrow buttons to see different pages of the report.

Click on Cancel to stop the preview while it's being created.

## Exercise

Create a new advanced report, using the PDS Header/Footer template, and the following fields and options:

- Family ID, in bold black text.
- Family Name, in bold blue text, next to the ID.
- Family Address Block, beneath the Family Name, in regular text. (hint: use DBMemo)
- Family Phone List, next to Family Address Block, in regular text. (hint: use DBMemo)
- Family E Mail, beneath Family Address Block, and ShiftWithParent.

In the Summary band, add a Count of families.

Here's how to do it:

1. This will be a Family report. From the Main Menu, click on **Reports | Family | Add | Advanced | Report**. Click on **Next**, and **Next** again to get to the Layout screen. Click on **Modify the Report**.
2. Add the template. Click on **Templates | PDS Header/Footer**.

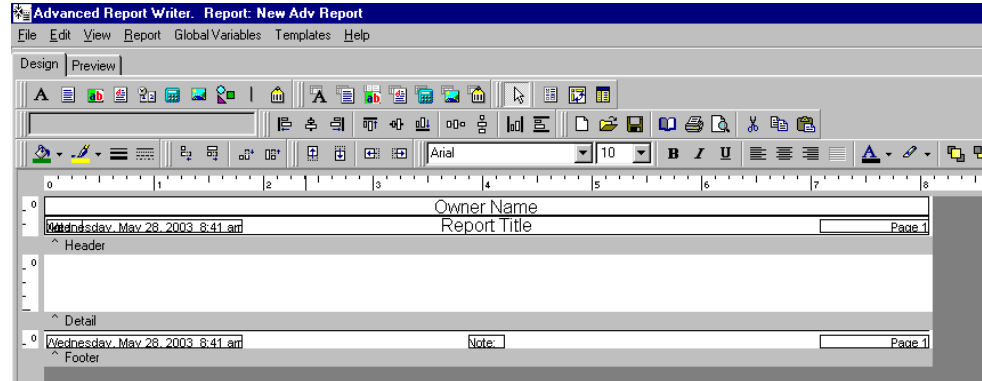


Fig. 2-26

3. Add a **DBText** object in the top left corner of the Detail band. Set it to **Fam Family ID/Env Number**. Make it bold.
4. Add a **DBText** object directly next to the Family ID. Set it to **Fam Fam Name**. Right-click on it and choose **Autosize**. Make it bold and blue.
5. Add a **DBMemo** object directly beneath the Family Name. Set it to **Fam Address Block**. Right-click on it and choose **Stretch**.
6. Add a **DBMemo** object directly next to the Address Block. Set it to **Fam Phone List**. Right-click on it and choose **Stretch**.
7. Add a **DBText** object directly beneath the Address Block. Set it to **Fam E Mail**. Autosize it.

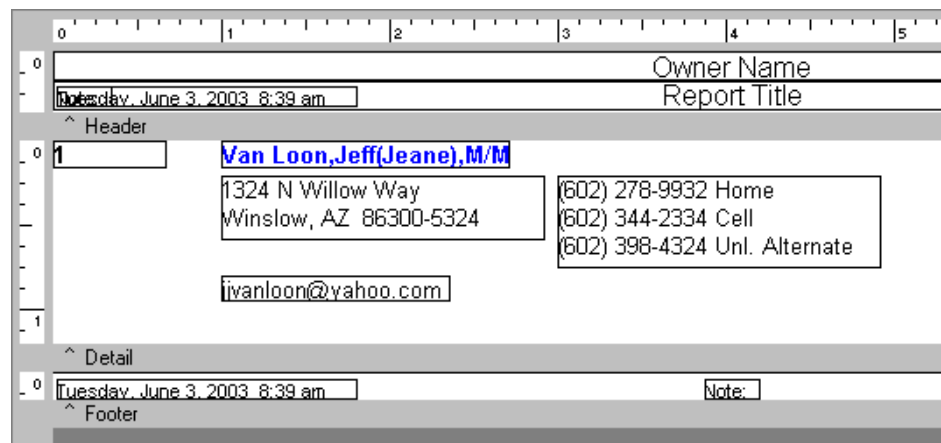


Fig. 2-27

8. Add the Summary band. Click on **Report | Summary**.
9. Add a **DBCalc** object to the Summary band. Right-click on it and choose **Calculations**, then choose **Count**. Set it to **Fam Fam Unique ID**.

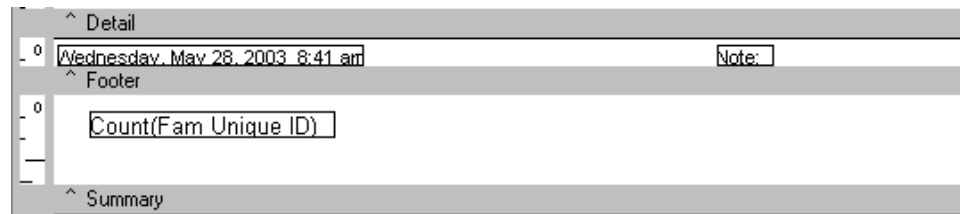


Fig. 2-28

10. Click on **Preview**.

The report should look something like this:

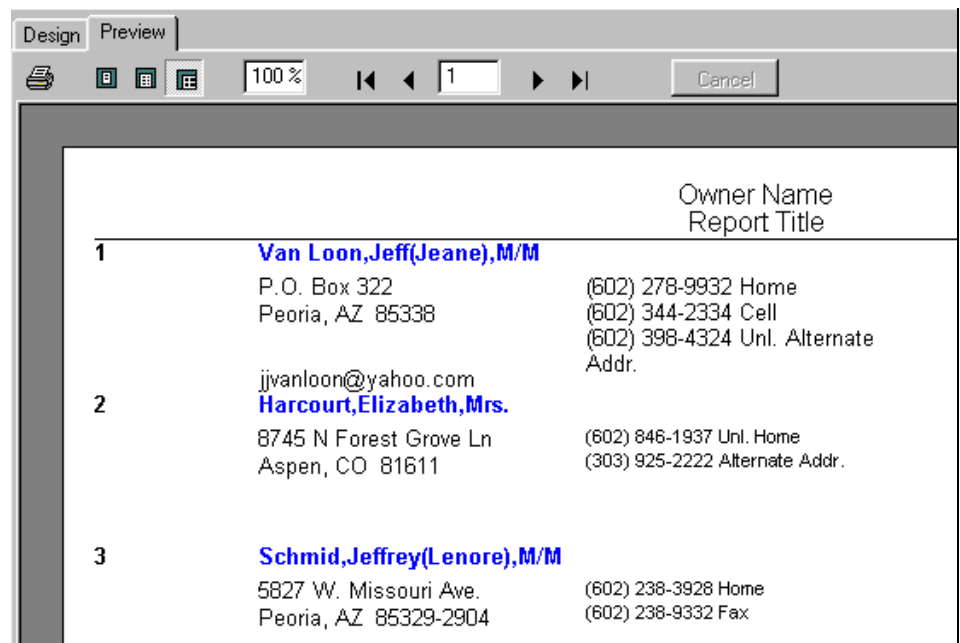


Fig. 2-29

## Notes

## 3: Sub-reports, Groups and Regions

### Overview

This chapter will discuss sub-reports and their properties, groups and regions.

A sub-report is a report within a report; for example, a list of families, with each family showing a list of its members. The family list would be the main report, and the member list would be a sub-report.

A group is a workspace designation that is used to keep everything in the Detail section printed together on a page.

A region is a group of objects that are boxed together and handled as a unit. Everything in the region does as the region does. For example, if the region is Visible, then all objects in the region are Visible; if the region is not visible, the objects within the region are not visible.

### Sub-reports

#### Adding a Sub-report



A sub-report is added just like any other object. Click on the **SubReport** button, which looks like a yellow legal notepad, then click in the workspace.

When a sub-report is added, it is automatically stretched to match the workspace width, has no data pipeline associated with it, and creates a sub-report tab at the bottom of the ARW screen.

The initial name of the sub-report object is **PDSSubReport1: No Data Pipeline Assigned**.

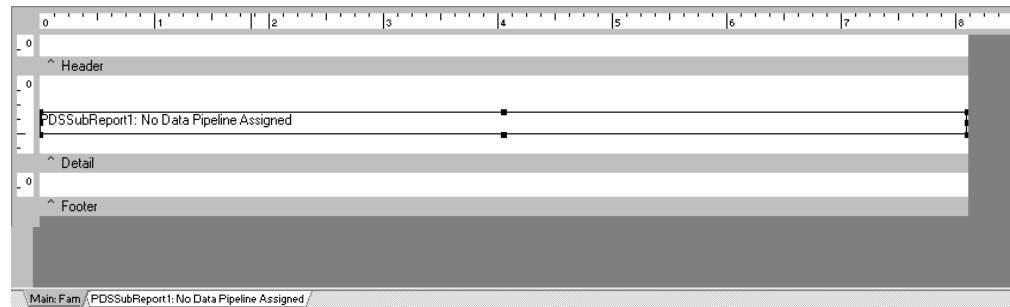


Fig. 3-1

To change the width of a sub-report, right-click on the object and click on **ParentWidth**. This unlocks it from matching the section width, so that it can be resized.

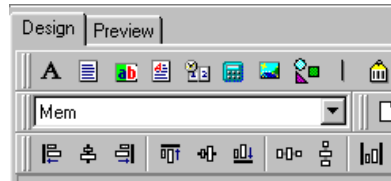
Sub-reports can also be nested inside of other sub-reports.

## Data Pipeline

The Data Pipeline is the data category path that determines which data fields the sub-report can use.

For example, to access the Ministry fields in a family report, you would need a Ministry sub-report inside of a Member sub-report inside of a Family report.

To set the Data Pipeline for a sub-report, select it from the **Data Pipeline Name** drop-down list in the Toolbar.



Once the pipeline is chosen the object name becomes **PDSSubReport1: pipeline**.

Notice that Financial fields are not available through the member pipeline, and vice versa.

Not all sub-reports need a data pipeline. For example, you have a family report that includes Sub-report 1. Sub-report 1 contains some Heading labels and Sub-report 2. Sub-report 2 has a pipeline of Member, but Sub-report 1 has no pipeline – it is an intermediate level designed to hold the Headings.

Also, the Sub-report does not need a data pipeline if it is using fields found in the Main data pipeline.

### The Data Pipeline hierarchy:

- Family
  - Family Keywords
  - Family Letters
  - Family Phone
  - Members
    - Attendance
    - Background Check
    - E-mail
    - Member Keywords
    - Member Letters
    - Member Ministries
    - Member Other
      - Requirements
    - Member Phones
    - Member Talents
    - Sacraments
      - Sacrament List
      - Sacrament Extra Info
      - Sacrament Sponsor
      - Sacrament General
  - Financial
  - Other

## Modifying a Sub-report

To modify the contents of the sub-report, click on the tab at the bottom of the workspace – **PDSSubReport1: pipeline**.



A sub-report begins with Title, Detail and Summary bands. Usually the Title and Summary bands are not used, and can be removed by clicking on **Report | Title** and **Report | Summary**.



Add objects to the sub-report in the same way that you do to the main report. Anything placed in the Detail band of the sub-report will be repeated once for each item. That process will be repeated for each item in the Detail band of the Main report.

If you are using several sub-reports, you may want to use the Report Tree described on page 2-10.

## Exercise

To illustrate sub-reports, let's create a family report, with a member sub-report.

1. Create a new ARW report.
2. In the Detail band of the Main report, put in a DBText object, and connect it to Fam Fam Name. Right-click on Fam Name and check Autosize.
3. Also in the Detail band, put in a Sub-report beneath the Fam Name.

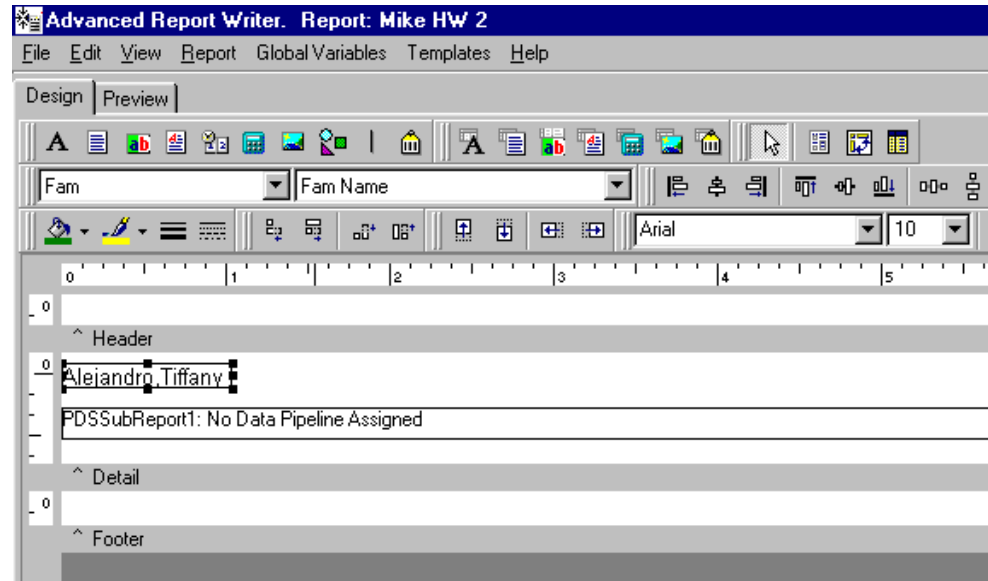


Fig. 3-2

4. Select the sub-report and assign the data pipeline with the **Data Pipeline Name** drop-down list on the Toolbar.
5. Click on the sub-report's tab at the bottom of the workspace.
6. Turn off the Title and Summary bands by clicking on **Report | Title** and **Report | Summary**.
7. In the Detail band of the Sub-report, put in a DBText object, and connect it to Mem Mem Name. Autosize it and move it 0.5 inch to the right. Re-size the Detail band to match the Mem Name.

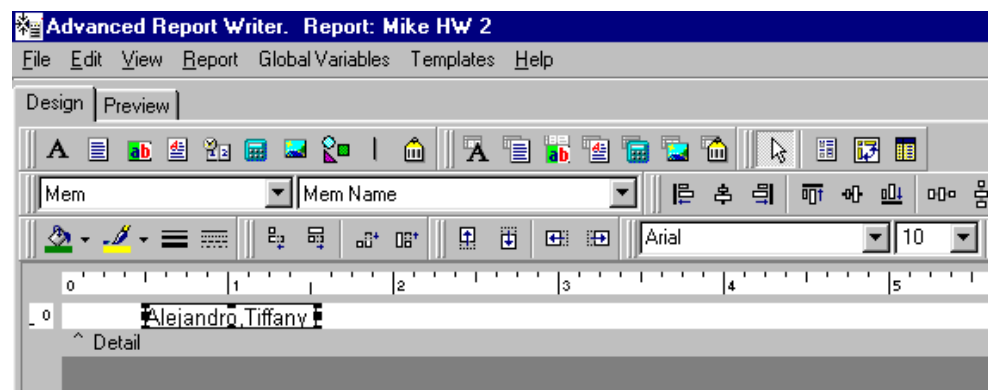


Fig. 3-3

8. Click on **Preview**.

The report will look similar to this:

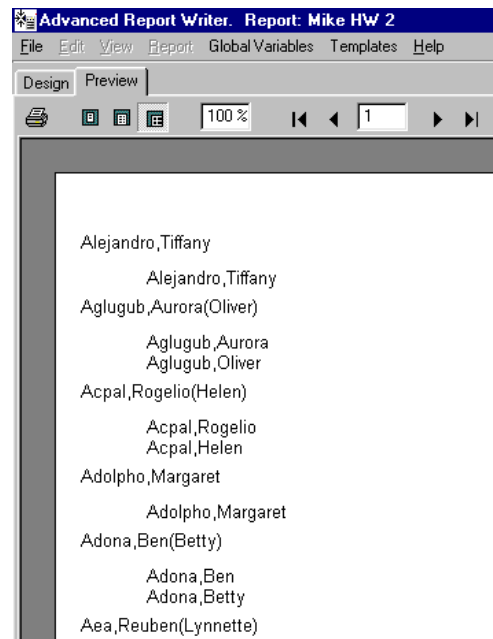


Fig. 3-4

The sub-report prints each member within the family.

## Right Click on Sub-reports

You can right-click on a sub-report to change its options or properties. An ellipsis [...] in the property name indicates additional controls.

**Bring to Front/Send to Back:** Moves a sub-report in front of or in back of other objects.

**Child:** The sub-report is a dependent part of the report, printing when called by the main report.

**Fixed:** Lock the position, size, and other elements of the sub-report, independent and regardless of objects around it. The opposite would be ShiftRelativeTo.

**Section:** The sub-report prints and acts separately from the main report or other sub-reports, including printer, paper, margins, etc. Used to create several separate reports in one.

**Drill Down...:** Connects a sub-report to an object. When the object is selected in Preview mode, the contents of the sub-report are displayed and printed.

**KeepTogether:** Keep all parts of a sub-report together on one page. Do not break contents across two pages.

**ParentWidth:** Make this sub-report the same width as the workspace.

**Position...:** Where is the sub-report positioned in the workspace, including width and height.

**ShiftRelativeTo...:** Keep the position of the sub-report relative to another sub-report or object.

**Stretch:** Automatically sizes the sub-report vertically to match its contents.

**Visible:** Make the sub-report (and its contents) visible or invisible.



## Groups

A group is a special kind of band, consisting of a Header and Footer that surrounds the Detail band. It can be used to keep the objects of the Detail together on a page, or to indicate when to start the next Detail on the next page.

For example, you have the Family Name, Address and E-mail in the Detail. As the report prints, a family's information may *break* (be split) between two pages. You want to keep the family's information together; if there's not enough room on the first page, print it all on the second page. Creating a group around the Detail will keep the family together.

### Add a Group

To create a group, click on **Report | Groups....**

The first step is to choose a field to be used as the **Break On** point. As the program loops through the Detail, each pass that includes this field is considered a group to be kept together.

The **Break On** field can be an internal **Data Field** or a **Custom Field**. The **Groups** drop-down list shows what is available.

If **Data Field** was chosen, the **Groups** list will include all available fields within the data pipeline, even if it is not used in the report.

If **Custom Field** is chosen, the **Groups** list will include only the objects that are in the report.

Choose the field and click on **Add**.

The field name appears in the Groups box as **Group[x]:fieldname** or **Group[x]:objectname**.

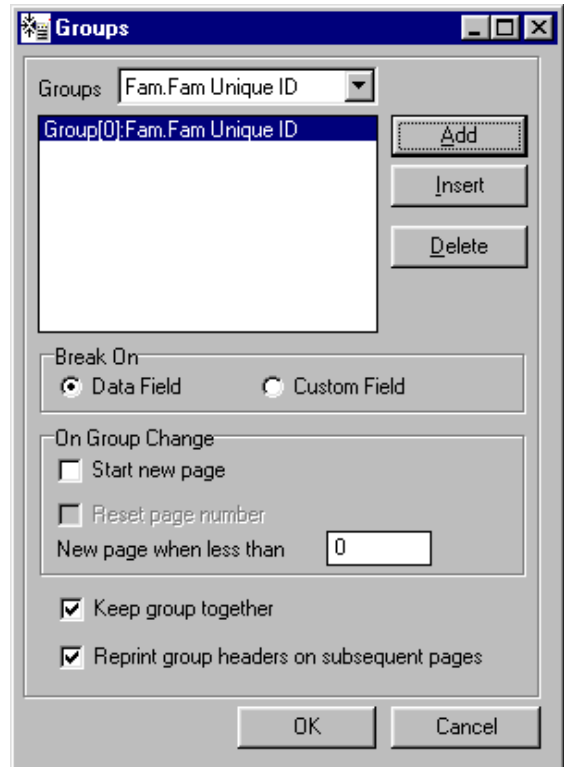


Fig. 3-5

The number in the brackets [ ] is the group's break hierarchy – the lower the number, the first in priority to keep together as a group. Therefore, Group[0] would be kept together first, then Group[1], etc.

Multiple group break points can be added, and Data Field groups can be mixed with Custom Field groups.

## Other Group Options

At the **Group Change** – when the program sees each new group - you can indicate how the new group is printed:

- **Start New Page:** Start each group on a new page.
- **Reset page number:** If each group is on a new page, should the page number be reset.
- **New page when less than:** Begin a new page when the space remaining at the bottom of the page (not including the bottom margin) is less than the value chosen. For example, if less than 1 inch remains, begin a new page.

**Keep Group Together:** Check this box to keep everything in the group together on the page.

**Reprint Group Headers on Subsequent pages:** Check this box to print the Group Header information at the beginning of each page.

## Group Header and Footer

The **Group Header** prints at the top of each group, just as the Header prints at the top of each page.

The **Group Footer** prints at the end of each group, just as the Footer prints at the end of each page.

For example, if the Group Header had a series of labels to identify columns, those labels would print above every family.

In this example, the Group Break On value is set as Fam Unique ID.

Fig. 3-6

Each unique family is treated as a group, and has the header printed above it.

Fig. 3-7

## Regions

A region is an object that is used to group other objects together. It is different than a Group in that it doesn't encompass the entire Detail band, and has no break-points. It is different than a sub-report in that it doesn't have its own tab or its own Header/Detail/Footer bands and cannot have a different data pipeline than the band it is in.

One of its main uses is to box a number of objects together so that they can be made visible or not visible as a group. It can also be set to Shift Relative to another object, sub-report or region.

### Add a Region



A region is added like any other object. To add a region, click on the **Region** button, then click in the workspace.

To get objects into a region:

Create the region first, then drop objects into it. Anything moved completely into a region is considered part of the region. Now as you move the Region, all of the objects inside move with it.

To remove an object from a region:

Click on the object and drag it outside of the region.

The **Report Tree** will also show the Region and the objects inside of it.

Regions automatically have an outline around them. **Right-click** and choose **Transparent** to turn off the outline, if desired.

### Right Click on a Region

You can right-click on a region to change its options or properties. An ellipsis [...] in the property name indicates additional controls.

**Bring to Front/Send to Back:** Moves the region in front of or behind its objects. If a region is not Sent to Back, and is moved over other objects, those objects cannot be seen, even though they are there, behind the region. When a region is first created, it is in Front.

**KeepTogether:** Keep all parts of a region together on one page. Do not break contents across two pages.

**ParentHeight:** Make this region the same height as the workspace.

**ParentWidth:** Make this region the same width as the workspace.

**Position...:** Where is the region positioned in the workspace, including width and height.

**ReprintOnOverflow:** If the region is Stretched, and would go to a 2<sup>nd</sup> page, reprint this region on that 2<sup>nd</sup> page.

**ShiftRelativeTo...:** Keep the position of the region relative to another sub-report or object.

**ShiftWithParent:** Keep the position of the region relative to the workspace.

**Stretch:** When printing, automatically expand the region to encompass its objects.

**StretchWithParent:** When printing, automatically expand the region to match the workspace.

**Transparent:** Turn off the outline that automatically surrounds a region.

**Visible:** Make the region visible or not visible, including its contents.

## Exercise

Borrowing from the exercise in chapter 2, create a Region next to the Family Phone List, and place these objects inside:

- A label and data field for **Date Registered**.
- A label and data field for **Geographic Area**.
- A label and data field for **Family Status**.
- Place a line object under each data field. This has the effect of creating a “fill in the blanks” area. If the data is already in the program, it will appear on the lines; otherwise, the lines are blank and ready to be completed.

The screenshot shows a report design window with a header section. The header contains the following fields:

- Owner Name: Van Loon, Jeff(Jeane), M/M
- Report Title: (blank)
- Date Registered: 10/05/1990
- Geog. Area: 21-33
- Family Status: Couple

The form also includes a line object under each data field, creating a “fill in the blanks” area.

Fig. 3-8

Click on **Preview**. The report should look similar to this:

The screenshot shows the Advanced Report Writer Preview window. The preview displays the report design for the first entry, which is for Van Loon, Jeff(Jeane), M/M. The report includes the following fields:

- Owner Name: Van Loon, Jeff(Jeane), M/M
- Report Title: (blank)
- Date Registered: 10/05/1990
- Geog. Area: 21-33
- Family Status: Couple

The preview also shows the report design for the second entry, which is for Harcourt, Elizabeth, Mrs. The report includes the following fields:

- Owner Name: Harcourt, Elizabeth, Mrs.
- Report Title: (blank)
- Date Registered: 06/01/1970
- Geog. Area: 4-20
- Family Status: Individual

The preview also shows the report design for the third entry, which is for Schmid, Jeffrey(Lenore), M/M. The report includes the following fields:

- Owner Name: Schmid, Jeffrey(Lenore), M/M
- Report Title: (blank)
- Date Registered: 11/01/1987
- Geog. Area: 20-49
- Family Status: Extended Family

The preview also shows the report design for the fourth entry, which is for Ramirez, Tomas(Louise), M/M. The report includes the following fields:

- Owner Name: Ramirez, Tomas(Louise), M/M
- Report Title: (blank)
- Date Registered: 03/01/1993
- Geog. Area: 26-25
- Family Status: Two Parent Family

The preview also shows the report design for the fifth entry, which is for Garinger, Ronald(Lucero-Garinger, Jenny, Ms.), Mr. The report includes the following fields:

- Owner Name: Garinger, Ronald(Lucero-Garinger, Jenny, Ms.), Mr.
- Report Title: (blank)
- Date Registered: 03/01/1996
- Geog. Area: 11-45
- Family Status: (blank)

Fig. 3-9

# Project: Building the Family Directory

Beginning with this chapter, we will build the Family Directory report from scratch. Each subsequent chapter will add a new concept or level of complexity to the report. At the end, we will have a hand-made, working copy of the Family Directory.

In this first installment, we will use the concepts covered in chapters 2 and 3 – namely, objects, sub-reports, regions, and groups – to build the basics.

## Start the Report

The first step is to add a new ARW report. From the Main Menu:

1. Click on **Families | Family Reports | Add | Advanced Reports | Report**.
2. Click on **Next**. Name the report “Family Directory Tutorial” and provide the description “Family Directory build from scratch”. Click **Next**. Click on **Modify the Report**.

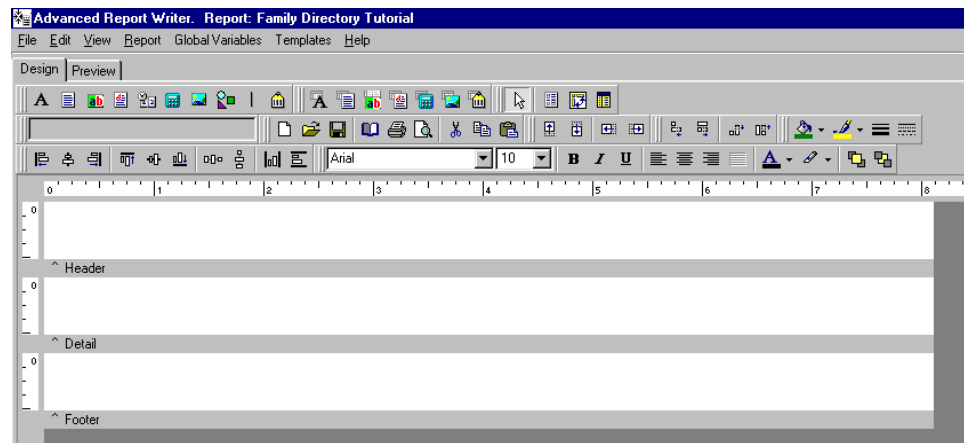
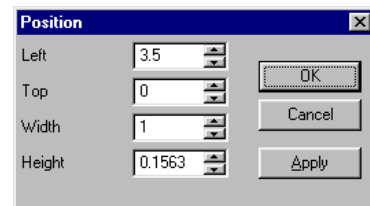
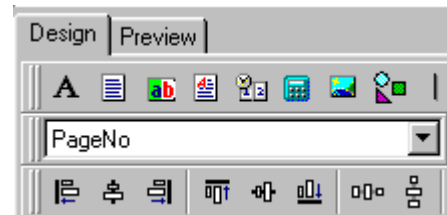
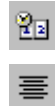


Fig. 3-10

## The Header

The Header of the report is fairly straightforward.

1. Click on the **System Variable** button, and drop it in the Header.
2. Click on the **Center** button to center the value inside of the object.
3. Choose the System Variable as **PageNo** in the object selection box.
4. Right-click on the object and choose **Position...**. Set the Width at one inch, the Top at zero, and the Left at 3.5 inch. Click OK.



Right-click on the gray Header band, choose Position, and set the Height to 0.1875.

You should now have a page number system variable, centered in the top middle of the Header.

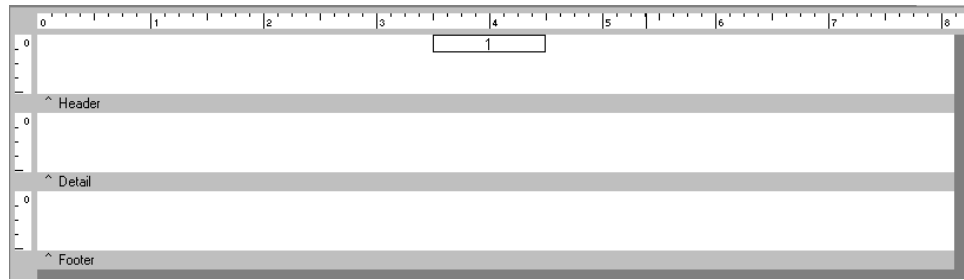


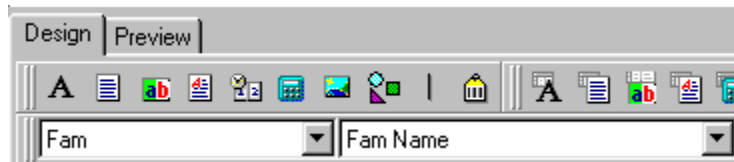
Fig. 3-11

## The Detail

The Detail band is where most of the work gets done.

Here we will be adding the Family Name, Address Block, Phone List, and a Member subreport. We will be leaving space for a column heading letter, taken from the first letter of the last name, which will be added in chapter 4.

1. Click on the **DBText** button, and drop a label in the Detail band.
2. Right-click and choose **Position**. Left: 0.125, Top: 0.25, Width: 3.00.
3. Choose the Data Pipeline as **Fam Fam Name**.



4. Click on the **DBMemo** button, and drop a memo in the Detail band, underneath the Fam Name.
5. Right-click, choose **Position**, and set as Left: 0.25, Top: 0.50, Width: 3.00, Height: 0.20.
6. Right-click again and choose **Stretch**.
7. Choose the Data Pipeline as **Fam Fam Address Block**.
8. Click on the **DBMemo** button, and drop a memo in the Detail band, underneath the Fam Address Block.
9. Right-click, choose **Position**, and set as Left: 0.25, Top: 0.6875, Width: 3.00, Height: 0.20.
10. Right-click again and choose **Stretch**.
11. Right-click again and choose **ShiftRelativeTo...**. In the dialog box, choose **Fam Address Block (DBMemo1)**.  
ShiftRelativeTo forces the object to adjust its position based on another object. In this case, we want to keep the Phone List below the Address Block, regardless of how large the Address Block is. Otherwise, they would print on top of one another.
12. Choose the Data Pipeline as **Fam Fam Phone List**.



So far, the report should look like this:

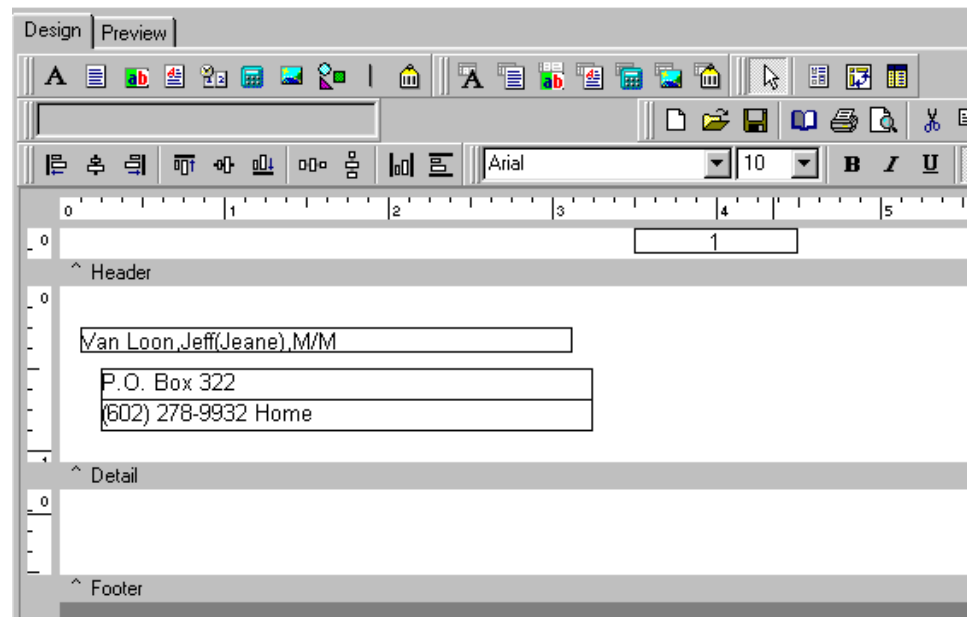


Fig. 3-12

Click **Preview** to get this view.

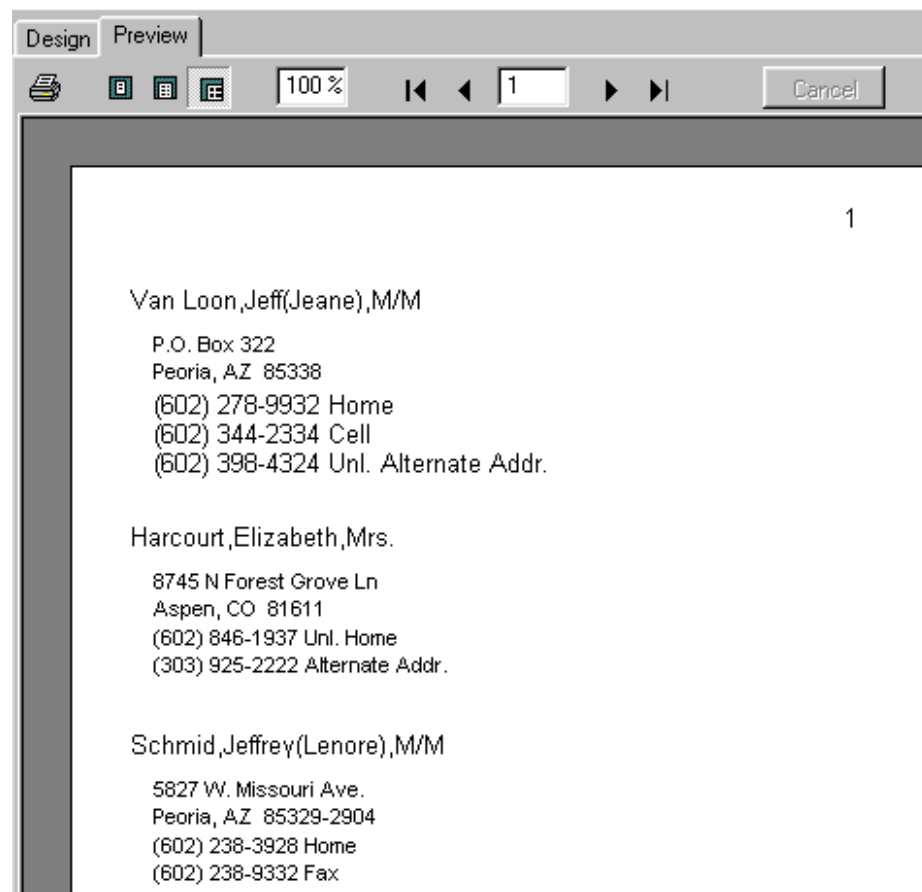
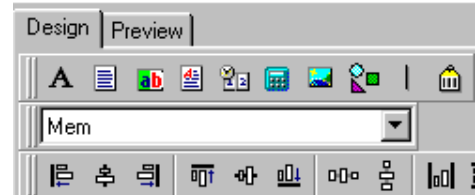


Fig. 3-13

## Add the Sub-Report

Next is the Member sub-report, added to the Detail band.

1. Click on the **Sub-report** button, and drop a sub-report beneath the Fam Phone List.
2. Right-click on the Sub-report and choose **ParentWidth**. This turns off the automatic horizontal stretching.
3. Right-click again and choose **Position**. Set Left: 0.50, Top: 0.875, Width: 3.0.
4. Right-click again and choose **ShiftRelativeTo**. In the dialog box, choose **Fam Phone List (DBMemo2)**.
5. Select the **Data Pipeline** as Mem.



The Detail band should now look like this:

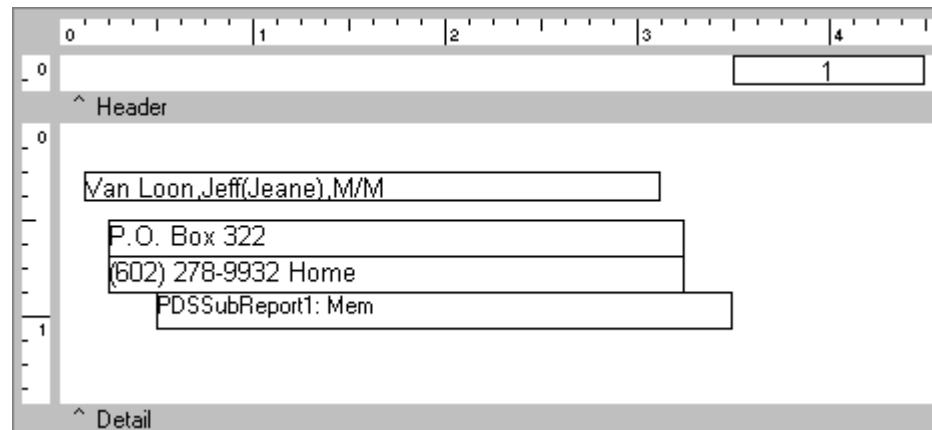


Fig. 3-14

## Setup the Sub-Report

Now that we have a sub-report, we can add the appropriate objects to it. Click on the sub-report tab at the bottom of the workspace.



The Mem sub-report starts with Title, Detail and Summary bands.

To remove the Title and Summary bands, click on **Report | Title**, then **Report | Summary**.

Right-click on the Detail band and choose **DynamicHeight**. This will stretch or contract the Detail band vertically to match the objects inside.



We will add the Member Name, Age, and Grade to the Detail Band.

1. Drop a **DBText object** into the Detail band.
2. Right-click and set the **Position** as Left: 0.00, Top: 0.00, Width: 2.0.
3. Set the **Data Pipeline** as Mem Mem Name.
4. Drop another **DBText object** next to the Mem Name.
5. Right-click and set the **Position** as Left: 2.125, Top: 0.00, Width: 0.375.
6. Set the **Data Pipeline** as Mem Mem Age.
7. Drop another **DBText object** next to the Mem Age.
8. Right-click and set the **Position** as Left: 2.625, Top: 0.00, Width: 0.875.
9. Set the **Data Pipeline** as Mem Mem Grade.

The sub-report Detail band will look like this:

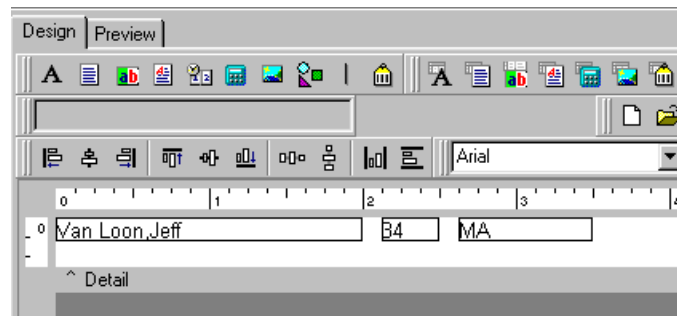


Fig. 3-15

Click on Preview to see our progress.

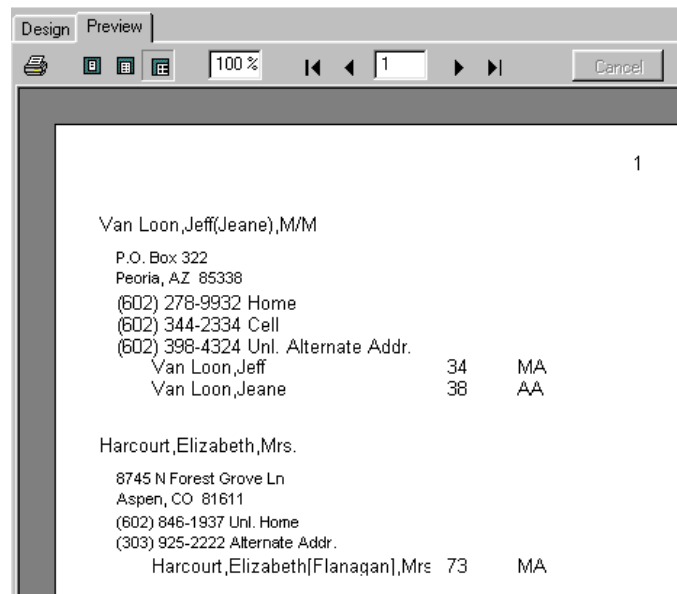


Fig. 3-16

## Adding the Group

When printing this directory, we don't want to split a family across a page. In other words, if a family – with their address, phones, and members – is too long to fit at the bottom of the page, print them on the next page instead. This is a group.

Click on **Design** to return to the workspace area, then click on the **Main: Fam** tab at the bottom of the screen to return to the main report.

To add a group:

1. Click on **Report | Groups...**
2. In the Groups field, choose **Fam Fam Unique ID**. Click on **Add**.

This creates the group as **Group[0]:Fam.Fam Unique ID**. We chose this field to make sure each family is identified in a unique way. If we were to choose Fam Name, and there were two families with the same name, the Group could not tell them apart.

3. We will leave the remaining settings on this dialog box set to their defaults. The most important one for this example is the checkbox **Keep Group Together**. If this is checked, each family will be kept together as a group, and not split between pages.
4. Click **OK**.

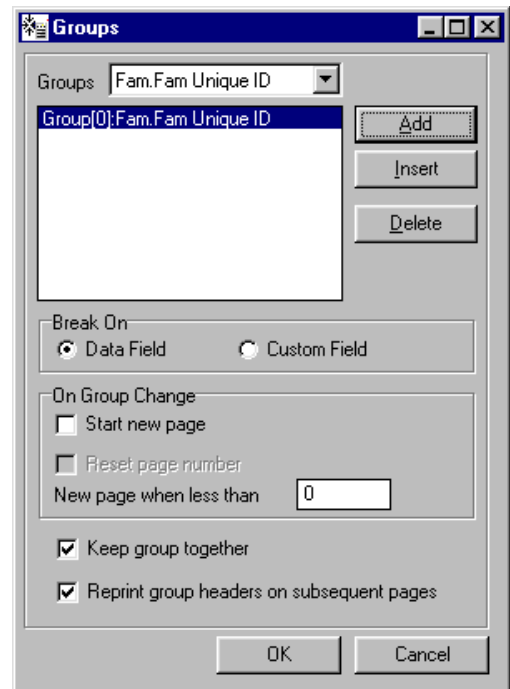


Fig. 3-17

The Detail band now has a Group Header and Group Footer band on either side, indicating that it is now controlled by a group, and showing the field we chose as the group break point – Fam Unique ID.

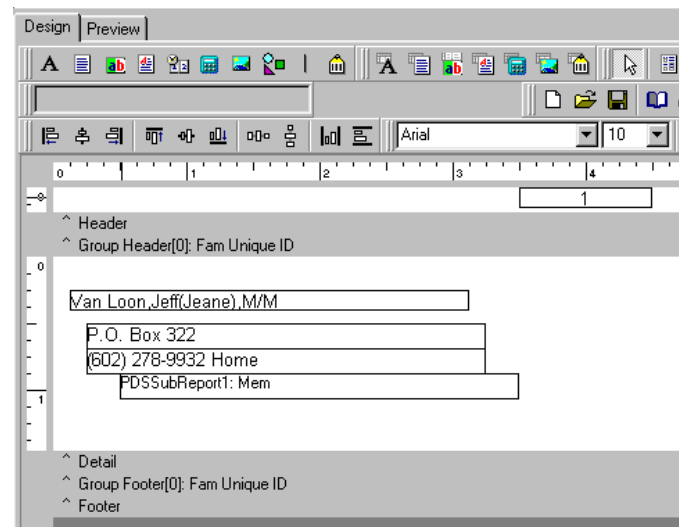


Fig. 3-18

Click on **File | Save** to save the report, and run it normally to see the finished product. In chapter 4, we will add a column label and another group.

---

## 4: Embedded Code

### Overview

Built into the Advanced Report Writer is the ability to include pieces of 'code' embedded in the report to be 'run' when a particular event occurs. The code is a list of instructions in a form that the computer can understand. The computer will 'run' this by loading each instruction and doing what that instruction tells it to do.

The code is only part of the story. In addition to the code there is also the information or 'data'. The code uses and manipulates this data. Code and data together make it possible to do complicated things in the reports.

### Code Editor

The code editor is where the code for an event or object is entered. The code editor can be accessed in a variety of places. The easiest way is to right-click on an object, and choose one of its events, such as **Get Text** or **Do Before Print**.

For example, this is the code editor for **Label1**, for the event **Do Before Print**.

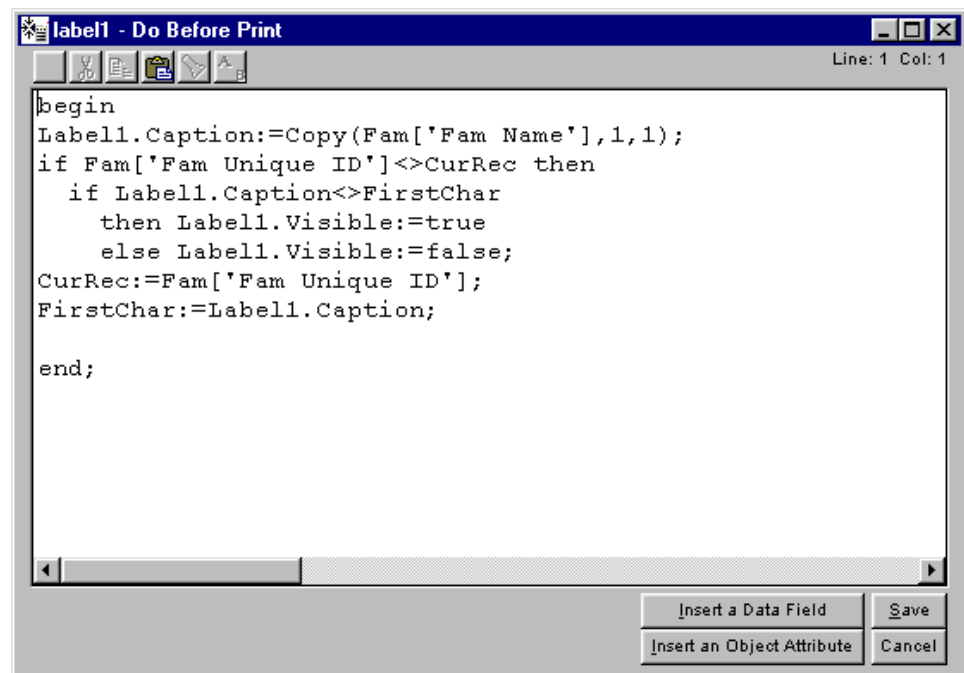


Fig. 4-1

If the object is a variable, you can right-click and choose **Calculations**. This will open the code editor for the calculations of that variable.

You can click on **Embedded Code | Define Global Variables** or **Initialize Global Variables** to open the code editor for the global variables.

Once your events, objects and other codes are in place, you can click on **Embedded Code** | **View All Embedded Code** to choose which event to view. When opened this way, the code editor will indicate which event the code belongs to in the Procedure name.

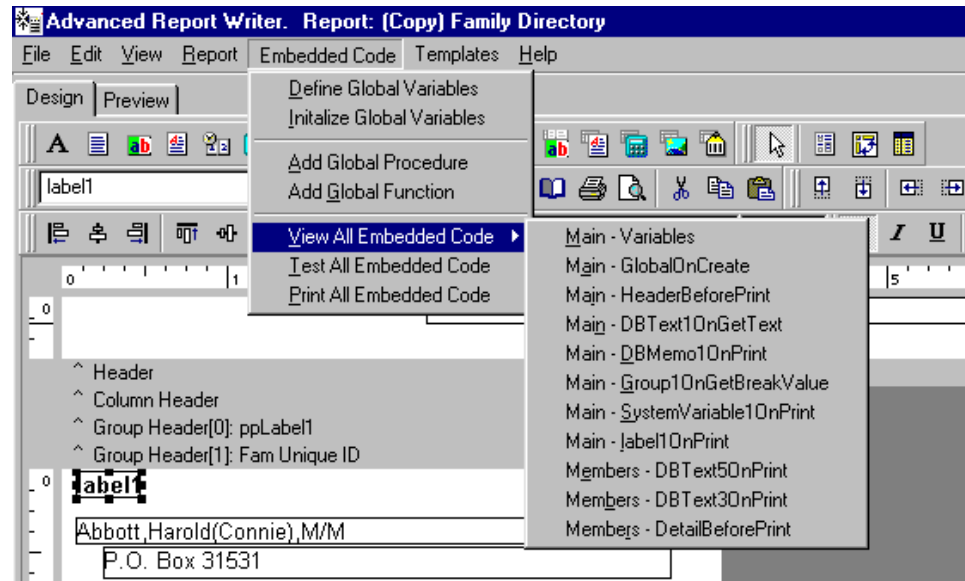


Fig. 4-2

Below is the same **Label1**, **Do Before Print** code, but viewed from the **View Embedded Code** menu. Note the procedure name is added as **Label1OnPrint**.

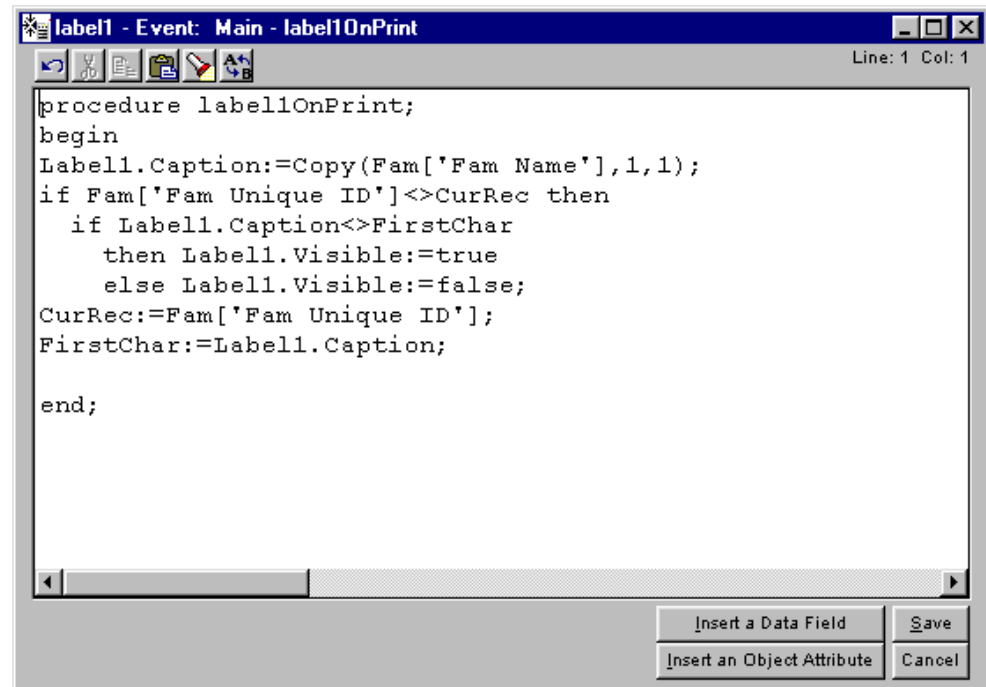


Fig. 4-3

## Data

In reports, data has two general aspects: the kind of data and where it is stored. The first is what kind of data. A piece of data may be an integer or it may be a string of letters for a name or it may be a financial amount. There are many different kinds of data. The most useful types are:

Integer:	a whole number (with no decimals) between -2147483648 and 2147483647
String:	a series of characters, which may include numerical digits
Currency:	a decimal financial amount between -922337203685477.5808 and 922337203685477.5807
Byte:	a single 8 bit byte 0 to 255
Single:	$1.5 \times 10^{45}$ to $3.4 \times 10^{38}$ (equal to 4 bytes)
Double:	$5.0 \times 10^{324}$ to $1.7 \times 10^{308}$ (equal to 8 bytes)
Boolean:	a true or false value
Tdate:	Same as a Double, representing the number of days since 12/30/1899
TdateTime:	Same as Tdate, but with time represented as a fraction of a day

Each kind of data has different things that can be done to it. For example, a piece of data that is an integer can have 2 subtracted from it. This would make no sense for a string of letters. Some of the built-in operations are detailed in the Calculations section.

There are three different places data is stored in reports – data fields, objects and variables.

### Data Fields

Data fields are pieces of information that are usually stored on the disk. Data fields have values assigned to them, for example a family's name or a member's age. Inside of the code, a data field like member age looks like this:

```
Mem[ 'Mem Age' ]
```

This data field is from the Mem table and the field name is 'Mem Age'. To make it easier when entering code, there is a button in the code editor titled **Insert a Data Field** (See figure 4-1). This will bring up a list of all the available data fields broken down by table.

### Objects

Objects are the things put into the report in the report designer. For example, when you add a label to a report by clicking on the label icon and clicking in a band, that label is an object. An object is a kind of holder of information. For example, a label object holds the text of what is printed for that label. It actually holds a number of pieces of information commonly referred to as 'Attributes'. In the case of a label, the object also stores the color of that label, its position and a number of different pieces of information. The most important piece of information is the object's name.

Each object in a report can be accessed by the code using the object's name. Every object placed into a report has a name in the form of the type of object (i.e. Label, DBText, Line) and a unique number. So, if we start with a new report and we place a label on it, the name of that object will be **Label1**. If we place another label, it will be **Label2**. If we place a line under one of the, this object will have the name **Line1**, and so on. The name of the object is used in referencing that object in the code.

You can change the name of any object by opening the Report Tree, right-clicking on the object, and choosing **Rename**. An object name *cannot* have any of the following characters: spaces, tabs, single quotes, double quotes, periods, commas, semicolons, colons or any of these symbols: -, +, /, \*, (, ), [, ], {, }, =, #, ~, |, \, <, >, ?, !, @, %, ^, &.

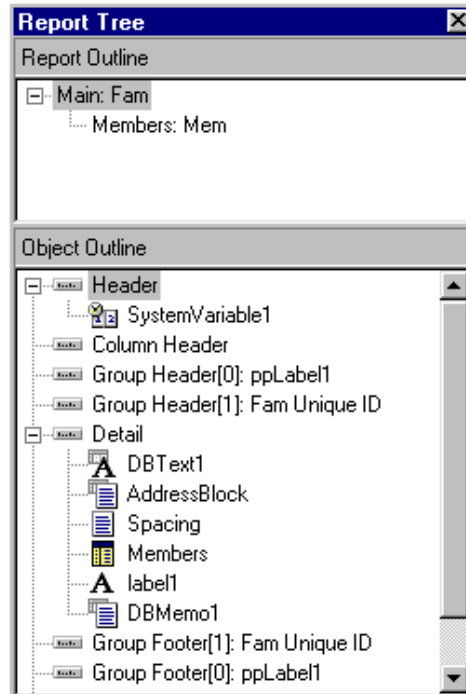


Fig. 4-4

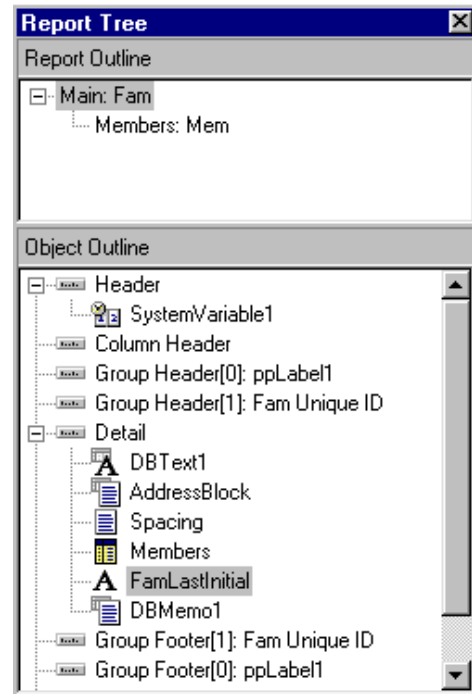


Fig. 4-5

In this example, the object **Label1** has been renamed to **FamLastInitial**, to better indicate what the label represents.

When an object is renamed, existing code using the old name must be changed to use the new name, or an error will result and the report will not run.

### Attributes

Associated with every object is a list of attributes. As mentioned earlier, attributes can be things like the color of the object or the text of the caption or whether or not the object is visible, etc. Each type of object can have its own unique set of attributes. For example, a label has a caption but a line does not.

An attribute can be set in the report designer or by embedded code. In the example in the overview, we set the color of the font for **Label1** to red just before it was printed. We could have set it in the report designer by selecting the **Label1** object and then clicking on the icon that sets the color of text and choosing “red”.

Some attributes are set by selecting the object and then using the right mouse button to open a list of attributes. Click on the attribute to set it, for example **Visible** or **Autosize**.

If we can set the attributes of an object in the report designer, why would we ever set them using embedded code? Well, if we are just setting an attribute, then there is no reason to use embedded code to do it. But, if we are setting the attribute based on some condition or calculation, then we would use the embedded code to determine what the value of the attribute should be and then set it. In our earlier example we might want to set the label's text to "red" if the value of the member age field is less than five (5). The code in this case is shown in the IF-THEN statement later in the code section.

Each object has many attributes. To make it easier when entering code, there is a button in the code editor titled **Insert an Object Attribute**. This will bring up a list of all the available objects and their attributes.

To set an object's attribute in embedded code, identify the attribute in this format:

```
Object.attribute:=value;
```

For example:

```
Label1.Color:=clRed  
Label1.Caption:=' Family Name' ;  
Label1.Visible:=true;
```

## Events

In addition to attributes, every object has a list of events that we can attach embedded code to. Each type of object can have its own unique events. For example, a label can have an event called **Get Text**. Code attached to this event is run when the computer needs to know what the text of that label is. We can use this event to customize the text of a label.

## Variables

The last form of data is variables. Unfortunately, there is also a type of object called a "variable". In this case, we are not referring to that object, but to temporary boxes for holding data that are defined and set inside the code.

There are two types of variables: global and local. Global and local variables differ only in how long they exist. Global variables exist as long as the report is running, and are created by clicking on **Embedded Code | Define Global Variable** on the tool bar. Click on **Embedded Code | Initialize Global Variable** to set the initial value of the variable.

Local variables are created in the event code itself. They only exist as long as the event where they are defined is being used. There is a 'Local Variable declaration' section later which gives more detail on this.

## Code

The computer language used for the code in events is **Pascal**. It has been around since the 1970's. In Pascal, code is divided up into routines.

Each routine can be a single statement or can be made up of a group of other statements. Each statement must end with a semicolon. It also helps to indent some parts of the code to make the routines more readable. The capitalization of the statements and objects in the code is not important.

Most, if not all, of the code embedded in a report is associated with an event on an object. Right-click on an object to see the attributes and the events, which are at the bottom. Clicking on one of these events takes you to the code editor for that event. The code editor will always put in a `BEGIN...END` statement. It may put in additional code to help you with that object. For example, in the **Get Text** event of a label, the code editor automatically puts in a `TEXT :=` statement, since you are most likely going to be setting the text of the label.

There are a number of different types of statements, listed below. Some of the examples in the following section may use types of statements that are defined later in the list.

### BEGIN-END statements

The `BEGIN...END` statement is used to group other statements. There are many times when you want to treat a number of statements as a block of statements. You can use a `BEGIN` and `END` to create a block of statements. You usually indent the statements that are inside the block to make it more readable. Examples of this can be seen in the `IF...THEN` statement.

### Local Variable declarations

Local variables are created by adding a declaration statement before the very first `BEGIN` in an event. So, if we select an event, the code editor puts in the initial code for us:

```
begin  
  
end;
```

To add three local variables: `MyName` which is a string, `ItemCount` which we want to be an integer and `ItemMax` which is also an integer; we would add a declaration before the `BEGIN`.

The variable declaration begins with `VAR` and then lists the variable names and variable types. The variable names and variable types are separated by a colon, and the individual variables are separated with a semicolon. If there is more than one variable of the same type, you can enter the variable names separated by commas, then the colon, and then the type. So, our example would be:

```
var  
    MyName: string;  
    ItemCount, ItemMax: integer;  
begin  
end;
```



## Assignment Statements

This is used to 'set' a value into a variable or an attribute. It can be a simple value or a complicated calculation. The form of this statement is a variable followed by a colon and an equal sign and then the value or calculation. Some examples would be:

```
a:=5;
Total:=Total+100.00;
Label1.Caption:='The Family is inactive';
```

Notice that each statement is ended with a semicolon. There is a section later on that is devoted to calculations.

## IF-THEN statements

The IF statement is one of the basic building blocks of any code. It allows you to test if some condition exists and to run a set of instructions if it does.

The basic form of this statement is the word 'If' followed by some type of comparison or calculation that can be evaluated to be either true or false (this is covered in more detail later in the Conditions section). This is followed by the word 'Then'. After the 'Then' is a statement (or a group of statements in a BEGIN...END block). We will refer to this as the 'THEN-statements'. These statements are run if the condition is true. A few examples would be:

```
if (i>100)
then i:=100;
```

In this example, if the value in the variable 'i' is greater than 100, the same variable 'i' is set to be 100.

Also note that the IF statement is not followed by a semicolon.

```
if (Mem[ 'Mem age' ]>=21) then
begin
    Label1.Caption:='Can Supervise';
    Label1.Visible:=true;
end;
```

In this example, if the member's age is greater than or equal to 21, then we make the object **Label1** visible by making its 'Visible' attribute equal to 'true', and we set its 'Caption' attribute to 'Can Supervise'.

You can also put the word ELSE after the THEN-statement to put in a statement (or a group of statements in a BEGIN...END block). These statements will be run when the IF condition is not true. If there is an ELSE statement, then there is no semicolon after the THEN-statement. Some examples are:

```
I:=Mem['Mem age'];
if (I>=21) and (I<30) then
begin
  Label1.Caption:='Can Supervise';
  Label1.Visible:=true;
end
else
begin
  Label1.Caption:='Can Supervise';
  Label1.Visible:=true;
  if I<5
  then Label1.Font.Color:=clRed
  else Label1.Font.Color:=clBlack;
end;
```

As you can see in the example above, IF statements can be nested inside one another. In most cases it is best to put the nested statements inside of a BEGIN...END block to eliminate confusion as to which statements go with which IF condition. An example would be:

```
if (I>5) then
  if I<16
  then Label1.Caption:='the value is 5 to 15'
  else if I>20
  then Label1.Caption:='the value is above 20'
  else Label1.Caption:='the value is 16 to 20'
else Label1.Caption:='the value is less than 5';
```

This is complicated. The indenting helps a lot but it is hard to tell what the final ELSE is for. This can be made easier by adding a couple of BEGIN...END blocks. Compare the above example with the following one.

```
if (I>5) then
begin
  if I<16
  then Label1.Caption:='the value is 5 to 15'
  else
  begin
    if I>20
    then Label1.Caption:='the value is above 20'
    else Label1.Caption:='the value is 16 to 20';
  end;
end
else
begin
  Label1.Caption:='the value is less than 5';
end;
```

## CASE statements

A CASE statement is a short cut for doing many IF statements. Its purpose is to make some code structures easier to read. It replaces a series of IF...THEN statements with an easier form. For example, code such as this:

```
if I=1
  then s:='January'
else if I=2
  then s:='February'
else if I=3
  then s:='March'
else if I=4
  then s:='April'
else if I=5
  then s:='May'
else s:='Other';
```

Would become this CASE statement:

```
case I of
  1: s:='January';
  2: s:='February';
  3: s:='March';
  4: s:='April';
  5: s:='May';
  else s:='Other';
end;
```

## WHILE loops

Often you need to do something over and over while some condition exists. This is the purpose of the WHILE loop. A WHILE loop continually runs a group of statements as long as the given condition is still true. (A statement is referred to as a 'loop' if it executes itself over and over again) For example:

```
I:=10-length(s);
While I>=1 do
  Begin
    s:='0'+s;
    I:=I-1;
  End;
```

This piece of code takes a variable string named 's' and pads it with leading zeroes, making the value of 's' at least 10 characters long.

## REPEAT-UNTIL loops

The REPEAT...UNTIL loop is very similar to the WHILE loop, except it tests the condition *after* running the statement and repeats the statements until the condition becomes true. For example:

```
Repeat
  s:='0'+s;
  i:=i-1;
until i=0;
```

## FOR loops

Many times we need to run a group of statements a specific number of times. We could use a WHILE loop to do this, but there is a statement specifically designed for this function. It's called a FOR loop. For example:

```
s:='';
for i:=1 to 9 do
begin
  s:=s+chr(48+i);
end;
```

This would be the same as:

```
s:='123456789'
```

The variable used in the FOR statement must be declared and must be an integer. It will assign to it the current value of the loop. Instead of the 'to' we can use 'down to' if we want to go backwards. For example:

```
s:='';
for i:=9 down to 1 do
begin
  s:=s+chr(48+i);
end;
```

This would be the same as:

```
s:='987654321'
```

You can also use a 'step' if you want to jump in increments other than 1. For example:

```
s:='';
for i:=1 to 9 step 2 do
begin
  s:=s+chr(48+i);
end;
```

Any of the values after the ':' in the FOR statement can be a variable or a calculation. For example:

```
s:='';
for i:=1 to ListCount step delta do
begin
  s:=s+chr(48+i);
end;
```

## Built-in Procedure statements

There are two built-in procedures: `MessageBeep` and `ShowMessage(str)`. The `MessageBeep` procedure will cause the computer to generate a beep sound. The `ShowMessage` procedure will display a dialog box with the message that is in the parenthesis. The message can be a literal string, a string variable or a string calculation.

There are other built-in routines but they deal with scripting and are detailed in that chapter.

## Conditions

The conditions used in `IF` statements, `WHILE` statements and `REPEAT` statements can be simple comparisons or can be complex calculations. The common comparison operations that are used in if statements are:

<code>&gt;</code>	Greater than
<code>&gt;=</code>	Greater than or equal to
<code>=</code>	Is equal to
<code>&lt;</code>	Less than
<code>&lt;=</code>	Less than or equal to
<code>&lt;&gt;</code>	Not equal to

You can also use some Boolean operations:

- And – both or all conditions must be true
- Or – either or any conditions must be true
- Xor – only one of the conditions can be true
- Not – none of the conditions can be true

When using these Boolean operations with the comparison operations, you must use parentheses. For example:

`I>2 and Y<3`

must be entered as

`(I>2) and (Y<3)`

or you will get an error.

## Calculations

Calculations can be done inside of assignment statements and inside of conditions. A calculation can be made up of a single item referred to as a literal. Examples of these would be 5, 'Test', true or 10.20.

More commonly, a calculation is an equation with reference to object attributes or to a host of built-in routines. In the following examples the variables starting with 's' are assumed to be strings, 'i' are assumed to be integers, 'd' are assumed to be dates and 'a' are assumed to be currencies:

```
s1+' and '+s2;  
copy(s,1,1)+' - '+trim(copy(s,2,255));  
(i*100-i1) div 3;  
c*15.0/100.0;
```

## Comments

A particular section of code may seem obvious to you now, but later it may seem confusing. Comments can be entered into the code to make it more readable and to help explain what the code does. This is especially helpful if others will be viewing the code.

To enter a comment anywhere in your code, simply enclose it in 'curly' brackets: { }. For example:

```
{This is a comment. In this code, only those over 21
can supervise.}
if (Mem['Mem age']>=21)
  then Label1.Caption:='Can Supervise';
  else Label1.Visible:=true;
```

There are special menu commands that allow you to test all of the embedded code, view the code for any event, and print a report of the code.

## Test the Code

To test the code, click on **Embedded Code | Test All Embedded Code**. The program will test the code for syntax (formatting) errors, and for items such as uninitialized variables.

Some common errors are:

- Leaving off a semicolon at the end of a statement.
- Putting a semicolon on the line before an ELSE statement.
- Setting the attribute in one case but not resetting it in the alternate case. For example, we might want the color of a label to be red when a particular value is more than \$100. We must remember to set the color to red but also to set the color to black in the other case. If we do not then once we get to a record that is red every record after that will be red even though their values are less the \$100.
- Spelling the field name incorrectly.
- Not setting up or initializing global variables.

Using Boolean fields inside of an IF, WHILE or REPEAT statement sometimes does not work. It works better if the value is set into a string first. For example:

```
Var
  V: string;
Begin
  V:=Inv['Inv Is Reconciled'];
  If v='Yes'
    Then Label1.Caption:='reconciled'
    Else Label1.Caption:='';
End;
```

## View the Code

To view any embedded code that has been created, click on **Embedded Code | View All Embedded Code**. Then choose which event you wish to view. The code editor will show the procedure name and the code.

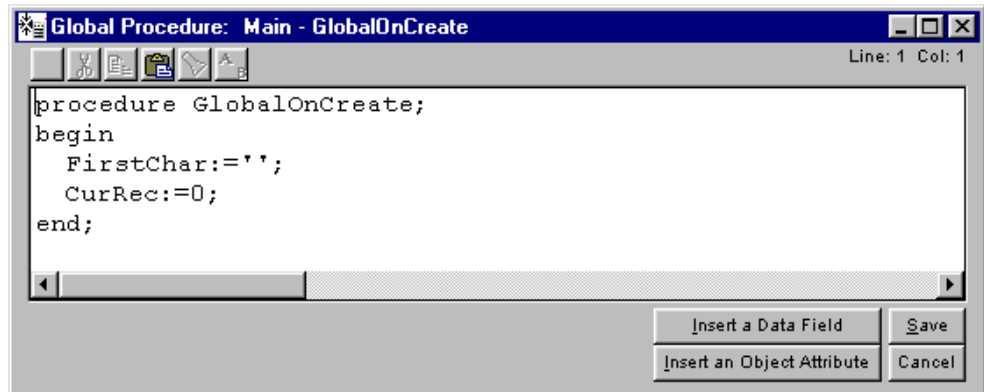


Fig. 4-6

## Print the Code

When working with embedded code, sometimes it is easiest to see all of the code at once – so that errors can be found and the flow of variables can be tracked more effectively.

To print (or view) all the embedded code in one report, including global variables, events, etc., click on **Embedded Code | Print All Embedded Code**.

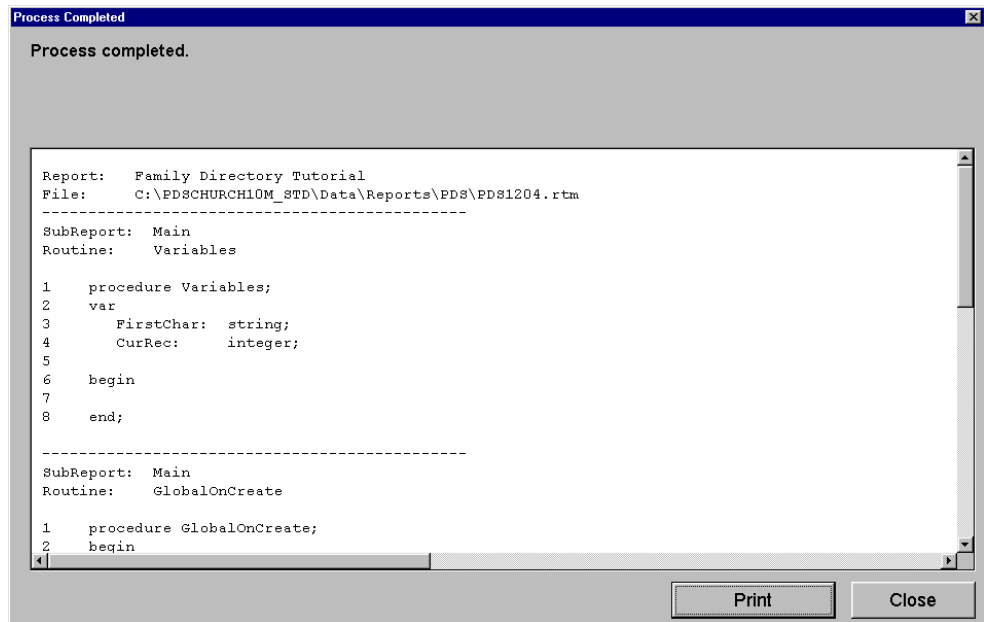


Fig. 4-7

## Ongoing Project: Building the Family Directory

In Chapter 3, we began building the Family Directory report from scratch. At the end of that chapter, we had added the basic labels and data fields, and had included a sub-report and a group.

In this installment, we will add some embedded code.

### Open the Report

Open the Family Directory Tutorial report.

From the Main Menu:

Click on **Reports** |  
**Family Reports** |  
**Family Easy Reports** |  
**Family Directory Tutorial** |  
**Next** | **Next** |  
**Modify the Report.**

Currently, the report looks like figure 4-8.

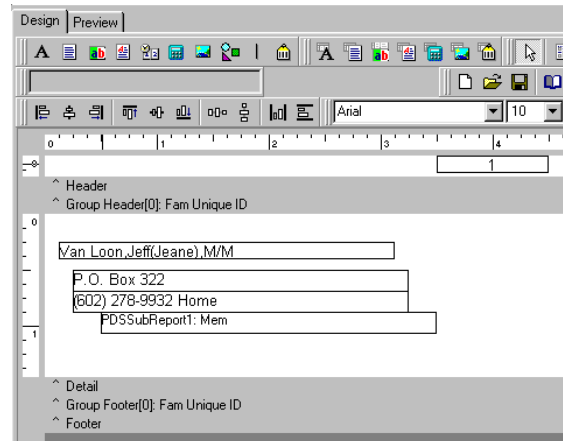


Fig. 4-8

We want to add a label to the Detail band, above the Family Name, that consists of the first letter of the family's last name. We only want the letter to appear at the top of the page or when the letter changes.

In order to do this, we need to create and initialize some global variables, add some embedded code to the Family Name's **Get Text** event, and add some embedded code to the new label's **Do Before Print** event.

### Create Global Variables

To create the global variables necessary, click on **Embedded Code** | **Define Global Variables**. The program opens the code editor and provides the word 'var', short for 'variable'. Type in these variable names and types:

```
FirstChar:  string;  
CurRec:   integer;
```

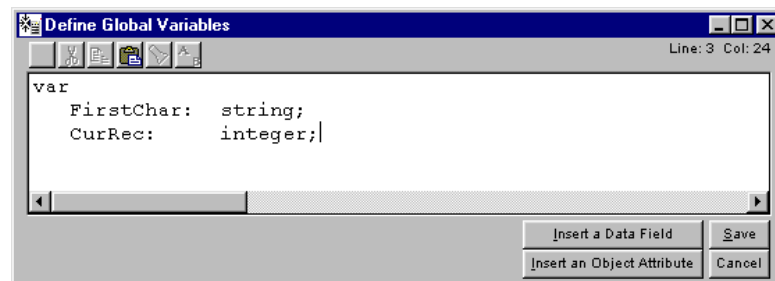


Fig. 4-9

`FirstChar` will represent the first character of each family name as it is encountered by the report. `CurRec` will represent the current family record that the report is looking at.

Click on **Save**.



## Initialize Global Variables

After creating the global variables, we need to initialize them, or set them to their beginning values.

Click on **Embedded Code | Initialize Global Variables**.

Between the `BEGIN` and `END` statements:

- Set `FirstChar` equal to a null character. This is represented as two single quotes with no space between them. This is NOT a double quote.

```
FirstChar:='' ;
```

- Set `CurRec` equal to zero.

```
CurRec:=0;
```

Note that we use 'colon-equals' [ `:=` ] in these assignment statements, instead of regular 'equals'.

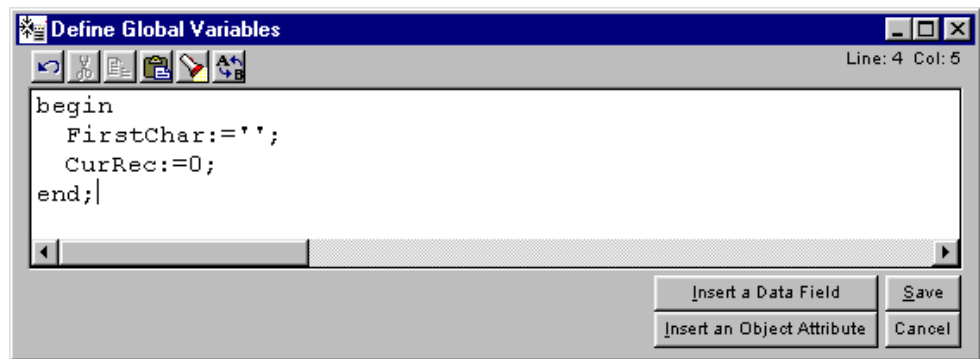


Fig. 4-10

Click on **Save**.

## Inactive Families

Next, we want to add some embedded code to the **Family Name** field, so that the word 'Inactive' prints next to families who have been marked as inactive. We need to print this at the same time that the family name prints, so we will embed the code in the **Get Text** event.

Select the Family Name, which is **DBText1**. Right-click and select **Get Text**. This will open the code editor.

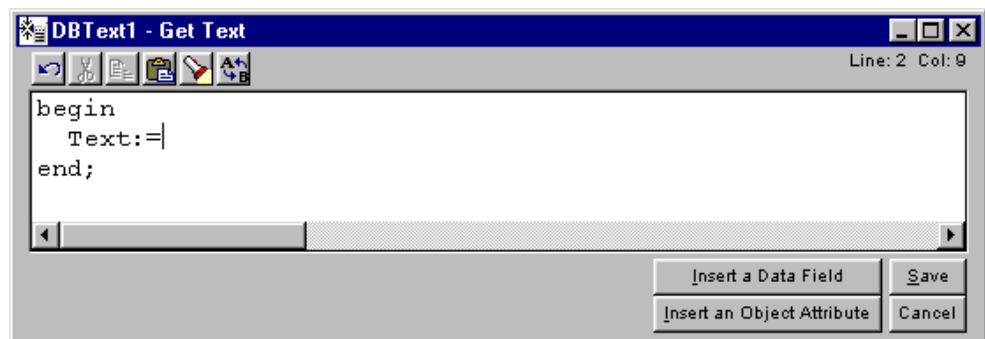


Fig. 4-11

Above the `BEGIN` statement, create a local variable 's', which will be a string type and will represent whether the family is inactive.

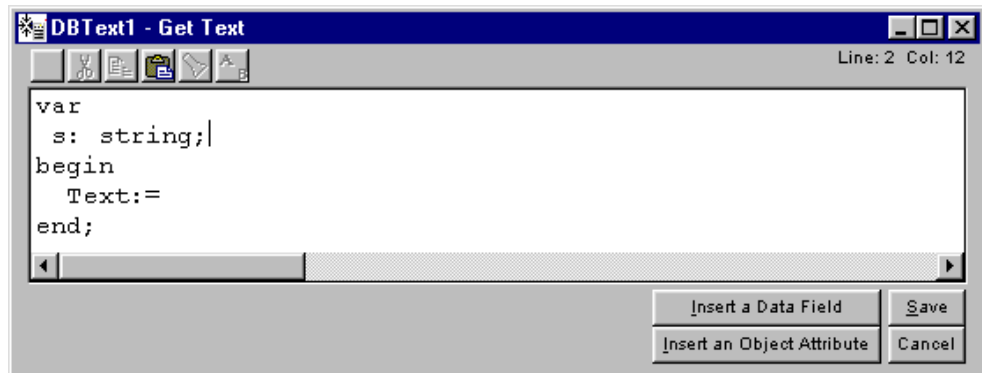


Fig. 4-12

Inside the `BEGIN-END` block, set 's' equal to the field **Fam Fam Inactive**. You can type this or use the **Insert a Data Field** button to choose Fam Inactive from the Fam table.

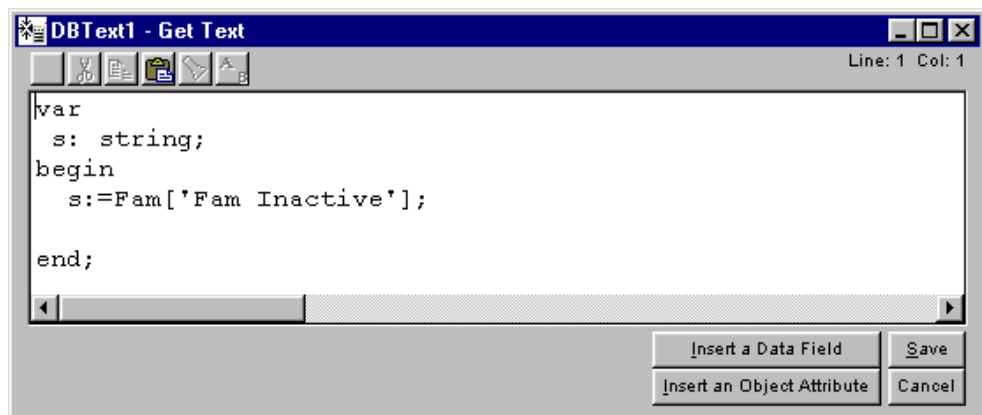


Fig. 4-13

Add an `IF-THEN-ELSE` statement that says, "if 's' is equal to 'yes', then the text of the field is equal to itself plus the word 'inactive', else the text is equal to itself."

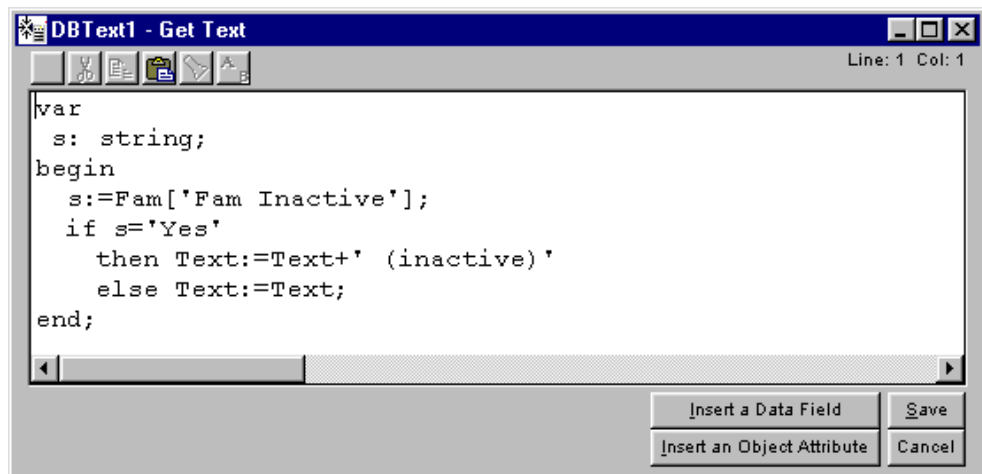


Fig. 4-14

Notice a few things about this block of code:

- The equals used to compare 's' to the word 'Yes' does not have a colon in front. The colon is not used in comparison statements like IF, only in assignment statements.
- The IF and the THEN statements have no semicolon following them. The semicolon is only placed after the last statement of a block.
- The ELSE statement assigns Text equal to itself. This is the text of DBText1. We must provide some direction for the code when the IF statement is false, otherwise the text would be blank.
- The indentation inside of the BEGIN-END block, and inside (below) the IF statement, helps make the code easier to read.

Click **Save**.

## Create the new Label

We are now ready to create the new label that will display the first letter of the families' last names.

- Drop a **Label** (Label1) into the Detail band, in the top left corner above the family name. Make it bold.
- Right-click on the label, and choose **Do Before Print**.

In the code editor, enter the lines as you see them below.

Click on **Insert a Data Field** to enter the Fam fields, Fam Unique ID and Fam Name; and **Insert an Object Attribute** for the label attributes, Visible and Caption.

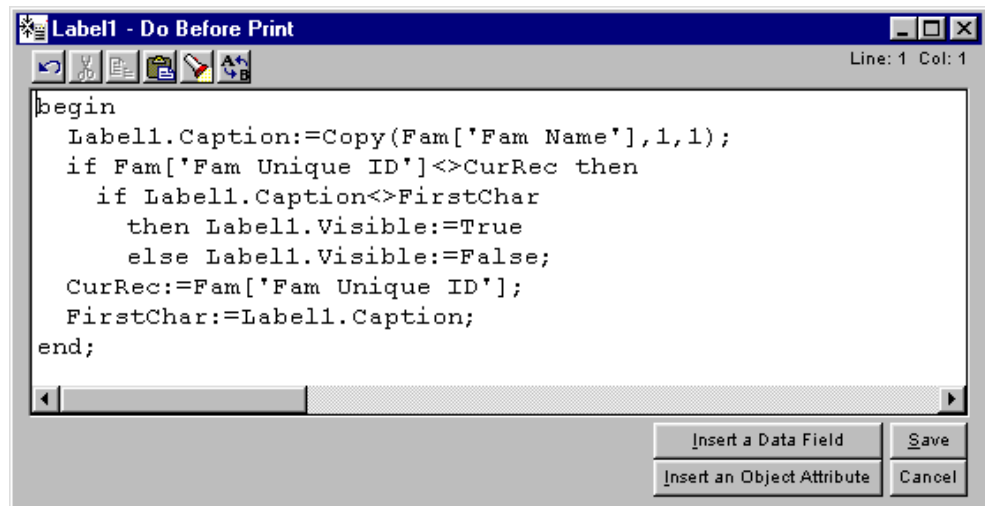


Fig. 4-15

In English, the code is performing these steps:

1. Assign the caption of Label1 equal to the first letter of Fam Fam Name. The Copy(s,i,j) command takes string 's' and copies 'j' letters, starting at position 'i'.
2. If the family's unique ID number is not equal to the current record, then
  - a. If the caption of Label1 (the first letter of the current family's last name) is not already equal to the variable FirstChar:
    - i. Then 'turn on' the visibility of Label1, because it has found a new letter.
    - ii. Else 'turn off' the visibility of Label1, because this family has the same initial letter as the family before it.
3. Regardless of the results of the IF statement, set CurRec equal to the current family's ID number
4. Set FirstChar equal to the current value of Label1.

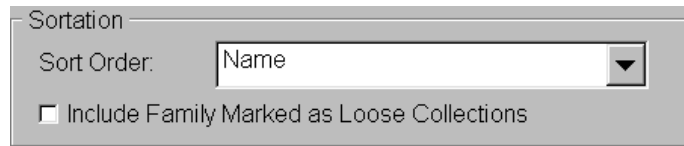
In other words, look at the last initial of the current family. If it is the same as the previous family, turn off the printing of the label. If it is different, turn it on (print it). Set this family's ID and initial as the new test values, to test against the next family.

Click on **Save** to save the code. Click on **File | Save** to save the report. Click on **File | Close** to close the report.

## Run the Report

Run the report as normal.

On the Selection screen, set the Sortation to 'Name'.



Sortation

Sort Order:

☐ Include Family Marked as Loose Collections

The report should look like this:

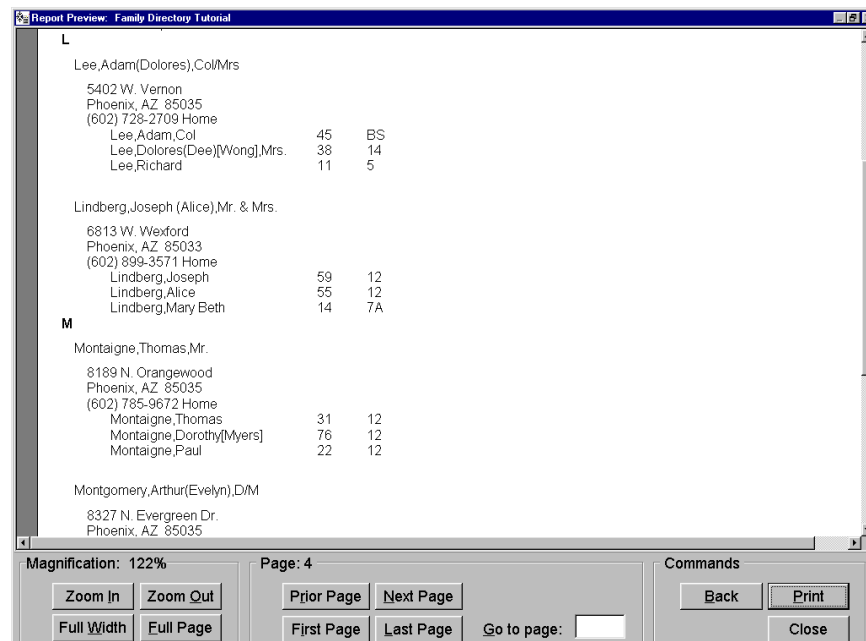


Fig. 4-16

The last initial label only appears once for each letter, before the families whose last names start with that letter.

## 5: Expanding Letters and Statements

### Overview

This chapter explores ways to enhance letter reports in the program. Any letter report, whether predefined, copied or added, allows you to modify the text of the letter. It is in this word-processor-like editor that you can use IF statements, allow for multiple languages, and embed special LISTS.

#### IF statements

You can use the IF-ELSE-END structure described in chapter 4 directly in the text of a letter. This would be used to create multiple letters in one, printing different paragraphs of text depending on the conditions tested.

The financial statement, **Letter to Those Pledging**, has an example of this structure. Click on **Reports | Financial Reports | Financial Statements | Letter to Those Pledging | Next | Next | Modify the Text of the Letter**.

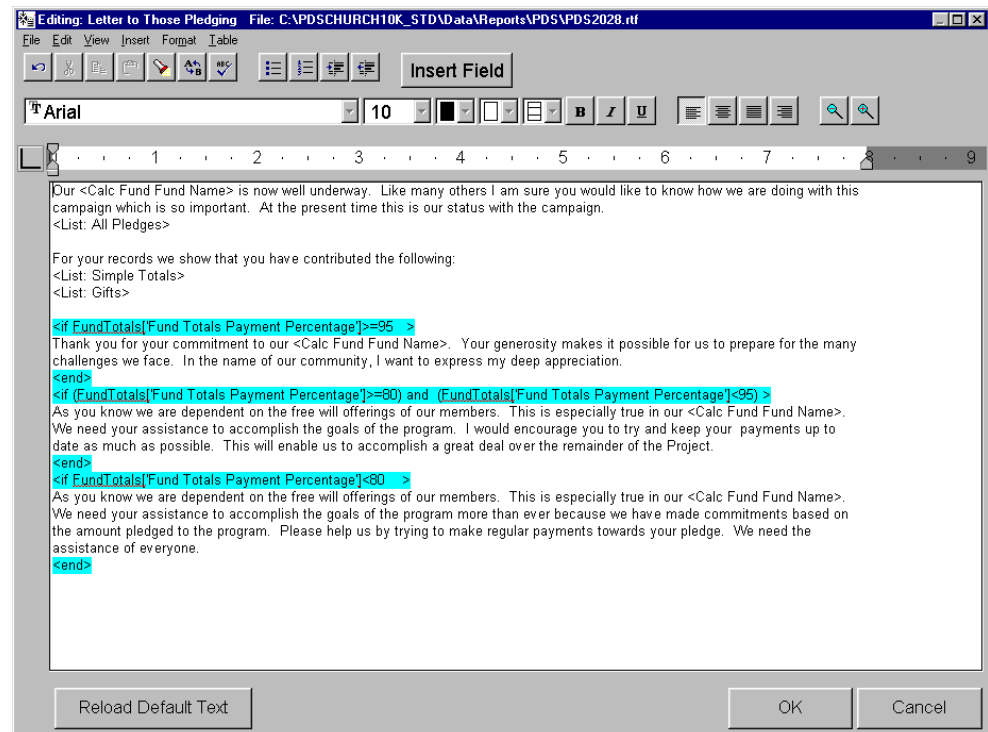


Fig. 5-1

In this letter, different paragraphs are printed depending on the family's Fund Totals Payment Percentage. If the percentage is greater than or equal to 95 ('>=95'), then the first paragraph will print. If the percentage is between 80 and 95 percent, the second paragraph will print. If the percentage is less than 80, the third paragraph will print.

## IF statement format

The format of the IF statement is slightly different than in an ARW report. In the text of a letter, the IF statement is surrounded by triangle brackets, or the less than '<' and greater than '>' signs.

**Example:** `<if FundTotals['Fund Totals Payment Percentage']>=95 >`

Fields used in the statement must have the format of **Category**[**'fieldname'**]. Note the use of single quotes and square brackets around the field name.

**Example:** `<if FundTotals['Fund Totals Payment Percentage']>=95 >`

The categories and field names can be typed in or can be chosen by clicking on **Insert Field**.

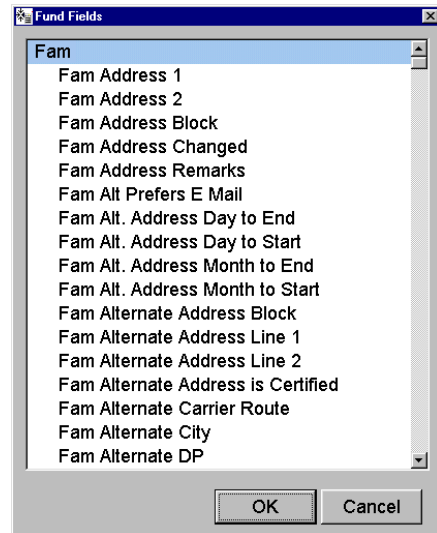
The category names are highlighted in color in the dialog box.

If the category name has a space, such as **Fund Totals**, it must be used without the space in the IF statement, i.e. **FundTotals**.

Field names should be entered exactly as they appear in the **Insert Field** listing.

### Examples:

Fam['Fam Address 1']  
FamPhone['Fam Phone Type']



*Fig. 5-2*

The comparison symbols are the same as in the Advanced Report Writer:

- |    |                          |
|----|--------------------------|
| >  | Greater than             |
| >= | Greater than or equal to |
| =  | Is equal to              |
| <  | Less than                |
| <= | Less than or equal to    |
| <> | Not equal to             |

Fields can be compared to other fields or to values. String values must be enclosed in single quotes.

**Examples:**

Fam['Fam Name']='Smith'  
Fam['Fam Address 1']<>Fam['Fam Alternate Address Line 1']  
Fam['Fam ID/Env Number']>='9000'

## Using ELSE

The ELSE portion of an IF-ELSE-END statement is used to create some action or print a different paragraph when the IF condition is not met.

**Example:**

```
<if Fam['Fam Number of Children']>0 >  
    You are invited to the Children's Day celebration.  
<else>  
    You are invited to the Adults Only celebration.  
<end>
```

You can also use ELSE IF to specify more than one condition.

**Example:**

```
<if Fam['Fam Number of Children']>0 >  
    You are invited to the Children's Day celebration.  
<else if (Fam['Fam Number of Children']=0) and (Fam['Fam Number of  
    Members']>0) >  
    You are invited to the Adults Only celebration.  
<else if Fam['Fam Number of Members']=1>  
    You are invited to the Singles Dance.  
<end>
```

When creating ELSE or ELSE IF conditions, make sure that in the end, all conditions are accounted for, otherwise you may get a blank letter for some families.

**Example:**

```
<if Fam['Fam Number of Children']>1 >  
    You are invited to the Children's Day celebration.  
<else if Fam['Fam Number of Children']=1 >  
    You are invited to the Families with Only Children celebration.  
<end>
```

In the above example, the program would generate blank letters for families with no children at all. The correct structure would be like this:

**Example:**

```
<if Fam['Fam Number of Children']>1 >  
    You are invited to the Children's Day celebration.  
<else if Fam['Fam Number of Children']=1 >  
    You are invited to the Families with Only Children celebration.  
<else>  
    You are invited to the Childless Families celebration.  
<end>
```

---

**Important Note:** All IF statements – whether using ELSE, ELSE IF or IF alone – must end with an END statement in the form <end>.

---

## Using AND and OR

You can test for more than one condition in a single IF statement by using AND and OR.

Each separate condition must be enclosed in parentheses. The conditions can use different fields, and they can test for different values.

### Example:

```
<if (Fam['Fam Number of Children']>0) and (Fam['Fam City']='Phoenix, AZ') >  
  ↑                               ↑  ↑                               ↑
```

**AND** is used to say that both (or all) conditions must be true in order for the following text to be printed.

### Example:

```
<if (Fam['Fam Number of Children']>0) and (Fam['Fam City']='Phoenix, AZ') >  
  You are invited to the Children's Day celebration to be held in Phoenix Municipal  
  Park.  
<end>
```

**OR** is used to say that either (or any) of the conditions can be true for the text to print.

### Example:

```
<if (Fam['Fam Suffix']='Jr.') or (Fam['Fam Suffix']='Sr.') >  
  You are invited to the Association of Suffix Holders (ASH) celebration.  
<else if (Fam['Fam Suffix']='II') or (Fam['Fam Suffix']='III') >  
  You are invited to the Association of Seconds and Thirds (AST) celebration.  
<end>
```

## Multiple Languages

One of the most common uses of IF statements is to create letters that print in different languages.

All it takes is the use of the proper character set on the keyboard and typing the text in the language you wish to use. This does not mean that the program will translate into different languages. It does mean you can enter different versions of the letter in the same report and the correct letter will print for each family. In other words when you send letters to many families, you can have the program print a letter in English, Spanish or any other language by running just one report.

There are a few requirements that must be kept in mind if you wish to use this feature:

- The language you intend to use in the letter must be entered on the Member Detail Screen in the Language field for at least the head of each family.
- If you use the concept of main language, secondary language (e.g. English/Spanish, Spanish/English), make sure the language you wish to use in letters is first.
- The name of the language must be spelled exactly the same in each combination.
- The salutation used with the family name letter should be in the same language as that of the head member. In other words it should match the language of the letter.
- If you will send letters to members using the multi language feature, you need to make sure each one has the proper language entered on the Member Detail Screen in the Language field.



## Inside Address Style

The **Family Salutation** field has provision for the use of titles. If you use multiple languages, make sure you have adjusted those titles that might require change, i.e. Mr. or Mrs. for Senor or Senorita.

Select **Reports | Family Reports | Family Easy Reports | Add | Letter | Next | Next**.

This will bring you to the Set the Letter Layout Screen. On this screen you need to adjust the different Styles and the Text of the Letter so they are set up for the different languages.

There are two screens involved in changing the Inside Address Style. Fig. 1 shows the Letter Layout screen. Fig. 2 shows the Inside Address Style. Select the style you wish to use, formal or informal, and then click on the **Edit Style** button. This will bring you to the screen shown in Fig. 2.

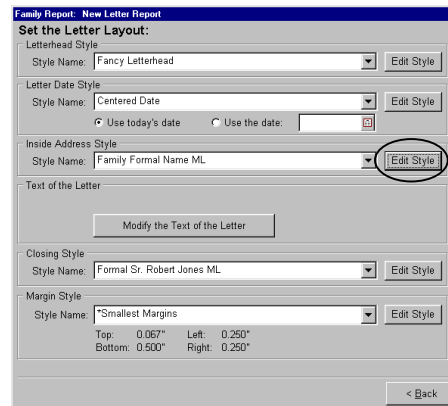


Fig. 5-3

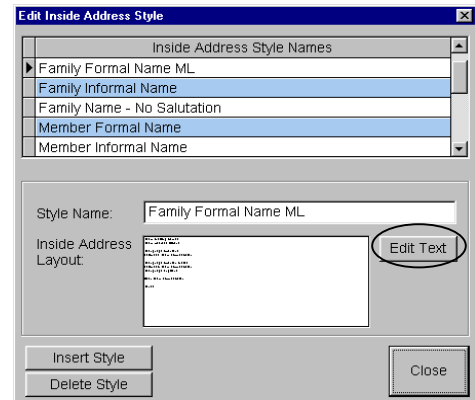


Fig. 5-4

Click on **Edit Text**. This will allow you to make the necessary changes to the Inside Address style. Here you can enter or modify the style. Depending upon the language, you can type in the term you wish to use for a greeting, i.e. the word Dear, Estimado etc.

<Fam Mailing Name>

<Fam Address Block>

<Language: Spanish (Male)>

<if pos('Sr. é Sra.',Fam['Fam Formal Sal'])>0>

Estimados <Fam Formal Sal>,

<else if pos('S/S',Fam['Fam Formal Sal'])>0>

Estimados <Fam Formal Sal>

<else >

Estimado <Fam Formal Sal>

<end>

This is the Address Style. The first two terms uses the mailing name and the address for each family.

The next term tells the program to look for the first family that has a male member with the term Spanish in the Language field.

If it uses Sr. é Sra. in the salutation, print the word 'Estimados'.

Else, if it uses S/S in the salutation, print the word 'Estimados'. If not, print 'Estimado' and then the family name.

---

**Note:** The command <Language: Spanish> is the same as <if Fam['Fam Letter Language']='Spanish'>

---

<Language: Spanish (Female)> Estimada <Fam Formal Sal>,	If the person is a female and has the term Spanish, it will print 'Estimada'.
<Language: English> Dear <Fam Formal Sal>,	If the person has a different language (such as English), print the word 'Dear' and the family name.
<End>	End tells the program to go on to the next family.

If you make changes to the style so that it will print the proper language salutation, make sure that you use the Save As button and give the Style a new identifying name, i.e. Family Formal Name ML.

## Modify the Text of the Letter

The body of the letter can also be created to have multiple language paragraphs.

<Language: Spanish> Esta es una cara de muestra escrita en Espanol.	The first instruction tells the program to print what follows if the language field is the word Spanish.
<Language: English> This is a sample letter written in English.	If the language field is English, print the following.
<End>	<End> tells the program to continue to the next family.

This method can be as simple as the one above or it can be more complex. For example: this report could have many more languages as long as the same format is used – <Language: Spanish>, <Language: English>, or <Language: Vietnamese> with the proper text inserted after each.

## Closing Style

This style is used to modify the closing portion of the letter. Use the same method as was used in the Inside Address Style.

<Language: Spanish> Sinceramente, Sr. Robert Jones Administrador de Oficinas	The first instruction tells the program to print what follows if the language field is the word Spanish.
<Language: English> Sincerely, Mr. Robert Jones Office Administrator	If the language field is English, print the following.
<End>	<End> tells the program to continue to the next family.



Fig. 5-5

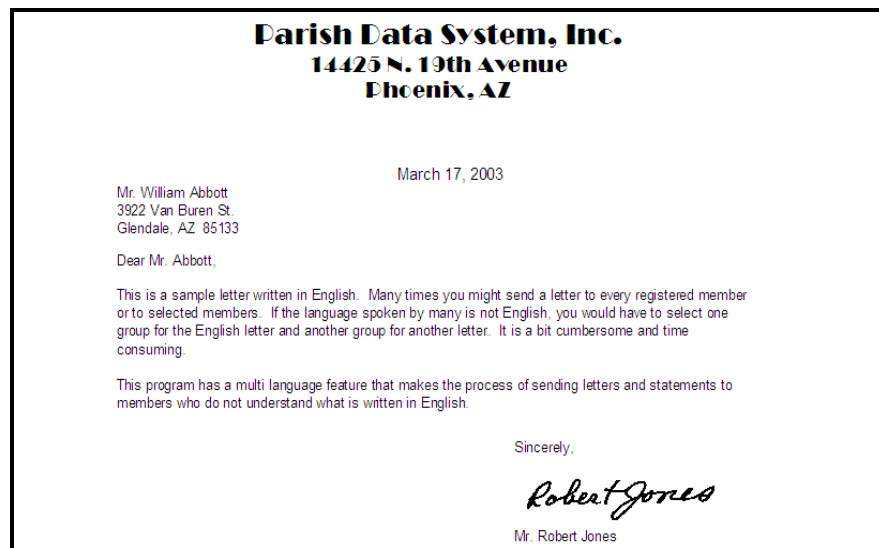


Fig. 5-6

---

**Note:** There is a free service available on the Internet that will translate English to Spanish, French, Italian and a few other languages. There is some cost if you want to go beyond the free aspect of the service. However, it can be used to translate a letter into Spanish that might be satisfactory. You can call up this service by entering [www.freetranslation.com](http://www.freetranslation.com). It is very easy to use. Type your letter in English and copy it to the clipboard. Then call up this website and paste the letter into the space provided. Then click on the request for a free translation. When the translation is finished, copy it to the clipboard and then paste it into your report.

---

## Embedded Lists

An embedded list is an Advanced Report Writer report inserted into a Letter. The Church Office Management report, Billing Statement, has several examples.

Click on **Reports** | **Financial Reports** | **Financial Statements** | **Billing Statement** | **Next** | **Next** | **Modify Text of the Letter**.

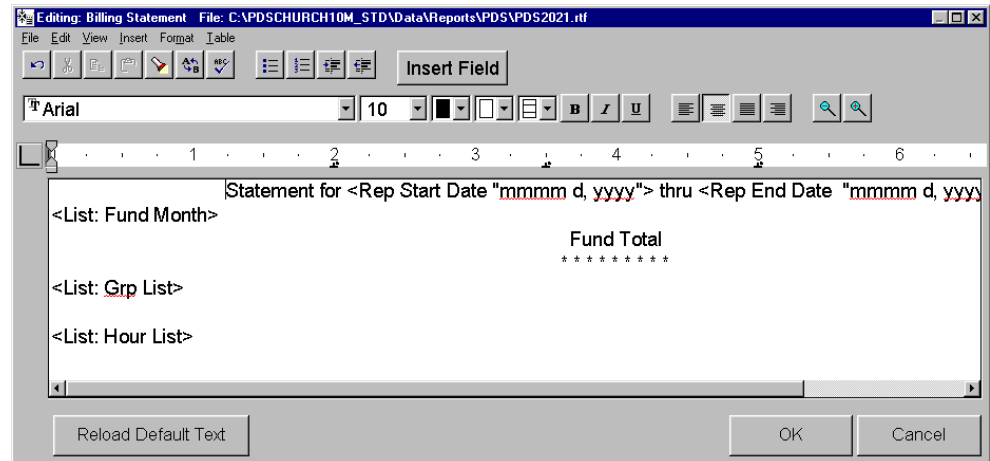


Fig. 5-7

Embedded lists can be added to a letter by typing the name in triangle brackets '<>'. The format is '<List: reportname>'. These are the lists available in Church Office Management.

<List: All Pledges>	Total of all families for all pledges for selected funds.
<List: Comm>	Communion data for each member.
<List: Confirm>	Confirmation data for each member.
<List: Coupon Bottom>	Prints a return coupon at the bottom of the letter.
<List: Coupon Top>	Prints a return coupon at the top of the letter.
<List: Delinq>	Family's delinquency information.
<List: Fund Month>	Fund name and total paid each month.
<List: Gifts>	Additional gifts totals.
<List: Grp List>	Fund groups with Due, Paid, 'Balance for stmt'.
<List: Grp List 2>	Fund groups with Due, 'Total Tax deduct.', balance
<List: Grp List 3>	Fund groups showing Amt Paid only.
<List: Hist List>	Activity history for the fund selected.
<List: Hour List>	Hours due, completed, balance; similar to GrpList.
<List: IRS Note>	Special tax note at the end of tax statements.
<List: Mem Amt>	Members of family and amount each has paid.
<List: Receipt History With Desc>	Activity history with date and description.
<List: Running Total>	Activity history with cumulative due/paid totals
<List: Simple Totals>	Total Pledged, Paid, Balance for all funds selected.

The lists for Formation, Ledger, and Ledger/Payroll are found in Chapter 11: Appendix.

The underlying ARW structure of embedded lists can be viewed by clicking on **File | Edit List Sections**.

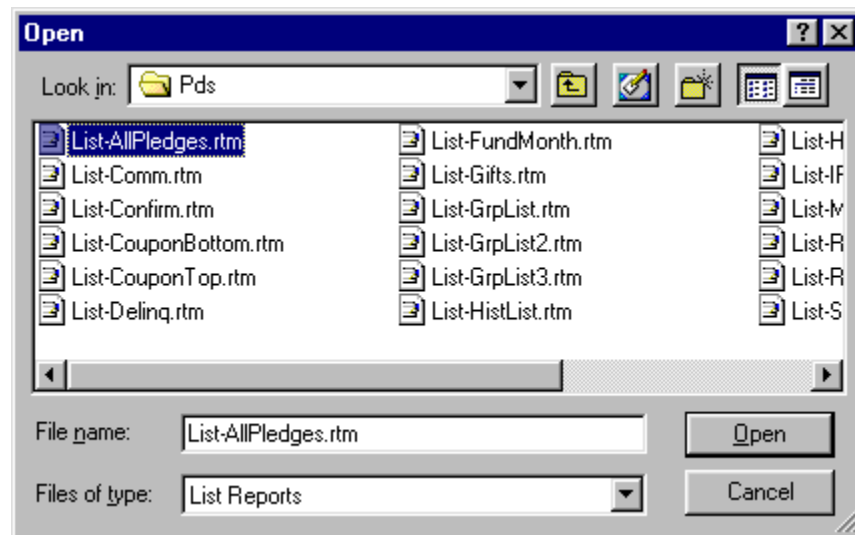


Fig. 5-8

This is the **List: All Pledges** embedded list as seen in the ARW editor.

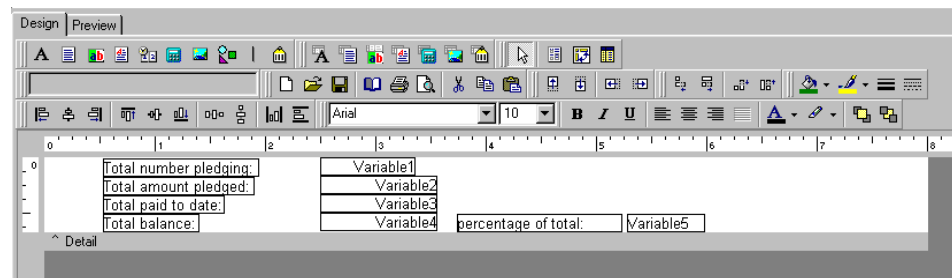


Fig. 5-9

Each embedded list can be modified following the guidelines in the first four chapters. Once modified, click on **File | Save** to overwrite a predefined list, or **File | Save As** to save the list under a new name. New embedded lists can be created from scratch in the Advanced Report Writer area.

Each embedded list must be saved with the following format: **List-filename.RTM** (note the 'dash' and the RTM extension). Embedded lists must be saved in the program's reports directory. For example, C:\PDSCHURCH\DATA\REPORTS\PDS.

## Exercise

In this exercise, we will create an embedded list report – a list of members and their ages, to be inserted into a family letter.

### Step 1: Start the report

The first step is to create the ARW report, which will be saved as an embedded list. The easiest way to do this is to go into the text of the letter that this list will be embedded.

- From the Main Menu, click on **Reports | Family Reports | Letters/Statements | Family Welcome Letter**.
- Click on **Next | Next | Modify the Text of the Letter**.
- Click on **File | Edit List Selections**. The program will automatically open the **Browse** dialog, with the Data/Reports/PDS folder open, showing all of the other embedded lists.
- Name the file “List-MemList.rtm”. The filename must begin with ‘List-’, have no spaces and must have an RTM extension. Click on **Open**.

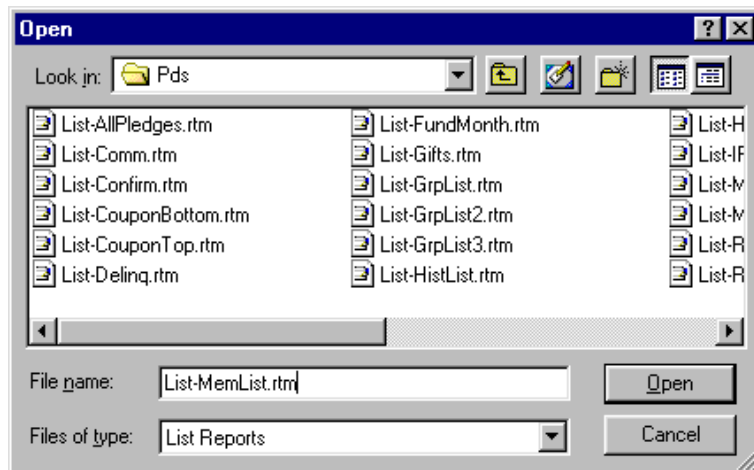


Fig. 5-10

The Advanced Report Writer editor will open automatically.

- Click on **Report | Header** and **Report | Footer** to remove the Header and Footer bands.
- Click on **Report | Title** to add the Title band.

The workspace should look like the figure on the right.

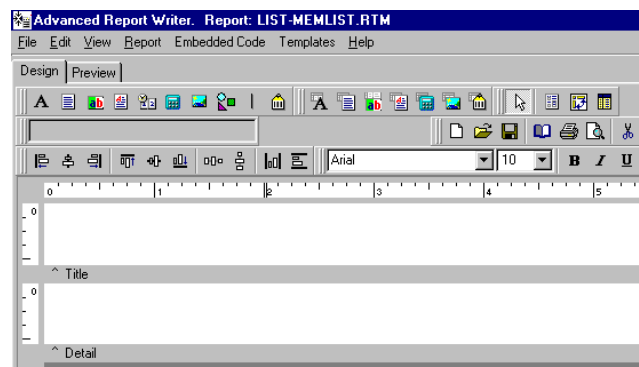


Fig. 5-11

### Step 2: Add the fields

- Drop a **DBText** object into the top left corner of the Detail band. Choose **Mem Mem Name Format 1** as the data pipeline.
- Right-click on the object and choose **Autosize**.
- Drop another **DBText** object next to the Member Name, at 2 ½ inches. You can right-click on the object and use **Position** if necessary.
- Choose **Mem Mem Age** as the data pipeline. Right-click and choose **Autosize**.
- Click and drag the Detail band up until there is no white space under the fields.
- Drop a **Label** object (Label1) into the top left corner of the Title band. Change its value to 'Member Name'. Make it bold.
- Drop another **Label** object (Label2) next to the first label, at 2 ½ inches. Change its value to 'Member Age'. Make it bold.
- Click and drag the Title band up until there is no white space under the labels.

The screen should look like this.

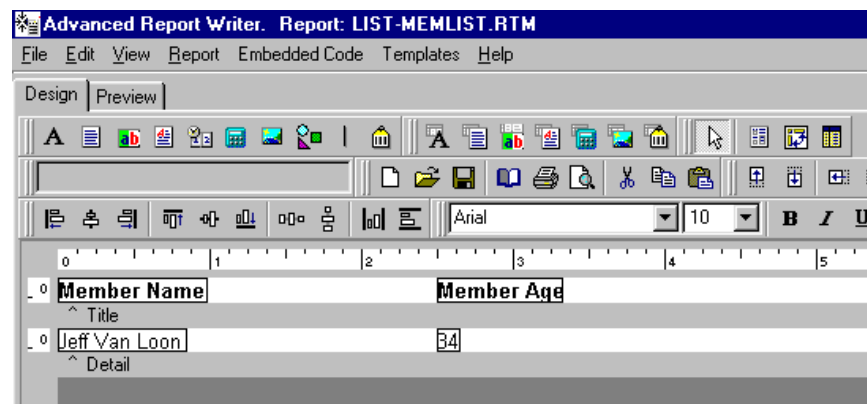


Fig. 5-12

### Step 3: Save the report.

- Click on **File | Save**. The program will save the embedded list under the filename we chose in step 1.
- Click on **File | Close** to close the ARW editor.

### Step 4: Use the embedded list in the letter.

We can now add the embedded list to the letter.

- At the end of the letter, on the line after the last sentence, insert the embedded list command: <List: MemList>.
- Note the triangle brackets around the name, and the fact that we use a colon and a space in the name. Also, the extension, RTM, is not used.

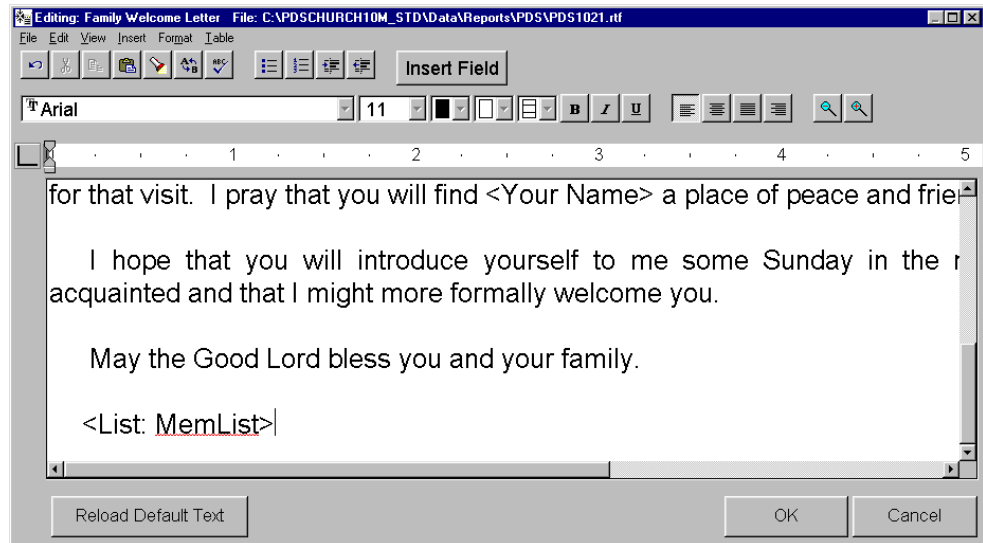


Fig. 5-13

Click **OK**, and run the report as normal.

The preview should look similar to this.

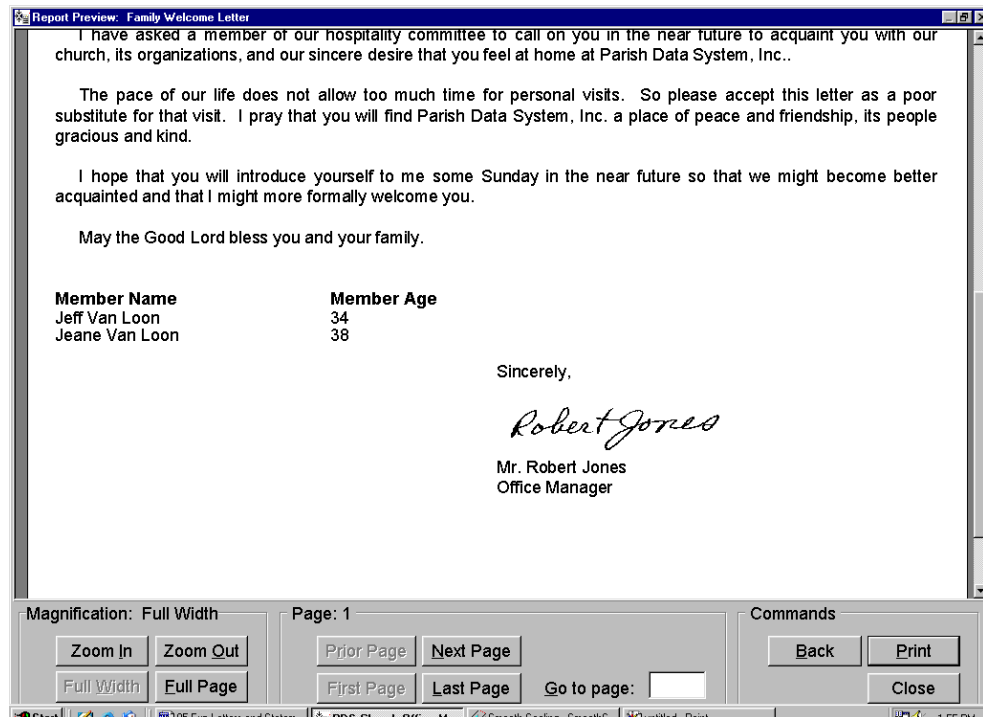


Fig. 5-14



## 6: Scripting Language

### Overview

Embedded code can affect things inside the report, but there are many situations where you would like to have some control outside of the report, where embedded code has no effect.

For example, you might want to ask the user some questions about what is going to be in the report, in order to customize it. This often eliminates the need to have multiple reports that do almost the same things. Or, you might want to create a report that is actually an export or an import of data. In this case we may never even generate any sort of printout at all. Another thing you might want to do is sort or manipulate the data in a way that is not designed into the system. None of these could be done with embedded code inside the report.

We need some way of controlling things outside the reports. That is Scripting. Scripting is the idea of having a programming language built into the program that allows you to control how that program runs. Scripting has been around for a long time but became more prominent when Microsoft started using it inside its products.

We have built scripting into the report portion of PDS products. So, a report can have a script associated with it. When the report is run, that script is activated. The program language we are using is Pascal, just like in the embedded code. It does have some differences, which will be pointed out as we go.

### Accessing the Script

The script can be created and changed from the Report Menu screen. Select the report and click on the **Adv. Script** button. This will bring up the Advanced Script dialog. Click on the **Pascal Script File** tab.

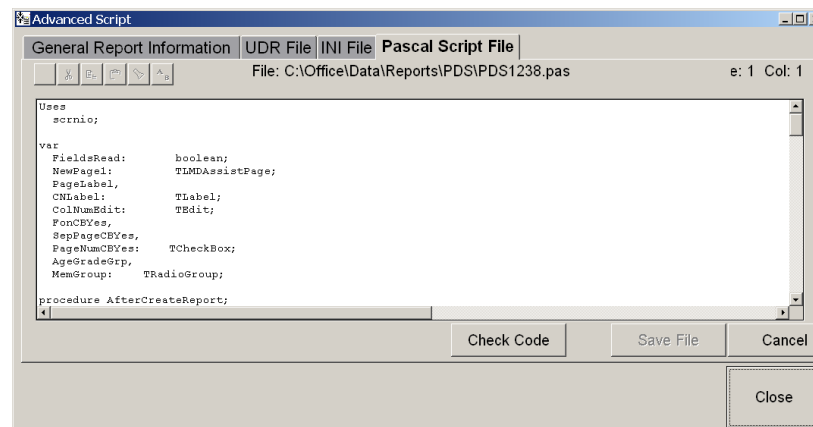


Fig. 6-1

It is here that we can edit the code of the script. Writing a script is different than writing the embedded code. Unlike the embedded code, all the code of a script is in one place. There is not a separate screen for each routine. You must enter all of the code including the procedure declarations. The names of the fields are different than those in the embedded code. Errors that you make may not show up until you run the report. There is a **Check Code** button that checks for many simple syntax errors but cannot detect a lot of the most common errors.

## Events

In the embedded code in reports we connected code to object events. In the script you also attach code to events but they are program events that happen as the program runs. For example, we can hook to an event called **AfterCreateReport**. The code will be run after the report is selected (The term Create is misleading but accurate. Every time you select a report the program creates a 'Report' object). You 'hook' into an event by including a procedure or function in your script with the name of the event you want to 'handle'. The program knows to call your routine when the proper time comes.

The list of procedure and functions that can be used are in the table below. These events are also in the order they are executed.

procedure <b>AfterCreateReport</b> ;	Called right after the report is selected.
procedure <b>AfterLoadReport</b> ;	Called after report loaded.
procedure <b>CreateExternalData</b> ;	Called right after <b>AfterLoadReport</b> . This gives us a change to set up any table we need for this report. This may not be the best place to add the records to those tables since the selections have not yet been attached, but it is convenient place to create the tables.
function <b>BeforeReport</b> : boolean;	Called just after the preview button has been pressed. It gives you a chance to detect any problems with the values they have selected and cancel the report. The report can be canceled by setting <b>Result:=false</b> ; in the event.
procedure <b>BeforeSelectReport</b> ;	Called before simple filters have been attached.
procedure <b>AfterSelectReport</b> ;	Called after simple filters have been attached.
procedure <b>AfterSelectionsBuilt</b> ;	Called after addition selections are added. Many things are usually done in this event since all the tables have their selections attached. For example we might go through the tables and get grand totals so that we can refer to a percentage of the total in the report.
function <b>PrintDefaultReport</b> : boolean;	Called before printing. You can use this event in conjunction with the <b>BeforeSetupPrint</b> to tell the program that this is not a 'normal' report. If we are doing an export or an import, where there is no report to be generated, we would add this event and set <b>Result:=false</b> ; We also need to put in <b>BeforeSetupPrint</b> with the same <b>Result:=false</b> . This causes the preview screen to not show.
function <b>BeforeSetupPrint</b> : boolean;	See the description above.
procedure <b>BeforePrintReport</b> ;	Called before we begin printing.
procedure <b>AfterPrintReport</b> ;	Called after the report has printed.
procedure <b>AfterReport</b> ;	Called after the report even if this report was not designed to print anything.
procedure <b>FreeExternalData</b> ;	Called at the end of the report. This gives us a chance to free any data tables that we may have created in this script (probably in the <b>CreateExternalData</b> event).
procedure <b>BeforeDestroyReport</b> ;	Called at the very end.

For example, we want the program to flash a simple message that says hello right after it has built selections but before the report is run. We would go to Member Reports and add an Advance Report (it actually could be an Easy List, a Label or any kind of report). We open up the advance script and you would enter in the Pascal Script:

```
procedure AfterSelectReport;
begin
    ShowMessage('Hello');
end;
```

Save the report and run it. When you press the **Preview** button the program should popup the message. Since we did not add any fields to print in this report, it will then bring up a blank page.

There are also two special events that exist only in Church Office, Formation Office and School Office. The events occur (or ‘fire’) while the report is running.

procedure OrderMailTbl;	Called after each entry is made in to the Mail Table.
procedure CallAfterCalc;	Called on each family after the Family table has finished its calculations for that family.

**OrderMailTbl** is called on every new entry made to the Mail table. The Mail table is used to do special sorts where billing addresses and courtesy copies are going to be included. To speed up the program, **OrderMailTbl** will not be called unless there is a line in the default of the UDR that reads `OrderMailTbl=1`. You may also need to set `ForceMailTbl=1` as well. This can be used to create reports stored in complicated orders very easily by setting **PDSMDData.MailTblKey.AsString** to the value that you want to sort by inside the **OrderMailTbl** event.

**CallAfterCalc** is called when the report goes to a family. After the program has calculated all of its internal items, it will call this procedure. To speed up the program, **CallAfterCalc** will not be called unless there is a line in the script that says `UseCallAfterCalc:=true;`. This is usually in the **AfterCreateReport** event.

## Objects

In reports, we have touched on the idea of objects. Objects were items put into the report with the Advanced Report Writer editor/designer. In scripts, objects are much more. Everything on the screen and everything in tables are objects. For example, the family table is really an object that is stored in an object called a *datamodule*. So, to go to the first family we would enter a line:

```
PDSMDData.FamTbl.First;
```

This is telling the computer to go to the PDSMDData datamodule and find the FamTbl object and run the routine that is there called ‘First’.

## Additional Code Statements

The program language used in scripts has a few additional statements. They are:

- With statements
- Try-Finally statements
- Try-Except statements
- Exit statements
- Break statements
- Continue statements
- Goto statements.

There is also more ways of putting in comments.

## With Statements

The With statement is a short cut used when there are many attributes of an object being set. For example:

```
Label1.Caption := 'Test';
Label1.Top:=100;
Label1.Left:=200;
Label1.Width:=100;
```

Can be written using the With as:

```
With Label1 do
begin
  Caption := 'Test';
  Top:=100;
  Left:=200;
  Width:=100;
end;
```

If there is more than one object being used, the With statements can be nested inside one another or the objects can be separated by commas. Also, an object may have an attribute that has attributes itself (i.e. font) – these can be used in With statements as well. If a With gets too complicated, its clearer not to use them.

## Try-Finally and Try-Except statements

The Try-Finally and the Try-Except statements are used to handle error conditions. The Try-Finally statement says to Try and do something and no matter what happened, do what is in the Finally section.

For example, the following code will try to create a file named 'A:\test.txt', and write a few lines of text to it. No matter what happens (i.e. the disk is full), the program will run the **CloseFile** statement.

```
Var
  Fout: Text;
Begin
  Assign(Fout, 'A:\Test.txt');
  Try
    Rewrite(Fout);

    Write(Fout, 'Test');
    Writeln(Fout, ' The end of a line');

  Finally
    CloseFile(Fout);
  End;
End;
```

The Try-Except statement is similar, but the Except statement is run only if an error occurs. For example, we could Try to divide two numbers. We need to have an Except in case the denominator is zero.

```

Var
  I, J: integer;
Begin
  I:=5;
  J:=0;
  Try
    I:=I div J;
  Except
    ShowMessage('Cannot do '+
      IntToStr(i)+'/'+IntToStr(j));
  End;
End;

```

There can be any number of statements inside the Finally clause. We might even nest these two types of statements. If an error is encountered and there is no Except, you will get an unexpected error dialog. Try-Finally and Try-Except statements are rarely required but they make the report better able to tolerate problems without giving the user an 'Unexpected Error' messages.

## Exit, Break, Continue and Goto statements

The Exit, Break, Continue and Goto Statements allow you to control the flow of the program when some situation arises. Using the Exit statement causes the program to get out of the procedure or event that it is in and return to whatever called it and start running from there. The Break and the Continue statements are used inside of loops. The Break statement causes the program to jump out of the loop. The Continue statement will cause the program to go to the bottom of the loop. The Goto statement allows us to put a label in and go to that label. Here is an example that uses all 4 statements. This example is very contrived and really does not do anything.

```

Procedure Test;

Label TryBackwards;    // define a label for the Goto

Var
  I, J: integer;
Begin
  If PDSMDData.FamTbl.RecordCount=0
    Then Exit;    // get out of this routine completely
  J:=0;
  For I:=1 to 5 do
    begin
      PDSMDData.FamTbl.First;
      If PDSMDData.MemTbl.RecordCount=0
        Then Continue;    // get the next I
    end
  end

```

```

While not PDSMDData.FamTbl.Eof do
Begin

    If PDSMDData.MemTbl.RecordCount=1
        Then Break;    // get out of the while loop

    If PDSMDData.MemTbl.RecordCount=1
        Then goto TryBackwards;    // jump to the
                                    // label

    J:=J+1;
    PDSMDData.FamTbl.Next;
End;

TryBackwards:

    PDSMDData.FamTbl.Last;
    While not PDSMDData.FamTbl.Bof do
        Begin
            J:=J+1;
            PDSMDData.FamTbl.Prior;
        End;

End
End;

```

In most cases you do not need to use these statements. But, in the rare case that you do they can save a great deal of work.

## More Comments Options

In addition to the braces `{ }` that the embedded code uses for comments, script can use an open parentheses followed by an asterisk (`*` to start a comment, and an asterisk followed by a close parentheses `*)` to close a comment. Also, anything after a `//` until the end of the line is considered a comment in scripts.

## Detecting Errors

There is a button on the script editor to check the code. This only does the simplest syntax checking. It cannot detect a great many problems. When you run the report, the program will bring up a dialog box giving as much information as it has:

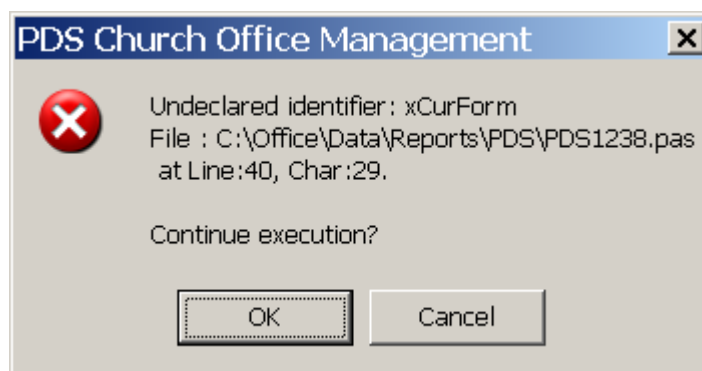


Fig. 6-2

In this case, it is telling you that it could not find the variable named `xCurForm` and that it was on line 40, character 29 when it happened.

## Common Problems and Things to Help

There are a few things that you need to watch out for in scripting.

- √ The program crashes more when writing scripts.

There are less protections from the report doing something that stops the program dead. Windows XP is a much better operating system to use when developing. Lockups are less likely to crash the computer. Also, it handles resources differently. This means we don't run out of memory as fast.

- √ It is very easy to make a mistake in the name of the events.

The program will not tell you that you have the name wrong. It just won't run that procedure. One way to avoid this is to put a `Showmessage` command in the procedure when you first add it to make sure everything you think is running is actually running.

- √ Scripts have problems if you don't initialize your variables.

## Notes



---

## 7: Adding and Accessing Screens

### Overview

One common reason for using scripts is to ask questions of the user and then to use the answers to customize the report. In this chapter we will continue on with the directory example that we have been using in the prior chapters. We will add a layout screen to this example. A great deal can be learned from the predefined report already in the program. The Member Involvement reports are useful for seeing how to make layout screens.

### Simple Messages and Questions

We have already seen one way to inform the user of things using the ShowMessage statement. The simplest way to get information from the user is to bring up a question in a dialog box with buttons that say 'Yes', 'No', 'Cancel', 'Ok', etc. We can do that using the built-in function PDSMessageDlg. This will display a message to the user in a dialog with buttons at the bottom. It will return which button the user presses. The call to this function has the form:

```
PDSMessageDlg(S, MessageType, ButtonSet, HelpContext);
```

'S' is the message that we want to display.

'MessageType' is what general type of message this is. The type of message controls the caption of the dialog box and what icon is displayed. The choices are:

mtWarning	A message box containing a yellow exclamation point.
mtError	A message box containing a red stop sign.
mtInformation	A message box containing a blue "i".
mtConfirmation	A message box containing a green question mark.
mtCustom	A message box containing no bitmap. The caption of the message box is the name of the application's executable file.

The 'ButtonSet' is a list of buttons inside of an **MkSet** function call. The choices for buttons are:

mbYes	A button with 'Yes' on its face.
mbNo	A button the text 'No' on its face.
mbOK	A button the text 'OK' on its face.
mbCancel	A button with the text 'Cancel' on its face.
mbAbort	A button with the text 'Abort' on its face.
mbRetry	A button with the text 'Retry' on its face.
mbIgnore	A button the text 'Ignore' on its face.
mbAll	A button with the text 'All' on its face.
mbNoToAll	A button with the text 'No to All' on its face.
mbYesToAll	A button with the text 'Yes to All' on its face.
mbHelp	A button with the text 'Help' on its face.

The last item is the 'HelpContext', which for us would always be the number zero. This function will return which button was pressed. This will be one of the following values:

mrNone	The X in the top right corner was pressed.
mrYes	The Yes was pressed.
mrNo	The No was pressed.
mrOk	The Ok was pressed.
mrCancel	The Cancel was pressed.
mrAbort	The Abort was pressed.
mrRetry	The Retry was pressed.
mrIgnore	The Ignore was pressed.
mrAll	The All was pressed.

For example, we want a message asking if the user is sure, with buttons for Yes and No:

```
If PDSMessageDlg('Are you sure?', mtConfirmation,
                 MkSet(mbYes,mbNo), 0)=mrYes
Then ShowMessage('You said Yes')
Else ShowMessage('You did not say Yes');
```

You can save the result in an integer if you need to have more than one button. For example:

```
Var
  I: integer;
Begin
  I:=PDSMessageDlg('What should we do?', mtError,
                  MkSet(mbRetry,mrIgnore,mbCancel), 0);
  If I=mrRetry
  Then ShowMessage('You said Retry')
  Else If I=mrIgnore
  Then ShowMessage('You said Ignore')
  Else ShowMessage('You did Cancel or pressed the X');
End;
```

Where there is more than one question or the response is some text the user must enter, we need to create a new screen in the report dialog. We sometimes refer to this as a second layout screen.

## Screen routines

In order to make this easier we have included a library of common routines called 'ScrnIO.pas', which stands for 'Screen Input/Output'. It is in the reports folder of all of our programs. It can be accessed in a script by putting

```
Uses
  scrnio;
```

at the top of the script.

---

**Note:** In all following examples, you don't have to use the variable names S, L, E, and N – use the names of the variables *you* declare or literals of the right type (i.e. strings where strings go and integers where integers go). Also, don't put in the variable types, such as string or TCheckbox – they are here for reference only.

---

ScrNIO adds the following routines to the script.

NewScreenPage;

This adds a new page to the report wizard and sets everything up to use it. This initializes all the internal variables used to position fields.

AddHeading(S: string; L: TLabel);

Add a heading label with the caption of S. The actual label object is returned to you in the variable you put as L.

For example:

```
AddHeading('Talent/Ministry Options:', MyLabel);
```

Would create:

**Talent / Ministry Options:**

AddHeadingCheckBox(s: string; E: TCheckBox);

Add a heading label with a check box.

AddLabel(s: string; L: TLabel);

Add a simple label to the screen.

Talent List

AddEdit(s: string; L: TLabel; E: TEdit; N: integer);

Add a label and an edit box with 'N' equal to the maximum number of characters to be entered.

Starting Month: 5

AddPhone(s: string; L: TLabel; E: TDBPhone; N: integer);

Add a label and an edit box, formatted for a phone number.

AddDate(s: string; L: TLabel; E: TDBDateBtn; N: integer);

Add a label and an edit box, formatted for a date, with a button that brings up the calendar.

Date for the Alternate Address: 06/23/2003

AddAmt(s: string; L: TLabel; E: TDBAmount; N: integer);

Add a label and an edit box, formatted for an amount, with a button that brings up a calculator.

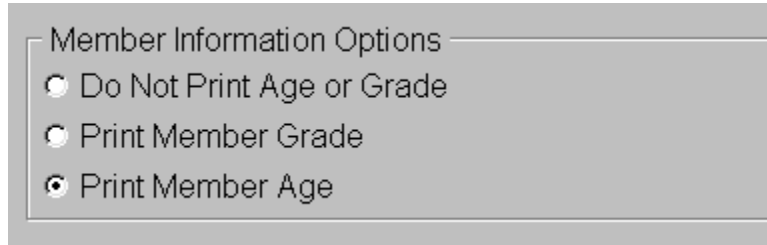
Families with an outstanding balance of more than: \$0.00

AddTotal(s: string; L: TLabel; E: TDBAmount; N: integer);

Add a label and a box, formatted for an amount like the AddAmt. The difference is the AddTotal box does not have a button and is disabled so amounts can't be entered. This is used when you want to display an amount which is the sum of other boxes on the screen.

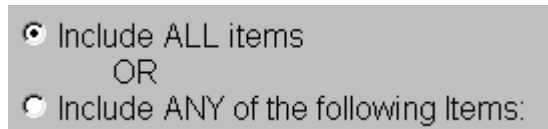
AddRadioButtonGroup(S: string; List: string; E: TRadioGroup);

Add a radio button group that has a caption in S and a list of radio buttons in the List. Each button in the list is separated by a comma (i.e. 'Red, Yellow, Blue'). The object is returned in the last item. Which radio button is selected can be determined or set with the E.ItemIndex (where E is the name of the TRadioGroup Object you have declared and used as the third parameter).



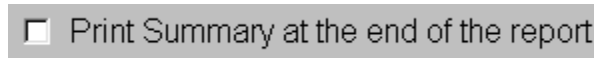
AddIncAll(S1, S2: string; IncOr: boolean; E1, E2: TRadioButton);

Add a pair of radio buttons that indicate INCLUDE ALL and INCLUDE ANY OF the items in a table. The OR that comes between the messages can be turned on or off with IncOr.



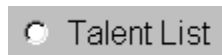
AddCheckBox(s: string; E: TCheckBox);

Add a check box with a caption.



AddRadioButton(s: string; E: TRadioButton);

Add a Radio button with a caption.



AddDropDown(s: string; L: TLabel; E: TComboBox; N: integer);

Add a drop down combo box with a caption. The maximum number of characters is set to N.



AddLine(N: integer);

Add a line of 'equals signs' as a separator. 'N' is the number of equals signs.

AddBlank;

Add a space under the last object placed on the screen.

AddPage(Pg: LMDAssistPage; N: integer);

This adds a new page to the report wizard like NewScreenPage but returns the new page object in the variable you put as 'Pg'. 'N' is the position of the new page. Usually this is 3. This is useful if you need access to the page object.

## Screen Objects

These routines create objects and add them to the screen. Before we can use any of these routines we must declare the objects we are going to have on the screen. We declare the object we want to use in a 'var' statement at the top of the script. Some common types of objects are:

TLMDAssistPage	A Page on the wizard.
Tlabel	A Label on the screen.
TEdit	A String edit field.
TComboBox	A String edit field with a pull down list.
TDBDateBtn	A Date edit field with a button to bring up calendar. It can be attached to a database field.
TDBAmount	An Amount edit field with a button to bring up a calculator. It can be attached to a database field.
TDBPhone	A Phone edit field. It can be attached to a database field.
TCheckBox	A single check box with a captions.
TRadioButton	A single radio button with a caption.
TRadioGroup	A set of radio button grouped inside a box with a caption.
TButton	A simple button.
TPDSDBGGrid	A grid that can attach to a database.

So, for the example we have been creating we need a new page to the report wizard. We need a couple of labels and three check boxes and 2 radio button groups. This would be declared as:

```
var
  NewPage1:          TLMDAssistPage;
  PageLabel,
  CNLabel:           TLabel;
  FonCByes,
  SepPageCByes,
  PageNumCByes:      TCheckBox;
  AgeGradeGrp,
  MemGroup:          TRadioGroup;
```

## Create the New Screen

Now we need to create the objects on the screen. We want the new page of the wizard to look like this:

The screenshot shows a window titled "Family Report: (Copy) Family Directory". It contains two main sections: "Page Layout Options" and "Information Options".

**Page Layout Options:**

- ☐ Print names beginning with a new letter on a separate page
- ☒ Print page numbers
- ☒ Print phone numbers

**Information Options:**

**Member Printing Options**

- ☒ Do Not Print Members
- ☐ Print Only Adults
- ☐ Print Only Children
- ☐ Print Children & Adults

**Member Information Options**

- ☒ Do Not Print Age or Grade
- ☐ Print Member Grade
- ☐ Print Member Age

At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

We usually put the code for creating a page and doing things on the screen in the **AfterLoadReport** event. This event fires before the user gets onto the screen but after the report has been created.

```
procedure AfterLoadReport;
begin
    NewScreenPage;
    AddHeading('Page Layout Options:',PageLabel);
    AddCheckBox('Print names beginning with a new letter '+'
        'on a separate page',SepPageCByes);
    AddCheckBox(' Print page numbers',PageNumCByes);
    AddCheckBox(' Print phone numbers',FonCByes);
    AddBlank;
    AddHeading('Information Options:',PageLabel);
    AddRadioButtonGroup(' Member Printing Options ',
        'Do Not Print Members','+
        'Print Only Adults','+
        'Print Only Children','+
        'Print Children & Adults',MemGroup);
    AddBlank;
    AddRadioButtonGroup(' Member Information Options ',
        'Do Not Print Age or Grade','+
        'Print Member Grade','+
        'Print Member Age',AgeGradeGrp);
    AddBlank;
    MemGroup.ItemIndex:=3; //Set 'Children & Adults' as
                           //default, counting 0-3.
    AgeGradeGrp.ItemIndex:=2; //Set Member Age as default
end;
```

## Saving/Loading the Screen Information

Now that we have the information in, we would like to save it so that the next time we run this report we have the values the same as we did before. To do this we would like to write to the INI file for this report. We usually save the values just before the report is finished in the **BeforeDestroyReport** event. In this case we have five parameters to save.

But, we only want to save the values if they have valid values. If the program stops due to an error before the old values were read in, we do not want to save the values because we would be writing over good values with blanks. So, we add a variable called 'FieldsRead'. We initialize it to 'False' in the **AfterCreateReport**, check it before we write, and set it to True after we read in the values.

```
Var
    FieldsRead:      boolean;

procedure AfterCreateReport;
begin
    FieldsRead:=false;
end;

procedure BeforeDestroyReport;
var
    IniFile: TIniFile;
begin
    if FieldsRead then
    begin
        // ACCESS INI FILE save the values
        IniFile:=TIniFile.Create(trim(DMReport.MainRepName)+
            '.ini');
        try
            IniFile.WriteInteger('LastInfo', 'MemGroup',
                MemGroup.ItemIndex);
            IniFile.WriteInteger('LastInfo', 'AgeGradeGrp',
                AgeGradeGrp.ItemIndex);
            IniFile.WriteBool('LastInfo', 'ChkPgNum',
                PageNumCByes.Checked);
            IniFile.WriteBool('LastInfo', 'ChkSepPg',
                SepPageCByes.Checked);
            IniFile.WriteBool('LastInfo', 'FonYes',
                FonCByes.Checked);
        finally
            IniFile.Free;
            IniFile:=nil;
        end;
    end;
end;
```

After running this report, the INI file will have a [LastInfo] section:

```
[LastInfo]
MemGroup=3
AgeGradeGrp=2
ChkPgNum=Yes
ChkSepPg=No
FonYes=Yes
```

The reading of the INI file goes at the end of the **AfterLoadReport** event. We put it there because we want the screen object to already exist. That way we don't need to keep temporary variables around to hold the items stored in the INI file. We load them directly into the screen objects.

```
// READ INI FILE  set the initial values
IniFile:=TIniFile.Create(trim(DMReport.MainRepName)+
'.ini');
try
  MemGroup.ItemIndex:=IniFile.ReadInteger('LastInfo',
'MemGroup',0);
  AgeGradeGrp.ItemIndex:=IniFile.ReadInteger('LastInfo',
'AgeGradeGrp',0);
  PageNumCByes.Checked:=IniFile.ReadBool('LastInfo',
'ChkPgNum',true);
  SepPageCByes.Checked:=IniFile.ReadBool('LastInfo',
'ChkSepPg',true);
  FonCByes.Checked:=IniFile.ReadBool('LastInfo',
'FonYes',true);
  FieldsRead:=true;
finally
  IniFile.Free;
  IniFile:=nil;
end;
```

## Communicating with the Report

Now that you have the information about what the user wants, you must be able to tell the report what you have learned. To do this we have added two routines:

```
Procedure PDSSetParam(I: integer; S: string);
Function PDSGetParam(I: integer): string;
```

With these two routines, we can send information to the report and the report can even send information back to the script. The 'I' is an integer that allows you to have any number of parameters. 'I' can be 0 to 32000. Each value of 'I' is a different parameter. Each parameter is a string. These routines are available in the script and in the embedded code. So, in the script you could have a line that read:

```
begin
  PDSSetParam(0,EditName.Text);
End;
```

Where 'EditName' is an edit object that we have showing on the screen. And in the embedded code we could have a **GetText** event of a label that reads:

```
begin
  Text:=PDSGetParam(0);
End;
```

The report will print the value that we entered on our screen. We can pass any type of value between the script and the report but we have to convert it to a string before we send it. Most of the time we can use it as a string in the embedded code but sometimes we need it as the type it was originally and we have to convert it back in the embedded code. This is often true for dates when we want to use the date in some sort of comparison.



The most common place to put the setting of the parameter in the script is in the **AfterSelectReport** event. We put it in there because we know when we get to that point the user cannot change anything on the screen. Sometimes there are many parameters to pass and so we create a procedure to do that.

```
procedure SetParams;
begin
    // PUT THE ITEMS INTO PARAMETER FOR THE REPORT
    PDSSetParam(0,UpperCase(OwnerName));
    if SepPageCByes.Checked
    then PDSSetParam(1,'Yes')
    else PDSSetParam(1,'No');
    if PageNumCByes.Checked
    then PDSSetParam(2,'Yes')
    else PDSSetParam(2,'No');
    if FonCByes.Checked
    then PDSSetParam(4,'Yes')
    else PDSSetParam(4,'No');
    PDSSetParam(5,MemGroup.ItemIndex);
    PDSSetParam(6,AgeGradeGrp.ItemIndex);
end;

procedure AfterSelectReport;
begin
    SetParams;
end;
```

We also need to add a call to our new **SetParams** routine at the bottom of the **AfterLoadReport** event after the code to load the INI file. We need this call so that the preview report will work inside the Advance Report Writer editor.

In this case we are setting six parameters. One is a string that we capitalize before we set it. Three are Boolean values that are 'Yes' or 'No' that we are converting to strings. Two are integers. Now we need to include in the report the embedded code to take these parameters and use them.

On the **BeforePrint** event of the page number, at the top, enter:

```
var
    s: string;
begin
    s:=PDSGetParam(2);
    if s='Yes'
    then SystemVariable1.Visible:=True
    else SystemVariable1.Visible:=False;
end;
```

On the **BeforePrint** event of the phone number list, enter:

```
var
    s: string;
begin
    s:=PDSGetParam(4);
    if s='Yes'
    then DBMemor1.Visible:=True
    else DBMemor1.Visible:=False;
end;
```

On the **BeforePrint** event of the member grade field, enter:

```
procedure DBText5OnPrint;
var
  i: integer;
begin
  i:=StrToInt(PDSGetParam(6));
  if i=1
    then DBText5.Visible:=True
    else DBText5.Visible:=False;
end;
```

On the **BeforePrint** event of the member age field, enter:

```
var
  i: integer;
begin
  i:=StrToInt(PDSGetParam(6));
  if i=2
    then DBText3.Visible:=True
    else DBText3.Visible:=False;
end;
```

On the **BeforePrint** event of the member detail band, enter:

```
var
  i: integer;
  i2: integer;
begin
  i:=StrToInt(PDSGetParam(5));
  i2:=Mem['Mem Type Number'];
  if i=0
    then Detail.Visible:=False
    else
      begin
        if ((i=1) and (i2=4)) or ((i=2) and (i2<>4))
          then Detail.Visible:=False
          else Detail.Visible:=True;
        end;
      end;
end;
```

To add the ability to start each letter on a new page, we have to add a group that triggers when the first letter changes. To do that, we add a new group that is connected to **Label1** and has 'start new page' checked. We also enter the following code for its **Get Break Value** event:

```
var
  s: string;
begin
  s:=PDSGetParam(1);
  if s='Yes'
    then BreakValue:=Copy(Fam['Fam Name'],1,1)
    else BreakValue:='';
end;
```

Note that the capitalization must match or we would want to use **UpperCase(PDSGetParam(1))** to force the value to be in all caps.

In the code above we put the parameters in a variable and used that variable for comparisons. We could have compared directly with the values (i.e. if PDSGetParam(1)='Yes') but we have found that this sometimes does not work (we have not yet found the reason why) and so we always assign it to a variable and then do the comparison.

## Other Objects

There are other objects that can be placed on a screen, such as buttons and grids. Examples of these can be found in the Member Involvement reports.

Just like in the embedded code, objects in the script can have events, too. For example, if you add a button to a screen, you would like to add an event that fires when the button is clicked on. So, we can create a button:

```
Procedure MyBtnOnClick(Sender: Tobject);
Begin
  ShowMessage('Button clicked');
end;

Procedure AfterReportLoad;
var
  MyBtn: TButton;
begin
  NewScreenPage;
  MyBtn:=TButton.Create(CurForm);
  MyBtn.Caption:='My Button';
  MyBtn.Top:=10;
  MyBtn.Left:=10;
  MyBtn.Width:=30;
  MyBtn.Height:=10;
  MyBtn.Parent:=CurPage;
  AttachEvent(MyBtn, 'OnClick', 'MyBtnOnClick',false);
end;
```

The **AttachEvent** procedure connects an event to the procedure you want to run for a given object.

When we create an object that is not one in the ScrnIO, we must set a number of its attributes. The main attribute that must be set is the *Parent*. This tells the program where the object is displayed. In most cases, this is in the new page that we created called **CurPage**.

## Positioning objects

If you create an object you also have to set the top, left, width and height of the object on the screen. In the ScrnIO, we have declared some variables that we use to set these. We initialize them in the **CreateNewPage** or in the script itself before we create a new page to be a correct size for the current screen size. You can use these to, if you would like. The variables are:

FontH	The height of the font.
LabelT	The top of the next label object. Every time an object in the ScrnIO is put on the screen, we move this down by the amount in LabelN. If you use this, you should increment it by LabelN (i.e. <code>LabelT:=LabelT+LabelN;</code> ).
LabelH	The height of a standard label object.
LabelL	The left position of the label object.
LabelN	The increment to go to the next line down. Usually this is the same as the EditN.
EditT	The top of the next edit object. Every time an object in the ScrnIO is put on the screen, we move this down by the amount in EditN. If you use this, you should increment it by EditN (i.e. <code>EditT:=EditT+EditN;</code> ).
EditH	The height of a standard edit object.
EditL	The left position of the edit object.
EditW	The width of the edit object.
EditN	The increment to go to the next line down. Usually this is the same as the LabelN.

There are times when you need to change these values in order to control where the object is placed. You can see examples of this in the Member Involvement reports. In many of them we want to have two columns of fields.

You can position objects without using these variables. The benefit of using these variables is that they are adjusted based on the user's screen size. If you position the objects some other way, make sure you test the screen at different sizes.

---

## 8: Special Sorts

### Overview

Special sorts are another place where scripts are necessary. There are a number of different ways to get the information out of the program in a special order.

#### SelectTbl

The first and easiest way is to setup the **SelectTbl**. The SelectTbl is a table that is built into the program for doing sorts using SQL (Structured Query Language). The program knows that if it finds the SelectTbl set to active, it should use it as the order of the report.

For example, if we want to have a member report sorted by the date the family registered we can add this script to the member report.

```
procedure AfterSelectReport;
begin
    DM.SelectTbl.Sql.Text:=
        'Select '+
        '  Mem.MemRecNum, '+
        '  Fam.DateRegistered '+
        'from '+
        '  Mem, Fam '+
        'where '+
        '  Fam.FamRecNum=Mem.FamRecNum '+
        'order by '+
        '  DateRegistered';
    DM.SelectTbl.Active:=true;
end;
```

We put the SQL statements on multiple lines so that it is more readable. It could have been on one line. Also, we put a space at the end of each line so that it will not butt right up against the next line when it's pulled together.

In SQL, we have to include a field that links this table with the main table of the type of report we are writing for. If we are in a family report, we need to have the field FamRecNum. If we are in a member report, we need to have the field MemRecNum. This field is used by the program to connect with all the other tables. The program will get the first record of the SelectTbl and use this field to find the correct family record (or member record) and once that record is found, all the other tables that are linked to it.

We can also add conditions to the WHERE clause of the SQL, which cause the report to filter the families or members before printing. This would be in those cases where you want to filter the data without the user having to enter a selection.

#### MailTbl

A second way of doing a special sort in the Church Office, Formation Office and School Office is by using the **MailTbl**. The Mail table is a special table that works like the SelectTbl except it is not based on SQL. It is actually a table created in the user's temporary folder.

A record is added to this table for every family (or member) that is to be printed. It is used for zip code sorts, e-mail sorts, courtesy copy sorts, billing address sorts and any family or member sort that does not fit as SQL. It can have more than one entry on a family. The easiest way to use the MailTbl is to add two lines to the default section of UDR:

```
OrderMailTbl=1
ForceMailTbl=1
```

This will cause the OrderMailTbl event to ‘fire’ every time the program is ready to add an item to the MailTbl. You can then put in your own key value by setting

```
PDSMDData.MailTblKey.AsString
```

to a value or calculation that you want. When this event fires, all of the tables are setup and ready to be used, including FamTbl and MemTbl.

An example of this can be found in the Church Office financial reports. The **List in Order by Amount** report uses this method to do the sort. The script for this is:

```
procedure AfterLoadReport;
begin
  RepDlg.SortCB.Items.Clear;
  RepDlg.SortCB.Items.Add('Grand Total Paid -
Ascending'#9'0');
  RepDlg.SortCB.Items.Add('Grand Total Paid -
Descending'#9'1');
  RepDlg.SortCB.ItemIndex:=0;
end;

procedure OrderMailTbl;
var
  s: string;
  r: integer;
begin
  // Recalculate grand total based on family
  // and member info
  r:=PDSMDData.MailTblSepStmtRec.AsInteger;
  PDSDMCalc.MainSepStmtRec:=r;
  PDSDMCalc.BuildFundTbl;
  PDSDMCalc.BuildFundInfoTbl;

  if RepDlg.SortCB.ItemIndex=1
  then s:=FloatToStr(1000000000.00-
    PDSDMCalc.FundTotalsTblGrandTotalTotalPaid.AsString)
  else s:=FloatToStr(PDSDMCalc.
    FundTotalsTblGrandTotalTotalPaid.AsString);
  s:=copy(' '+s,length(s)+1,10); // 10 spaces
  PDSMDData.MailTblKey.AsString:=s;
  PDSDMCalc.MainSepStmtRec:=0;
end;
```

This report resets the list of ‘Sort By’ so that it includes two items. The [Default] section of the UDR for this report is:

```
[Default]
CanUseSepStmt=1
ForceMailTbl=1
OrderMailTbl=1
PageStyle=360
```

This tells the program that it can use separate statements, force the mail table to be built and used for sorting, and to try and call for script routines for each entry in the mail table. It also sets the default page style to be compressed print.

## Memory Tables

In some reports, there is a need to store information temporarily. Sometimes we can store them in variables. Sometimes we need more information than can fit in a variable. In these cases we can use a Memory Table. A Memory Table is a database table that resides in memory. It is not stored on disk (although there is a method to save and load memory tables if needed).

Lets start with a real world example of creating a memory table. In our financial statement, we want to include a note that includes the family's current rate and terms and the date of the last donation. Now, we can easily make an Advanced Report Writer report that does this, but it will calculate the values in a sub-report or an embedded list. This makes it hard to have the values plugged in exactly where we want them to be.

What we really want is to extend the report to have some new fields. This is where memory tables come in. If done properly, a memory table can be accessed just like any other data field (i.e. family name). They can be used in any kind of report and can even be used in selections in the reports that declare them. Lets begin.

### Declare the Variables

We first have to declare the variables that we will need. In this example, we are going to call the table **NewField** table. To implement a table, you usually need three variables: the table itself, the datasource, and the pipeline. So, at the top of the script we need:

```
var
  NewFieldTbl:    TKbmMemTable;
  NewFieldDS:    TDataSource;
  NewFieldPL:    TppDBPipeline;
```

The table is where we tell the program what the fields are and how the database is ordered and indexed. The datasource is a special connection to make connecting the database to the screen fields easier. The pipeline is the connection between the datasource and the report builder.

### Initialize the Variables

The next thing that we need to do is initialize these variables as soon as possible. That means in the **AfterCreateReport** procedure. We set these variables to *nil*, which means they have nothing in them.

```
procedure AfterCreateReport;
begin
  NewFieldTbl:=nil;
  NewFieldDS:=nil;
  NewFieldPL:=nil;
end;
```

We initialize them because if we don't, and the report stops early because of an error, we get an error message that confuses the debugging.

## Assign the Variables

Now that the variables are initialized, we can actually set them up with the correct values.

```
procedure CreateExternalData;
begin
    // CREATE A NEW MEMORY TABLE
    NewFieldTbl:=TKbmMemTable.Create(DMReport);
    with NewFieldTbl do
    begin
        // SETUP THE MEMORY TABLE
        FieldDefs.Clear; //We don't need this line, but
                        //it does not hurt either.

        // SETUP THE FIELD TO SORT BY
        FieldDefs.Add('Order',ftInteger, 0, False);

        // SETUP THE NEW FIELDS TO CALCULATE
        FieldDefs.Add('TermsAndRate',ftString, 40, False);
        FieldDefs.Add('Terms',ftString, 40, False);
        FieldDefs.Add('Rate',ftCurrency, 0, False);
        FieldDefs.Add('LastDate',ftDate, 0, false);

        // SETUP THE INDEX (NOT REALLY USED FOR THIS TABLE)
        IndexDefs.Add('', 'Order',MKSet(ixPrimary));
        NewFieldTbl.IndexFieldNames:='Order';
        CreateTable;
        Active:=true;
    end;

    // CREATE THE DATA SOURCE
    NewFieldDS:=TDataSource.Create(DMReport);
    NewFieldDS.DataSet:=NewFieldTbl;

    // CREATE THE PIPELINE
    NewFieldPL:=TppDBPipeline.Create(DMReport);
    NewFieldPL.Name:='NewFieldPL';
    NewFieldPL.UserName:='NewField'; // The name users see
    NewFieldPL.DataSource:=NewFieldDS;
    NewFieldPL.AutoCreateFields := true;

    // ADD IN THE FIRST AND ONLY RECORD
    NewFieldTbl.Append;
    NewFieldTbl.FieldByName('Order').AsInteger:=0;
    NewFieldTbl.FieldByName('TermsAndRate').AsString:='';
    NewFieldTbl.FieldByName('Terms').AsString:='';
    NewFieldTbl.FieldByName('Rate').AsCurrency:=0;
    NewFieldTbl.Post;
    UseCallAfterCalc:=True; // Tell the program to call
                            // CallAfterCalc after moving
                            // from one family to the next.
end;
```



Just to keep things tidy, let's also put in the code needed to get rid of the table when we are done. This code goes in the **FreeExternalData** event. We only free the items if they have been assigned (i.e. not equal to nil).

```
procedure FreeExternalData;
begin
  if assigned(NewFieldTbl)
  then NewFieldTbl.Free;
  if assigned(NewFieldDS)
  then NewFieldDS.Free;
  if assigned(NewFieldPL)
  then NewFieldPL.Free;
end;
```

## Attach the Tables

Now, since we added a new table and we want that table to be accessible to the reports and selections, we need to add it to our list of tables. This can be done in an event called **AttachNewTables**. If we had more than one, we would add all of them in this event.

```
procedure AttachNewTables;
begin
  AddTable(NewFieldTbl, NewFieldDS, 'NewField',
           'Order', 'Fam', 'FamRecNum', '');
end;
```

The **AddTable** procedure takes seven parameters.

- The first is the table we want to add.
- The second is the datasource for that table.
- The third is the name we want to show to the user.
- The fourth is the indexed field we want to use.
- The fifth is the table that this table is linked to.
- The sixth is the field in that table that must match the value of the third parameter (the join).
- The seventh parameter is always blank.

## Build the Table

Now we can define what should be done after every access to a different family record.

```
// CALCULATE THE VALUE OF THE NEW FIELDS. THIS
// IS CALLED EVERY TIME WE GET A NEW FAMILY.
procedure CallAfterCalc;
var
  s, s1: string;
  i: integer;
  d, d1: date;
```

(code continued on next page)

(continued from previous page)

```
begin
  // LOOP THROUGH THE CALCFUND TABLE AND THE
  // CACLFUNDHIST TO FIND THE LAST PAYMENT AND
  // THE CALCFUNDACT TO FIND THE FIRST ACTIVITY
  // THAT HAS AN RECURRING CHARGE ASSOICATED
  // WITH IT. WE THEN STORE IT IN THE NEWFIELD
  // TABLE.
  s:='';
  d:=0;
  with PDSDMCalc.CalcFundTbl do
    begin
      First;
      while not eof do
        begin
          // Determine the last date
          with PDSDMCalc.CalcFundHistTbl do
            begin
              First;
              while not eof do
                begin
                  if PDSDMCalc.CalcFundHistTblActFunc.
                     AsInteger<9 then // payment type;
                    begin
                      d1:=PDSDMCalc.
                        CalcFundHistTblFEDate.
                          AsDateTime;
                      if d1>d
                        then d:=d1;
                    end;
                  Next;
                end;
              end;
            // Get the last recurring description
            if s='' then
              with PDSDMCalc.CalcFundActTbl do
                begin
                  First;
                  while (s='') and not eof do
                    begin
                      s1:=trim(PDSDMCalc.
                        CalcFundActTblRecurDesc.
                          AsString);
                      if s1<>''
                        then s:=s1;
                    end;
                  Next;
                end;
              end;
            Next;
          end;
          First; // Move back to the first record just
                // in case anything depends on that
        end;
      end;
    end;
```

(code continued on next page)

(continued from previous page)

```
// THERE IS ONLY A SINGLE RECORD.  SO EDIT IT AND
// PUT IN THE NEW VALUES
NewFieldTbl.Edit;
NewFieldTbl.FieldName('Order').AsInteger:=
    PDSMDMData.FamTblFamRecNum.AsInteger;
NewFieldTbl.FieldName('TermsAndRate').AsString:=s;
i:=pos(' ',s);
if i<>0 then
    begin
        NewFieldTbl.FieldName('Terms').AsString:=
            copy(s,i+1,40);
        s:=CleanAmount(copy(s,1,i));
        NewFieldTbl.FieldName('Rate').AsCurrency:=
            StrToCurr(s);
    end
else
    begin
        NewFieldTbl.FieldName('Terms').AsString:=s;
        NewFieldTbl.FieldName('Rate').AsCurrency:=0;
    end;
if d=0
    then d:=now;
NewFieldTbl.FieldName('LastDate').AsDateTime:=d;
NewFieldTbl.Post;
end;
```

In the Church office programs we have built in some optimizations to try and speed up reports. Once a family is read in it is not recalculated. Well, in this case we want to recalculate it after we have connected it. So we need to trick the computer into thinking it has not yet seen the first record before. We do this in the **PrintDefaultReport** event.

```
procedure PrintDefaultReport;
begin
    // Force the program to recalculate just before
    // the report starts to print. If we don't do
    // that the first record is wrong.
    PDSDMCalc.MainCurCalcFamRec:=0;
    PDSMDMData.FamTbl.First;
end;
```

## Database Tables

There are occasions where there is either too much information, or we want to keep the information around; where we want to create and use an actual database table instead of a memory table. We also may want to create a table that is not a new table but just a connection to an existing table.

For example, in the Member Involvement reports, we create a table to access to the ministries and the ministry statuses. We then use these tables in a set of grids to give the user the ability to select ministries to print. In this case we don't need to have a pipeline since the table we are creating is just for use in the layout screen, not in the report.

```
// Build Table
MinTbl:=TwwTable.Create(CurForm);
MinTbl.Active:=false;
MinTbl.TableType:=ttParadox;
MinTbl.DatabaseName:=PDSDMData.FamTbl.DatabaseName;
MinTbl.TableName:='MinType.db';
MinTbl.IndexFieldNames:='Description';
MinTbl.FieldDefs.Update;
MinSelFld:=TBooleanField.Create(CurForm);
MinSelFld.FieldName:='Selected';
MinSelFld.Calculated:=true;
MinSelFld.Dataset:=MinTbl;
MinSelFld:=TStringField.Create(CurForm);
MinSelFld.FieldName:='Description';
MinSelFld.FieldKind:=fkData;
MinSelFld.Dataset:=MinTbl;
MinSelFld:=TIntegerField.Create(CurForm);
MinSelFld.FieldName:='MinDescRec';
MinSelFld.FieldKind:=fkData;
MinSelFld.Dataset:=MinTbl;
MinDS:=TwwDataSource.Create(CurForm);
MinDS.DataSet:=MinTbl;
MinTbl.Active:=true;
```

In this example, we create a database table (not a memory table) and we connect it to the 'MinType.db' file. We connect it to the same database that the family table is connected to, and we index it by description.

Then we create three field objects. One is attached to the Description; one is attached to the MinDescRec, which is a unique integer identifier for each record in the description; and one is a calculated Boolean field called 'Selected'.

Creating the fields makes it easier to use later in the grid, but since we actually don't need to use the field variables themselves, we used the MinSelFld three times instead of making three separate variables.

---

## 9: Exporting and Importing Data

### Overview

Built into reports is the ability to export fields. There are many times when we need to export information in a very precise way and the generic export is not enough. Scripting can be used to export any information and to create complicated formations.

### Export Events

When you define a report as being an export, the program provides 3 more events:

BeforeBuildExport: boolean	Called before the export is run. You can use this event in conjunction with the <b>PrintDefaultReport</b> to tell the program that this is not a 'normal' export. That means we will be handling all of the writing to the file. None of the built-in code for writing will be needed.
AfterBuildExport	Called after the internal export is run. If we are handling the export then it is called right after the <b>BeforeBuildExport</b> .
FinishBuildExport: Boolean	After the export, the program would usually bring up a dialog that says the export is finished. Sometimes we might bring up a different dialog or a summary screen. We return false (i.e. Result:=false;) if we don't want the default dialog box to show up.

### Simple Export

Using these events, we can create a simple export. We start by going to the family report screen and adding a new report. We choose 'Export' for the type of report. Now we go into the script editor and enter the following:

```
var
  f: TextFile;

function PrintDefaultReport: boolean;
begin
  result:=false;          // WE WILL CREATE THE FILE
                           // IN AfterBuildExport
end;

function BeforeBuildExport: boolean;
begin
  result:=false;          // WE WILL CREATE THE FILE
                           // IN AfterBuildExport
end;
```

```

procedure AfterBuildExport;
begin
  // BUILD THE EXPORT
  AssignFile(F,RepDlg.ExpFilename.Text);
  Rewrite(f);
  try
    with PDSMDData.FamTbl do
      begin
        First;
        while not EOF do
          begin
            write(f,PDSMDData.FamTbl['NameFormat1']);
            writeln(f,'');
            Next;
          end;
        end;
      finally
        CloseFile(f);
      end;
    end;
  end;
end;

```

This creates a TextFile, which is a variable that is a file on the disk. It assigns the name that we have on the export layout screen, and it creates the file (the file is erased if it already exists). Then it fills the file with family names (using Family Name Format 1), with each family on a new line. Finally, it closes the file. We could have included any fields or done any type of calculation to obtain the information to export.

## Improving the Export

This works fine, except that it brings up a dialog box as it goes through families, which does not give us any idea of where we are in the run. If we press cancel, it does not cancel until it is finished. To solve these problems we would add the following code just before the Next statement, which sets the caption of the export label that is on the dialog box, and keeps watching for the label to change to 'Stopping...'.

```

ExportLabel.Caption:='Family: '+
                    PDSMDData.FamTblName.AsString;
Application.ProcessMessages;
if ExportLabel.Caption='Stopping...'
then exit;

```

A few last, minor things. This export still has the 'Modify fields to export' button and the pull down for the list of formats. We would like to hide these. We can hide them using the following code:

```

procedure AfterLoadReport;
begin
  // MAKE THE REPORT DIALOG LOOK BETTER
  RepDlg.Label33.Caption:='My Export: ';
  RepDlg.GroupBox15.Visible:=false; // hide button
  RepDlg.Label35.Visible:=false;    // hide label
  RepDlg.ExpTypeCB.Visible:=false;  // hide pulldown
end;

```

## Other File Formats

Sometimes you need to output control characters into the export files. You can do this by using either the 'chr' function or a '#' in front of the value. For example, if we needed to put in a Control-G, which is an ASCII '9', we could enter a `chr(9)`, or we could use `#9` which is a literal string of one character having that value.

Sometimes we need commas between fields. In these cases we use a quote, a comma and another quote (i.e. ','). If we need to have a quote character we can use two quotes together inside the quotes. So to write the family name field with quotes around a name we would have:

```
write(f, ''''+PDSDMData.FamTbl['NameFormat1']+''');
```

or you can use 'chr(39)', which would be:

```
write(f, chr(39)+PDSDMData.FamTbl['NameFormat1']+chr(39));
```

There are some times when we need a fixed-width format. That means we want the fields to be in fixed positions in the file. To do this we have to create some 'helper' routines. These routines will take our fields and output the information in the fixed format, padding them to the proper width where needed. An example of a 'helper' routine would be one that takes a string and outputs it with a fixed width of N bytes:

```
procedure ExportStr(s: string; n: integer);
var
  i: integer;
begin
  s:=copy(s,1,n);
  while length(s)<n do
    s:=s+' ';
  for i:=1 to n do
    Write(f,s[i]);
end;
```

Another helper might output an integer and right justify it to fit in the field:

```
procedure ExportInt(i: integer; n: integer);
var
  s: string;
begin
  s:=IntToStr(i);
  s:=copy(s,1,n);
  while (length(s)<n) do
    s:=' '+s;
  for i:=1 to n do
    Write(f,s[i]);
end;
```

For fixed width fields, it usually is better to work with a 'File of Bytes' instead of a TextFile. TextFiles assume the data is broken down by lines and have a 32000 byte limit to the length of a line. A file of bytes doesn't have that limit.

Here is an example that uses the helper routine and a file of bytes. In this example we have 2 fields: the name and the number of members. The name is 40 bytes wide and the number is 10 bytes wide and is right justified.

```
var
  f: File of byte;

function PrintDefaultReport: boolean;
begin
  result:=false;      // WE WILL CREATE THE FILE
                      // IN AfterBuildExport
end;

function BeforeBuildExport: boolean;
begin
  result:=false;      // WE WILL CREATE THE FILE
                      // IN AfterBuildExport
end;

procedure ExportStr(s: string; n: integer);
var
  i: integer;
begin
  s:=copy(s,1,n);
  while length(s)<n do
    s:=s+' ';
  for i:=1 to n do
    Write(f,s[i]);
end;

procedure ExportInt(i: integer; n: integer);
var
  s: string;
begin
  s:=IntToStr(i);
  s:=copy(s,1,n);
  while (length(s)<n) do
    s:=''+s;
  for i:=1 to n do
    Write(f,s[i]);
end;
```



```

procedure AfterBuildExport;
begin
  // BUILD THE EXPORT
  AssignFile(F,RepDlg.ExpFilename.Text);
  Rewrite(f);
  try
    with PDSMDData.FamTbl do
      begin
        First;
        while not EOF do
          begin
            ExportStr(PDSMDData.
                      FamTbl['NameFormat1'],40);
            ExportInt(PDSMDData.
                     FamTbl['NumMem'],10);
            Next;
          end;
        end;
      finally
        CloseFile(f);
      end;
    end;
end;

```

## Adding a Summary

Many times, when we have finished the export, we would like to tell the user what was exported. We wish to display a summary and allow the user to print that summary if they wish. To create and display the summary, we can use the following code in our script.

```

procedure BuildSummary;
var
  ErrorRepDlg: TErrorRepDlg;
begin
  // SHOW NEXT FORM
  ErrorRepDlg:=TErrorRepDlg.Create(Self);
  try
    PDSFixupDlg(ErrorRepDlg);
    ErrorRepDlg.RTFText.ReadOnly:=false;
    ErrorRepDlg.DetailBtn.Visible:=False;
    ErrorRepDlg.SummaryBtn.Caption:='&Print';
    ErrorRepDlg.RTFText.Clear;
    MainRTF:=ErrorRepDlg.RTFText;
    MainFontName:='New Courier';
    ClearTabs; // clear tab stops
    SetTab(3*1440,0); // tab stop at 3 inches
    SetTab(4*1440,0); // tab stop at 4 inches
    // Here is what we want to say
    println('Summary');
    println('');
    println(DateToStr(Now));
    println('');
    println('');
  end;
end;

```

```

// GET THE DIALOG READY
ErrorRepDlg.ReportTitle:='Export Data For XYZ';
ErrorRepDlg.ReportSettingsFileName:=trim(DMReport.
    RepTblFileName.AsString)+'.ini';
ErrorRepDlg.ProcExpLabel.Caption:='Export Completed';
ErrorRepDlg.PrintExpLabel.Caption:='';
ErrorRepDlg.RTFText.ReadOnly:=true;
ErrorRepDlg.RTFText.CPPosition:=0;
PDSShowDlg(ErrorRepDlg);
finally
    PDSFreeDlg(ErrorRepDlg);
end;
end;

```

Then, at the end of the **AfterBuildExport** routine we would put in:

```
BuildSummary;
```

We would also add a **FinishBuildExport** event to tell the program not to show the last dialog box.

```

function FinishBuildExport: boolean;
begin
    Result:=false;    // We have already showed the
                      // summary so we are finished.
                      // Don't show the last dialog.
end;

```

In the Church Office program, we have a very complicated export for envelope companies. Please make a copy of this and look at the script for it. Looking at other reports can help you understand how things work.

## Exporting to OLE

We can export to a file, but we can also export directly to an OLE server such as Microsoft Excel or Microsoft Word. Here is a very small routine that opens Excel, creates a spreadsheet, and puts text in cell position '5,1', which is spreadsheet cell A5, with a font size of 20.

```

function PrintDefaultReport: boolean;
begin
    result:=false;    // WE WILL CREATE FILE IN
                      // AfterBuildExport
end;

function BeforeBuildExport: boolean;
begin
    result:=false;    // WE WILL CREATE FILE IN
                      // AfterBuildExport
end;

```

```

procedure AfterBuildExport;
var
  v: OLEVariant;
begin
  v:=CreateOleObject('Excel.Application');
  v.Visible:=true;
  v.WorkBooks.Add;
  v.ActiveSheet.Cells(5,1).Font.Size:=20;
  v.ActiveSheet.Cells(5,1):='PDS Ledger Data';
end;

```

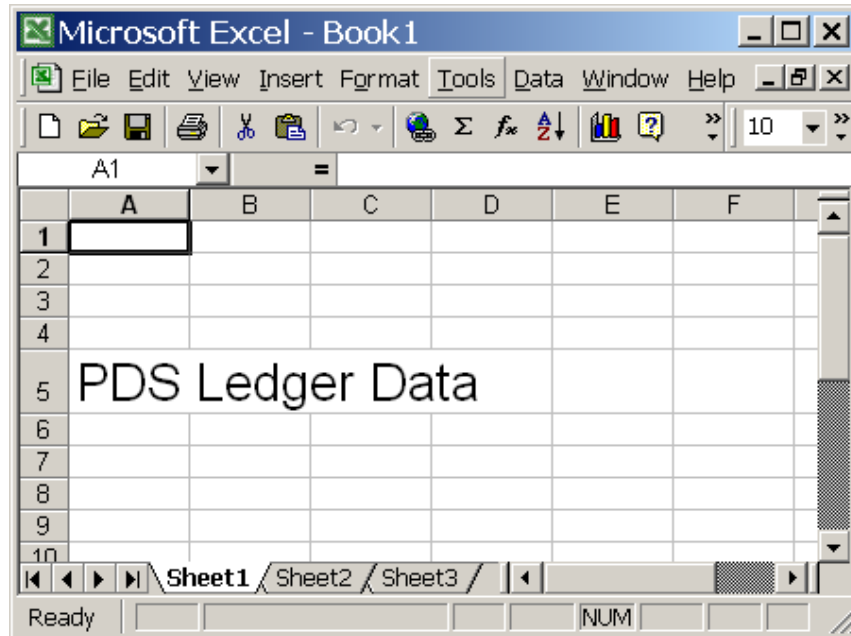


Fig. 9-1

## Importing Data

Just as there are cases for exporting data, there are cases where we need to import data. The scripting can handle that as well. To create an import, we start with what we learned in the exports. We create an export report and instead of writing the data to a disk, we read from it. All of the things we covered in exports also apply to imports. We might be dealing with straight text or it might be fixed width data. We want to show the user progress and they should be able to escape out. We might have summary at the end.

To read information from a file, we use commands that are similar to those for writing to a file. We assign the file variable to the file name. Instead of doing a 'rewrite', which we used in the export to delete the file and create a new file, we do a 'reset', which just opens the file. We do have to do some checking to make sure the file is there.

In the export we looped through the families. In the import we will be looping through the text file (or other type) and use 'readln' to read the next line into the string. We can then do something with that information. For example let's assume the disk contains the families' ID numbers, a comma, and a number to be put in the families' geographical area. We would need to read in the line, break it into the 2 parts, look up the family by id, and put in the new value.

```

var
  f: TextFile;

function PrintDefaultReport: boolean;
begin
  result:=false;      // WE WILL CREATE THE FILE
                      // IN AfterBuildExport
end;

function BeforeBuildExport: boolean;
begin
  result:=false;      // WE WILL CREATE THE FILE
                      // IN AfterBuildExport
end;

procedure AfterBuildExport;
label OpenTheFile;
var
  s: string;
  id,
  area: string;
  I: integer;
begin
  // BUILD THE EXPORT
OpenTheFile:
  AssignFile(f,'a:\temp.txt');

  try
    Reset(f);
  except
    If PDSMessageDlg('Cannot open the file.',mtError,
                     mkSet(mbCancel,mbRetry),0)=mrRetry
    then goto OpenTheFile
    else exit;
  end;
  Try
    While not eof(f) do
      begin
        try
          Readln(f,s);
          I:=pos(',',s);
        except
          I:=0;
        end;
        If I<>0 then
          Begin
            Id:=copy(s,1,I-1);
            // THE ID NUMBER IS IN ParKey
            // AND IS RIGHT JUSTIFIED.
            Id:=copy('          '+id,length(id)+1,10);
            Area:=Copy(s,I+1,length(s));

```

```

        If PDSMDData.FamTbl.Locate('ParKey',id,
            mkset(loCaseInsensitive)) then
            begin
                PDSMDData.FamTbl.edit;
                PDSMDData.FamTblAreaNumber.AsString:=area;
                PDSMDData.FamTbl.post;
            end;
        End;
    end;
finally
    CloseFile(f);
end;
end;

```

Importing requires a very extensive knowledge of the data. Simple items can be brought in. More complicated items may need a programmer to figure out how to access it.

## Data Manipulation

In the same way we could import the data, we could have manipulated it without pulling information in from a file. For example, we could use these techniques to change the name of all the families if the user has entered them wrong and other similar processes.

Many advanced things can be done with scripts. For example, here is a small report that adds a financial entry to family ID number 1 for \$999.54, to fund 1 Offering. It uses a procedure in **AdvRep.pas**.

```

uses
    AdvRep;

procedure AfterCreateReport;
begin
    if AddHistEntry(false,
        '1',
        'ParKey',
        '1',
        StrToDate('01/28/03'),
        999.54,
        'Offering',
        '1234',
        0)
    then showmessage('Success')
    else showmessage('Failed');
end;

```

AdvRep has a number of other useful routines you may want to look at.

## Notes

---

# 10: Appendix

## Overview

This chapter contains listings and tables related to Advanced Report Writer reports.

### UDR Options

#### [Default]

For fund reports, you can use:

√ FundLimit = xxx – this is the number of funds the report can print.

In the Ledger and Ledger/Payroll you can also use:

- CRasNeg= 1 - make credits as negative debits.
- TrnType= 1NNNNNNNNNN - which transaction types are included as a default from the transaction selection tab
- ForceTrnType= 1\*\*\*\*N\*\*\*\*
- CoaType= YYNNNNYNYYYY - which coa types are included as a default from the COA selection tab
- ForceCoaType= YYY\*\*\*\*N\*\*\*\*
- IsBalanceSheet= 1 - is the report a special type
- IsIncomeStmt= 1
- IsGeneralLedger= 1
- IsCashFlow= 1
- JustBeginningBalances= 1 – use only the Beginning Balance values

In Church, Formation and School:

- FamType= YNNYYN - Y-Active, N Inactive, N-Loose, Y-Fund, Y-Student, N-Teacher
- MemType= YN - Y-Active, N-Inactive
- CanUseEMail= 1
- CanSkipEMail= 1
- CanLog= 1
- CanUseCC= 1
- CanUseBulk= 1
- DisableScaling=1 - Don't reposition items to take up available paper size
- UseBap= 1
- ForceMailTbl= 1 - Force the mail table to be built
- OrderMailTbl= 1
- UseBap= 1 - Use the place of baptism as the address, city/state and zip

The following are fund related:

- IncludeFundIncludeHist= 1
- UseSepRecapDate= 1
- RelaxDates= 1
- FuncType= 1 - What functions are available on the fund selection screen
  - 0-4 items: Don't include, use in month, group totals only, itemize
  - 1-2 items: Don't include, include group
  - 2-3 items: Don't Include, include group, Itemize
  - other-2 items: Don't include, include group and in months

In Church Office

- CanUseSepStmt= 1 - Members with separate stmt checked can be treated differently.
- CanUseSepMem= 1

## Formation Embedded Lists

These are the predefined imbedded lists for Formation Office Management

<List: All Pledges>	Total of all families for all pledges for selected funds.
<List: Comm>	Communion data for each member.
<List: Confirm>	Confirmation data for each member.
<List: Coupon Bottom>	Prints a return coupon at the bottom of the letter.
<List: Coupon Top>	Prints a return coupon at the top of the letter.
<List: Delinq>	Family's delinquency information.
<List: Fam Sched>	Class schedule for all students in family.
<List: Fund Month>	Fund name and total paid each month.
<List: Gifts>	Additional gifts totals.
<List: Grp List>	Fund groups with Due, Paid, 'Balance for stmt'.
<List: Grp List 2>	Fund groups with Due, 'Total Tax deduct.', balance
<List: Grp List 3>	Fund groups showing Amt Paid only.
<List: Hist List>	Activity history for the fund selected.
<List: Hour List>	Hours due, completed, balance; similar to GrpList.
<List: IRS Note>	Special tax note at the end of tax statements.
<List: Mem Amt>	Members of family and amount each has paid.
<List: Receipt History With Desc>	Activity history with date and description.
<List: Running Total>	Activity history with cumulative due/paid totals
<List: Simple Totals>	Total Pledged, Paid, Balance for all funds selected.

## Built-in Routines:

### String Operations:

s1+s2	string concatenation
Capitalize(s)	string
Copy(s,i1,i2)	string
Delete(s,i1,i2)	
Fixed(s)	string
FixedName(s)	string
FixedAddress(s)	string
FixedCity(s)	string
Insert(s1,s2,1)	
Length(s)	integer
LowerCase(s)	string
Pos(s1,s2)	integer
Trim(s)	string
TrimLeft(s)	string
TrimRight(s)	string
UpperCase(s)	string

### Simple Math Operations

i1+i2	add
i1-i2	subtract
i1*i2	multiply
i1/i2	divide
i1 div i2	integer division
i1 mod i2	remainder
-i1	unary minus



<b>Math Routines</b>	ArcTan(x)	extended
	Cos(x)	extended
	Cosh(x)	extended
	Cotan(x)	extended
	Exp(x)	extended
	Frac(x)	extended
	Int(x)	extended
	IntPower(x,i)	extended
	Ln(x)	extended
	Power(x1,x2)	extended
	Round(x)	integer
	Sin(x)	extended
	Sqr(x)	extended
	Sqrt(x)	extended
	Tan(x)	extended
	Tanh(x)	extended
	Trunc(x)	integer
<b>Date / Time Operations</b>	CurrentDate	TDate
	CurrentDateTime	TDateTime
	CurrentTime	TDateTime
	DayOfWeek(d)	integer
	DecodeDate(d,iy,im,id)	
	DecodeTime(t,ih,im,is,ims)	
	EncodeDate(iy,im,id)	TDate
	EncodeTime(ih,im,is,ims)	TDateTime
<b>Conversions</b>	Chr(i)	string
	CurrToStr(c)	string
	DateTimeToStr(dt)	string
	DateToStr(d)	string
	FloatToStr(f)	string
	IntToStr(i)	string
	RGB(ir,ig,ib)	integer
	StrToCur(s)	currency
	StrToDate(s)	Tdate
	StrToDateTime(s)	TDateTime
	StrToFloat(s)	extended
	StrToInt(s)	integer
	StrToIntDef(s,i)	integer
	StrToTime(s)	TDateTime
	TimeToStr(t)	string
<b>Formatting</b>	FormatCurr(s,c)	string
	FormatDateTime(s,dt)	string
	FormatFloat(s,f)	string
	FormatMaskText(s1,s2)	string
	FormatString(s1,s2)	string

<b>Boolean Operations</b>	b1 and b2	
	b1 or b2	
	b1 xor b2	exclusive or
	not b2	
<b>PDS built-in Routines</b>	MainOverflow	
	MainHeaderVisible	boolean
	MainHeaderSetVisible(b)	
	MainFooterVisible	boolean
	MainFooterSetVisible(b)	
<b>PDS Scripting Routines</b>	PDSGetCrossTabRowCount	integer
	PDSGetParam(i)	string
	PDSSetParams(i,s)	
	CallFunc(s)	string
	CallFundP1(s,i1)	string
<b>Member Type Values:</b>	CallFuncP2(s,i1,i2)	string
	0: Head	
	1: Spouse	
	2: Adult	
	3: Young Adult	
	4: Child	
	5: Other	
	Mem.['Mem Type Number'] in embedded code	
	PDSMDData.MemTbl['MemberNum'] in script	
	Fund Period Values:	
	0: Weekly	
	1: Weekly on Sunday	
	2: Weekly on Monday	
	3: Weekly on Tuesday	
	4: Weekly on Wednesday	
	5: Weekly on Thursday	
	6: Weekly on Friday	
	7: Weekly on Saturday	
	8: Monthly	
	9: BiMonthly – every 2 months	
	10: SemiMonthly – twice a month	
	11: Quarterly	
	12: Semiannually – twice a year	
	13: Annually	
	14: Special	
	CalcRate.['Calc Rate Terms Period'] in embedded code	
	PDSMDData.CalcRateTbl['FDPeriod'] in script	

**Fund History  
Activity Types:**

- 2: System Rate Change
- 1: Rate Change
- 0: Payment – Deductable
- 1: PayDown – Deductable
- 2: Additional Gift – Deductable
- 3: Non-Cash – Deductable
- 4: Quid pro quo – Deductable
- 5: Payment – Non-Deductable
- 6: PayDown – Non-Deductable
- 7: Credit – Non-Deductable
- 8: Payment from last year – Non-Deductable
- 9: Recurring Charge
- 10: Charge
- 11: Refund – Deductable
- 12: Refund – Non-Deductable
- 13: Hours Pledged
- 14: Hours Completed
- 15: Hours Remaining
- 16: Balance
- 17: Group Total
- 18: Ignore

**Field Listings**

The following pages contain listings of data fields found in the programs.

The first column is the field names as they appear in listings, letters, labels and selections.  
The second column contains the field names as used in embedded code and in report scripting.

An asterisk at the end of the field name indicates that the field cannot be used in selections.

**Relational  
Tables**

Following the Field Listings are relational tables, showing how the different data tables relate to each other.

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Attendance Attn For	Attendance['Attendance Attn For'] PDSDMData.MemAttnTbl['AttnFor']
Attendance Date	Attendance['Attendance Date'] PDSDMData.MemAttnTbl['Date']
Attendance Hours	Attendance['Attendance Hours'] PDSDMData.MemAttnTbl['Hours']
Attendance Reason	Attendance['Attendance Reason'] PDSDMData.MemAttnTbl['Reason']
Attendance Type	Attendance['Attendance Type'] PDSDMData.MemAttnTbl['Type']
Attn Attn For	Attn['Attn Attn For'] PDSDMData.MemAttnTbl['AttnFor']
Attn Date	Attn['Attn Date'] PDSDMData.MemAttnTbl['Date']
Attn Hours	Attn['Attn Hours'] PDSDMData.MemAttnTbl['Hours']
Attn Reason	Attn['Attn Reason'] PDSDMData.MemAttnTbl['Reason']
Attn Type	Attn['Attn Type'] PDSDMData.MemAttnTbl['Type']
Background Chk Date	BackgroundChk['Background Chk Date'] PDSDMData.MemBGTbl['BGDate']
Background Chk Description	BackgroundChk['Background Chk Description'] PDSDMData.MemBGTbl['Description']
Background Chk Note	BackgroundChk['Background Chk Note'] PDSDMData.MemBGTbl['BGNote']
Background Chk Result Name	BackgroundChk['Background Chk Result Name'] PDSDMData.MemBGTbl['ResultName']
Calc Fund Fund Name	CalcFund['Calc Fund Fund Name'] PDSDMCalc.CalcFundTbl['FundName']
Calc Fund Fund Number	CalcFund['Calc Fund Fund Number'] PDSDMCalc.CalcFundTbl['FundNumber']
Calc Fund Goal *	CalcFund['Calc Fund Goal'] PDSDMCalc.CalcFundTbl['Goal']
Calc Fund Has Month Totals	CalcFund['Calc Fund Has Month Totals'] PDSDMCalc.CalcFundTbl['HasMonthTotals']
Calc Fund Order *	CalcFund['Calc Fund Order'] PDSDMCalc.CalcFundTbl['Order']
Calc Fund Total Pledged	CalcFund['Calc Fund Total Pledged'] PDSDMCalc.CalcFundTbl['TotalPledged']
Calc Fund Unique Fund ID *	CalcFund['Calc Fund Unique Fund ID'] PDSDMCalc.CalcFundTbl['FundRecNum']
Calc Fund Act Activity	CalcFundAct['Calc Fund Act Activity'] PDSDMCalc.CalcFundActTbl['Activity']
Calc Fund Act Amount	CalcFundAct['Calc Fund Act Amount'] PDSDMCalc.CalcFundActTbl['Amount']
Calc Fund Act Delinq Amt 1 (31 to 60 Days)	CalcFundAct['Calc Fund Act Delinq Amt 1'] PDSDMCalc.CalcFundActTbl['DelAmt1']
Calc Fund Act Delinq Amt 2 (61 to 90 Days)	CalcFundAct['Calc Fund Act Delinq Amt 2'] PDSDMCalc.CalcFundActTbl['DelAmt2']
Calc Fund Act Delinq Amt 3 (91 or More Days)	CalcFundAct['Calc Fund Act Delinq Amt 3'] PDSDMCalc.CalcFundActTbl['DelAmt3']
Calc Fund Act Frequency	CalcFundAct['Calc Fund Act Frequency'] PDSDMCalc.CalcFundActTbl['Frequency']
Calc Fund Act Function	CalcFundAct['Calc Fund Act Function'] PDSDMCalc.CalcFundActTbl['Function']
Calc Fund Act Fund Number	CalcFundAct['Calc Fund Act Fund Number'] PDSDMCalc.CalcFundActTbl['FundNumber']
Calc Fund Act Group Name	CalcFundAct['Calc Fund Act Group Name'] PDSDMCalc.CalcFundActTbl['GroupName']
Calc Fund Act Group Order *	CalcFundAct['Calc Fund Act Group Order'] PDSDMCalc.CalcFundActTbl['GroupOrder']
Calc Fund Act Order *	CalcFundAct['Calc Fund Act Order'] PDSDMCalc.CalcFundActTbl['Order']

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Calc Fund Act Recap Amount	CalcFundAct['Calc Fund Act Recap Amount'] PDSDMCalc.CalcFundActTbl['RecapAmount']
Calc Fund Act Recur Amount	CalcFundAct['Calc Fund Act Recur Amount'] PDSDMCalc.CalcFundActTbl['RecurAmount']
Calc Fund Act Recur Desc	CalcFundAct['Calc Fund Act Recur Desc'] PDSDMCalc.CalcFundActTbl['RecurDesc']
Calc Fund Act Recur Period	CalcFundAct['Calc Fund Act Recur Period'] PDSDMCalc.CalcFundActTbl['RecurPeriod']
Calc Fund Act Rep Func *	CalcFundAct['Calc Fund Act Rep Func'] PDSDMCalc.CalcFundActTbl['RepFunc']
Calc Fund Act Unique Fam ID *	CalcFundAct['Calc Fund Act Unique Fam ID'] PDSDMCalc.CalcFundActTbl['FamRecNum']
Calc Fund Hist Act Func	CalcFundHist['Calc Fund Hist Act Func'] PDSDMCalc.CalcFundHistTbl['ActFunc']
Calc Fund Hist Activity Name	CalcFundHist['Calc Fund Hist Activity Name'] PDSDMCalc.CalcFundHistTbl['ActivityName']
Calc Fund Hist Amount	CalcFundHist['Calc Fund Hist Amount'] PDSDMCalc.CalcFundHistTbl['FEAmt']
Calc Fund Hist Batch	CalcFundHist['Calc Fund Hist Batch'] PDSDMCalc.CalcFundHistTbl['FEBatch']
Calc Fund Hist Check Number	CalcFundHist['Calc Fund Hist Check Number'] PDSDMCalc.CalcFundHistTbl['FEChk']
Calc Fund Hist Comments	CalcFundHist['Calc Fund Hist Comments'] PDSDMCalc.CalcFundHistTbl['Comments']
Calc Fund Hist Date	CalcFundHist['Calc Fund Hist Date'] PDSDMCalc.CalcFundHistTbl['FEDate']
Calc Fund Hist Date To Apply	CalcFundHist['Calc Fund Hist Date To Apply'] PDSDMCalc.CalcFundHistTbl['DateToApply']
Calc Fund Hist Fund Number	CalcFundHist['Calc Fund Hist Fund Number'] PDSDMCalc.CalcFundHistTbl['FundNumber']
Calc Fund Hist Fund Year	CalcFundHist['Calc Fund Hist Fund Year'] PDSDMCalc.CalcFundHistTbl['FundYear']
Calc Fund Hist Has Comments	CalcFundHist['Calc Fund Hist Has Comments'] PDSDMCalc.CalcFundHistTbl['HasComments']
Calc Fund Hist Member	CalcFundHist['Calc Fund Hist Member'] PDSDMCalc.CalcFundHistTbl['Member']
Calc Fund Hist Recur Amount	CalcFundHist['Calc Fund Hist Recur Amount'] PDSDMCalc.CalcFundHistTbl['RecurAmount']
Calc Fund Hist Unique Act ID *	CalcFundHist['Calc Fund Hist Unique Act ID'] PDSDMCalc.CalcFundHistTbl['ActRecNum']
Calc Fund Hist Unique Fam ID *	CalcFundHist['Calc Fund Hist Unique Fam ID'] PDSDMCalc.CalcFundHistTbl['FEFamRec']
Calc Fund Hist Unique Fund ID *	CalcFundHist['Calc Fund Hist Unique Fund ID'] PDSDMCalc.CalcFundHistTbl['FEFundRec']
Calc Fund Hist Unique ID *	CalcFundHist['Calc Fund Hist Unique ID'] PDSDMCalc.CalcFundHistTbl['FERecNum']
Calc Fund Hist Unique Mem ID *	CalcFundHist['Calc Fund Hist Unique Mem ID'] PDSDMCalc.CalcFundHistTbl['MemRecNum']
Calc Fund Info End Date	CalcFundInfo['Calc Fund Info End Date'] PDSDMCalc.CalcFundInfoTbl['EndDate']
Calc Fund Info Has Balances	CalcFundInfo['Calc Fund Info Has Balances'] PDSDMCalc.CalcFundInfoTbl['HasBalances']
Calc Fund Info Has Hours	CalcFundInfo['Calc Fund Info Has Hours'] PDSDMCalc.CalcFundInfoTbl['HasHours']
Calc Fund Info Has Month Totals	CalcFundInfo['Calc Fund Info Has Month Totals'] PDSDMCalc.CalcFundInfoTbl['HasMonthTotals']
Calc Fund Info Start Date	CalcFundInfo['Calc Fund Info Start Date'] PDSDMCalc.CalcFundInfoTbl['StartDate']
Calc Fund Info Unique Fam ID *	CalcFundInfo['Calc Fund Info Unique Fam ID'] PDSDMCalc.CalcFundInfoTbl['FamRecNum']
Calc Fund Month Balance	CalcFundMonth['Calc Fund Month Balance'] PDSDMCalc.CalcFundMonthTbl['Balance']
Calc Fund Month Credit	CalcFundMonth['Calc Fund Month Credit'] PDSDMCalc.CalcFundMonthTbl['Credit']
Calc Fund Month Disp Date	CalcFundMonth['Calc Fund Month Disp Date'] PDSDMCalc.CalcFundMonthTbl['DispDate']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Calc Fund Month Due	CalcFundMonth['Calc Fund Month Due'] PDSDMCalc.CalcFundMonthTbl['Due']
Calc Fund Month Fund Name	CalcFundMonth['Calc Fund Month Fund Name'] PDSDMCalc.CalcFundMonthTbl['FundName']
Calc Fund Month Fund Number	CalcFundMonth['Calc Fund Month Fund Number'] PDSDMCalc.CalcFundMonthTbl['FundNumber']
Calc Fund Month Gifts	CalcFundMonth['Calc Fund Month Gifts'] PDSDMCalc.CalcFundMonthTbl['Gifts']
Calc Fund Month Hours	CalcFundMonth['Calc Fund Month Hours'] PDSDMCalc.CalcFundMonthTbl['Hours']
Calc Fund Month Hours Due	CalcFundMonth['Calc Fund Month Hours Due'] PDSDMCalc.CalcFundMonthTbl['HoursDue']
Calc Fund Month Ignore	CalcFundMonth['Calc Fund Month Ignore'] PDSDMCalc.CalcFundMonthTbl['Ignore']
Calc Fund Month Month	CalcFundMonth['Calc Fund Month Month'] PDSDMCalc.CalcFundMonthTbl['Month']
Calc Fund Month Month Name	CalcFundMonth['Calc Fund Month Month Name'] PDSDMCalc.CalcFundMonthTbl['MonthName']
Calc Fund Month Paid	CalcFundMonth['Calc Fund Month Paid'] PDSDMCalc.CalcFundMonthTbl['Paid']
Calc Fund Month Paid And Credit	CalcFundMonth['Calc Fund Month Paid And Credit'] PDSDMCalc.CalcFundMonthTbl['PaidAndCredit']
Calc Fund Month Rec Num *	CalcFundMonth['Calc Fund Month Rec Num'] PDSDMCalc.CalcFundMonthTbl['TotRecNum']
Calc Fund Month Refund	CalcFundMonth['Calc Fund Month Refund'] PDSDMCalc.CalcFundMonthTbl['Refund']
Calc Fund Month Tot Date	CalcFundMonth['Calc Fund Month Tot Date'] PDSDMCalc.CalcFundMonthTbl['TotDate']
Calc Fund Month Unique Fam ID *	CalcFundMonth['Calc Fund Month Unique Fam ID'] PDSDMCalc.CalcFundMonthTbl['FamRecNum']
Calc Grp List Activity	CalcGrpList['Calc Grp List Activity'] PDSDMCalc.CalcGrpListTbl['Activity']
Calc Grp List Act Order *	CalcGrpList['Calc Grp List Act Order'] PDSDMCalc.CalcGrpListTbl['ActOrder']
Calc Grp List Delinq Amt 1 (31 to 60 Days)	CalcGrpList['Calc Grp List Delinq Amt 1'] PDSDMCalc.CalcGrpListTbl['DelAmt1']
Calc Grp List Delinq Amt 2 (61 to 90 Days)	CalcGrpList['Calc Grp List Delinq Amt 2'] PDSDMCalc.CalcGrpListTbl['DelAmt2']
Calc Grp List Delinq Amt 3 (91 or More Days)	CalcGrpList['Calc Grp List Delinq Amt 3'] PDSDMCalc.CalcGrpListTbl['DelAmt3']
Calc Grp List Fund Number	CalcGrpList['Calc Grp List Fund Number'] PDSDMCalc.CalcGrpListTbl['FundNumber']
Calc Grp List Hours Completed	CalcGrpList['Calc Grp List Hours Completed'] PDSDMCalc.CalcGrpListTbl['HoursCompleted']
Calc Grp List Hours Pledged	CalcGrpList['Calc Grp List Hours Pledged'] PDSDMCalc.CalcGrpListTbl['HoursPledged']
Calc Grp List Hours Remaining	CalcGrpList['Calc Grp List Hours Remaining'] PDSDMCalc.CalcGrpListTbl['HoursRemaining']
Calc Grp List Is Hours	CalcGrpList['Calc Grp List Is Hours'] PDSDMCalc.CalcGrpListTbl['IsHours']
Calc Grp List Months Delinquent	CalcGrpList['Calc Grp List Months Delinquent'] PDSDMCalc.CalcGrpListTbl['MonthsDelinquent']
Calc Grp List Order *	CalcGrpList['Calc Grp List Order'] PDSDMCalc.CalcGrpListTbl['Order']
Calc Grp List Pmt Freq	CalcGrpList['Calc Grp List Pmt Freq'] PDSDMCalc.CalcGrpListTbl['PmtFreq']
Calc Grp List Recap Bal	CalcGrpList['Calc Grp List Recap Bal'] PDSDMCalc.CalcGrpListTbl['RecapBal']
Calc Grp List Recap Credits	CalcGrpList['Calc Grp List Recap Credits'] PDSDMCalc.CalcGrpListTbl['RecapCredits']
Calc Grp List Recap Due	CalcGrpList['Calc Grp List Recap Due'] PDSDMCalc.CalcGrpListTbl['RecapDue']
Calc Grp List Recap Gifts	CalcGrpList['Calc Grp List Recap Gifts'] PDSDMCalc.CalcGrpListTbl['RecapGifts']
Calc Grp List Recap Paid	CalcGrpList['Calc Grp List Recap Paid'] PDSDMCalc.CalcGrpListTbl['RecapPaid']

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Calc Grp List Recur Amount	CalcGrpList['Calc Grp List Recur Amount'] PDSDMCalc.CalcGrpListTbl['RecurAmount']
Calc Grp List Recur Desc	CalcGrpList['Calc Grp List Recur Desc'] PDSDMCalc.CalcGrpListTbl['RecurDesc']
Calc Grp List Recur Period	CalcGrpList['Calc Grp List Recur Period'] PDSDMCalc.CalcGrpListTbl['RecurPeriod']
Calc Grp List Total Bal	CalcGrpList['Calc Grp List Total Bal'] PDSDMCalc.CalcGrpListTbl['TotalBal']
Calc Grp List Total Credits	CalcGrpList['Calc Grp List Total Credits'] PDSDMCalc.CalcGrpListTbl['TotalCredits']
Calc Grp List Total Due	CalcGrpList['Calc Grp List Total Due'] PDSDMCalc.CalcGrpListTbl['TotalDue']
Calc Grp List Total Gifts	CalcGrpList['Calc Grp List Total Gifts'] PDSDMCalc.CalcGrpListTbl['TotalGifts']
Calc Grp List Total Paid	CalcGrpList['Calc Grp List Total Paid'] PDSDMCalc.CalcGrpListTbl['TotalPaid']
Calc Grp List Unique Fam ID *	CalcGrpList['Calc Grp List Unique Fam ID'] PDSDMCalc.CalcGrpListTbl['FamRecNum']
Calc Rate Desc	CalcRate['Calc Rate Desc'] PDSDMCalc.CalcRateTbl['Desc']
Calc Rate Last Total	CalcRate['Calc Rate Last Total'] PDSDMCalc.CalcRateTbl['LastTotal']
Calc Rate Member	CalcRate['Calc Rate Member'] PDSDMCalc.CalcRateTbl['Member']
Calc Rate Order	CalcRate['Calc Rate Order'] PDSDMCalc.CalcRateTbl['Order']
Calc Rate Period	CalcRate['Calc Rate Period'] PDSDMCalc.CalcRateTbl['Period']
Calc Rate Rate	CalcRate['Calc Rate Rate'] PDSDMCalc.CalcRateTbl['Rate']
Calc Rate Terms	CalcRate['Calc Rate Terms'] PDSDMCalc.CalcRateTbl['Terms']
Calc Rate Unique Act ID	CalcRate['Calc Rate Unique Act ID'] PDSDMCalc.CalcRateTbl['ActRecNum']
Calc Rate Unique Fam ID	CalcRate['Calc Rate Unique Fam ID'] PDSDMCalc.CalcRateTbl['FamRecNum']
Calc Rate Unique Mem ID	CalcRate['Calc Rate Unique Mem ID'] PDSDMCalc.CalcRateTbl['MemRecNum']
E Mail Description	EMail['E Mail Description'] PDSMDData.MemEMailTbl['EMailDesc']
E Mail E Mail Address	EMail['E Mail E Mail Address'] PDSMDData.MemEMailTbl['EMailAddress']
E Mail Prefers E Mail	EMail['E Mail Prefers E Mail'] PDSMDData.MemEMailTbl['EMailOverMail']
Fam Address 1	Fam['Fam Address 1'] PDSMDData.FamTbl['Address1']
Fam Address 2	Fam['Fam Address 2'] PDSMDData.FamTbl['Address2']
Fam Address Remarks	Fam['Fam Address Remarks'] PDSMDData.FamTbl['AddressRemarks']
Fam Address Block *	Fam['Fam Address Block'] PDSMDData.FamTbl['AddressBlock']
Fam Address Changed	Fam['Fam Address Changed'] PDSMDData.FamTbl['AddressChanged']
Fam All Member Names	Fam['Fam All Member Names'] PDSMDData.FamTbl['AllMemberNames']
Fam Alt. Address Day to End	Fam['Fam Alt. Address Day to End'] PDSMDData.FamTbl['AlternateEndDay']
Fam Alt. Address Day to Start	Fam['Fam Alt. Address Day to Start'] PDSMDData.FamTbl['AlternateStartDay']
Fam Alt. Address Month to End	Fam['Fam Alt. Address Month to End'] PDSMDData.FamTbl['AlternateEndMonth']
Fam Alt. Address Month to Start	Fam['Fam Alt. Address Month to Start'] PDSMDData.FamTbl['AlternateStartMonth']
Fam Alternate Address is Certified	Fam['Fam Alternate Address is Certified'] PDSMDData.FamTbl['AlternateCertified']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Alternate Address Line 1	Fam['Fam Alternate Address Line 1'] PDSMDData.FamTbl['AlternateAddress1']
Fam Alternate Address Line 2	Fam['Fam Alternate Address Line 2'] PDSMDData.FamTbl['AlternateAddress2']
Fam Alternate Carrier Route	Fam['Fam Alternate Carrier Route'] PDSMDData.FamTbl['AlternateCarrierRoute']
Fam Alternate DP	Fam['Fam Alternate DP'] PDSMDData.FamTbl['AlternateDP']
Fam Alternate E Mail	Fam['Fam Alternate E Mail'] PDSMDData.FamTbl['AlternateEMail']
Fam Alternate Zip	Fam['Fam Alternate Zip'] PDSMDData.FamTbl['AlternateZip']
Fam Alternate Address Block *	Fam['Fam Alternate Address Block'] PDSMDData.FamTbl['AlternateAddressBlock']
Fam Alternate City	Fam['Fam Alternate City'] PDSMDData.FamTbl['AlternateCity']
Fam Alternate Period	Fam['Fam Alternate Period'] PDSMDData.FamTbl['AlternatePeriod']
Fam Alt Prefers E Mail	Fam['Fam Alt Prefers E Mail'] PDSMDData.FamTbl['AltEMailOverMail']
Fam Blank1 *	Fam['Fam Blank1'] PDSMDData.FamTbl['Blank1']
Fam Blank2 *	Fam['Fam Blank2'] PDSMDData.FamTbl['Blank2']
Fam City	Fam['Fam City'] PDSMDData.FamTbl['City']
Fam City Only	Fam['Fam City Only'] PDSMDData.FamTbl['CityOnly']
Fam Confidential Remarks	Fam['Fam Confidential Remarks'] PDSMDData.FamTbl['ConfRemarks']
Fam Date Alt. Addr Changed	Fam['Fam Date Alt. Addr Changed'] PDSMDData.FamTbl['AlternateChanged']
Fam Date Alternate Addr Certified	Fam['Fam Date Alternate Addr Certified'] PDSMDData.FamTbl['AlternateCertDate']
Fam Date Changed	Fam['Fam Date Changed'] PDSMDData.FamTbl['DateChanged']
Fam Date Created	Fam['Fam Date Created'] PDSMDData.FamTbl['DateCreated']
Fam Date Funds Changed	Fam['Fam Date Funds Changed'] PDSMDData.FamTbl['FundDateChanged']
Fam Date Mailing Addr Certified	Fam['Fam Date Mailing Addr Certified'] PDSMDData.FamTbl['MailingCertDate']
Fam Date Mailing Addr Changed	Fam['Fam Date Mailing Addr Changed'] PDSMDData.FamTbl['MailingChanged']
Fam Date Name Changed	Fam['Fam Date Name Changed'] PDSMDData.FamTbl['NameChanged']
Fam Date Phone Changed	Fam['Fam Date Phone Changed'] PDSMDData.FamTbl['PhoneChanged']
Fam Date Registered	Fam['Fam Date Registered'] PDSMDData.FamTbl['DateRegistered']
Fam Date Street Addr Certified	Fam['Fam Date Street Addr Certified'] PDSMDData.FamTbl['StreetCertDate']
Fam Date Street Addr Changed	Fam['Fam Date Street Addr Changed'] PDSMDData.FamTbl['StreetChanged']
Fam Date Left Parish	Fam['Fam Date Left Parish'] PDSMDData.FamTbl['DateLeftParish']
Fam E Mail	Fam['Fam E Mail'] PDSMDData.FamTbl['CurrentEMail']
Fam Family Status	Fam['Fam Family Status'] PDSMDData.FamTbl['FamilyStatus']
Fam First Name	Fam['Fam First Name'] PDSMDData.FamTbl['FirstName']
Fam Formal Sal	Fam['Fam Formal Sal'] PDSMDData.FamTbl['CurrentFormalSal']
Fam General Remarks	Fam['Fam General Remarks'] PDSMDData.FamTbl['GenRemarks']



## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Geog. Area	Fam['Fam Geog. Area'] PDSMDData.FamTbl['AreaNumber']
Fam ID/Env Number	Fam['Fam ID/Env Number'] PDSMDData.FamTbl['CurrentIDNumber']
Fam Inactive	Fam['Fam Inactive'] PDSMDData.FamTbl['Inactive']
Fam Inactive Members	Fam['Fam Inactive Members'] PDSMDData.FamTbl['InactiveMembers']
Fam Informal Sal	Fam['Fam Informal Sal'] PDSMDData.FamTbl['CurrentInformalSal']
Fam Last Name	Fam['Fam Last Name'] PDSMDData.FamTbl['LastName']
Fam Letter Language	Fam['Fam Letter Language'] PDSMDData.FamTbl['LetterLanguage']
Fam License Name	Fam['Fam License Name'] PDSMDData.FamTbl['LicenseName']
Fam Mailing Address is Certified	Fam['Fam Mailing Address is Certified'] PDSMDData.FamTbl['MailingCertified']
Fam Mailing Address Line 1	Fam['Fam Mailing Address Line 1'] PDSMDData.FamTbl['MailingAddress1']
Fam Mailing Address Line 2	Fam['Fam Mailing Address Line 2'] PDSMDData.FamTbl['MailingAddress2']
Fam Mailing C/O Name	Fam['Fam Mailing C/O Name'] PDSMDData.FamTbl['MailingNameCO']
Fam Mailing Carrier Route	Fam['Fam Mailing Carrier Route'] PDSMDData.FamTbl['MailingCarrierRoute']
Fam Mailing DP	Fam['Fam Mailing DP'] PDSMDData.FamTbl['MailingDP']
Fam Mailing Zip	Fam['Fam Mailing Zip'] PDSMDData.FamTbl['MailingZip']
Fam Mailing Address Block *	Fam['Fam Mailing Address Block'] PDSMDData.FamTbl['MailingAddressBlock']
Fam Mailing City	Fam['Fam Mailing City'] PDSMDData.FamTbl['MailingCity']
Fam Mailing Name	Fam['Fam Mailing Name'] PDSMDData.FamTbl['CurrentMailingName']
Fam Mailing Name Block *	Fam['Fam Mailing Name Block'] PDSMDData.FamTbl['MailingNameBlock']
Fam Member Names	Fam['Fam Member Names'] PDSMDData.FamTbl['MemberNames']
Fam Name Format 1 ( Mr. & Mrs. John Smith, Jr.)	Fam['Fam Name Format 1'] PDSMDData.FamTbl['NameFormat1']
Fam Name Format 10 (nickname of head)	Fam['Fam Name Format 10'] PDSMDData.FamTbl['NameFormat10']
Fam Name Format 11 (nickname of spouse)	Fam['Fam Name Format 11'] PDSMDData.FamTbl['NameFormat11']
Fam Name Format 12 ( Jack & Dot)	Fam['Fam Name Format 12'] PDSMDData.FamTbl['NameFormat12']
Fam Name Format 13 ( John & Mary Smith, Jr.)	Fam['Fam Name Format 13'] PDSMDData.FamTbl['NameFormat13']
Fam Name Format 2 ( Mr. & Mrs. John & Mary Smith, Jr.)	Fam['Fam Name Format 2'] PDSMDData.FamTbl['NameFormat2']
Fam Name Format 3 ( SMITH, JR., John & Mary)	Fam['Fam Name Format 3'] PDSMDData.FamTbl['NameFormat3']
Fam Name Format 4 ( Mr. & Mrs. Smith)	Fam['Fam Name Format 4'] PDSMDData.FamTbl['NameFormat4']
Fam Name Format 5 (first name)	Fam['Fam Name Format 5'] PDSMDData.FamTbl['NameFormat5']
Fam Name Format 6 (spouse name)	Fam['Fam Name Format 6'] PDSMDData.FamTbl['NameFormat6']
Fam Name Format 7 ( Smith, Jr.)	Fam['Fam Name Format 7'] PDSMDData.FamTbl['NameFormat7']
Fam Name Format 8 ( Mr. & Mrs.)	Fam['Fam Name Format 8'] PDSMDData.FamTbl['NameFormat8']
Fam Name Format 9 ( John & Mary)	Fam['Fam Name Format 9'] PDSMDData.FamTbl['NameFormat9']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Name	Fam['Fam Name'] PDSDMData.FamTbl['CurrentName']
Fam Num All Children	Fam['Fam Num All Children'] PDSDMData.FamTbl['NumAllChild']
Fam Num All Members	Fam['Fam Num All Members'] PDSDMData.FamTbl['NumAllMem']
Fam Number Of Children	Fam['Fam Number Of Children'] PDSDMData.FamTbl['NumOfChildren']
Fam Num Children	Fam['Fam Num Children'] PDSDMData.FamTbl['NumChild']
Fam Num Members	Fam['Fam Num Members'] PDSDMData.FamTbl['NumMem']
Fam Orig ID Number *	Fam['Fam Orig ID Number'] PDSDMData.FamTbl['IDNumber']
Fam Orig Mailing Name *	Fam['Fam Orig Mailing Name'] PDSDMData.FamTbl['MailingName']
Fam Orig Formal Sal *	Fam['Fam Orig Formal Sal'] PDSDMData.FamTbl['FormalSal']
Fam Orig Informal Sal *	Fam['Fam Orig Informal Sal'] PDSDMData.FamTbl['InformalSal']
Fam Orig Name *	Fam['Fam Orig Name'] PDSDMData.FamTbl['Name']
Fam Phone List *	Fam['Fam Phone List'] PDSDMData.FamTbl['PhoneList']
Fam Picture	Fam['Fam Picture'] PDSDMData.FamTbl['PictureFile']
Fam Prefers E Mail	Fam['Fam Prefers E Mail'] PDSDMData.FamTbl['CurrentEMailOverMail']
Fam Raw Title *	Fam['Fam Raw Title'] PDSDMData.FamTbl['RawTitle']
Fam Second ID Number	Fam['Fam Second ID Number'] PDSDMData.FamTbl['SecondIDNumber']
Fam Send Mail to Alt. Address	Fam['Fam Send Mail to Alt. Address'] PDSDMData.FamTbl['SendMailToAlternate']
Fam Spouse First	Fam['Fam Spouse First'] PDSDMData.FamTbl['SpouseFirst']
Fam Spouse Last	Fam['Fam Spouse Last'] PDSDMData.FamTbl['SpouseLast']
Fam Spouse Raw Title *	Fam['Fam Spouse Raw Title'] PDSDMData.FamTbl['SpouseRawTitle']
Fam Spouse Title	Fam['Fam Spouse Title'] PDSDMData.FamTbl['SpouseTitle']
Fam State Only	Fam['Fam State Only'] PDSDMData.FamTbl['StateOnly']
Fam Street Address is Certified	Fam['Fam Street Address is Certified'] PDSDMData.FamTbl['StreetCertified']
Fam Street Address Line 1	Fam['Fam Street Address Line 1'] PDSDMData.FamTbl['StreetAddress1']
Fam Street Address Line 2	Fam['Fam Street Address Line 2'] PDSDMData.FamTbl['StreetAddress2']
Fam Street Carrier Route	Fam['Fam Street Carrier Route'] PDSDMData.FamTbl['StreetCarrierRoute']
Fam Street City	Fam['Fam Street City'] PDSDMData.FamTbl['StreetCity']
Fam Street DP	Fam['Fam Street DP'] PDSDMData.FamTbl['StreetDP']
Fam Street E Mail	Fam['Fam Street E Mail'] PDSDMData.FamTbl['EMail']
Fam Street Zip	Fam['Fam Street Zip'] PDSDMData.FamTbl['StreetZip']
Fam Street Address Block *	Fam['Fam Street Address Block'] PDSDMData.FamTbl['StreetAddressBlock']
Fam Street Prefers E Mail	Fam['Fam Street Prefers E Mail'] PDSDMData.FamTbl['EMailOverMail']
Fam Suffix	Fam['Fam Suffix'] PDSDMData.FamTbl['Suffix']

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Title	Fam['Fam Title'] PDSDMData.FamTbl['Title']
Fam Unique ID *	Fam['Fam Unique ID'] PDSDMData.FamTbl['FamRecNum']
Fam Use Alt	Fam['Fam Use Alt'] PDSDMData.FamTbl['UseAlt']
Fam Use E Mail	Fam['Fam Use E Mail'] PDSDMData.FamTbl['UseEmail']
Fam X Phone List *	Fam['Fam X Phone List'] PDSDMData.FamTbl['XPhoneList']
Fam Zip	Fam['Fam Zip'] PDSDMData.FamTbl['Zip']
Fam Zip DP	Fam['Fam Zip DP'] PDSDMData.FamTbl['ZipDP']
Fam Zip Prefix	Fam['Fam Zip Prefix'] PDSDMData.FamTbl['ZipPrefix']
Fam Zip Route	Fam['Fam Zip Route'] PDSDMData.FamTbl['ZipRoute']
Fam Keyword Description	FamKeyword['Fam Keyword Description'] PDSDMData.FamKWtbl['Description']
Fam Letters Date	FamLetters['Fam Letters Date'] PDSDMData.FamLetTbl['Date']
Fam Letters Description	FamLetters['Fam Letters Description'] PDSDMData.FamLetTbl['Name']
Fam Letters Note	FamLetters['Fam Letters Note'] PDSDMData.FamLetTbl['Note']
Fam Letters Type	FamLetters['Fam Letters Type'] PDSDMData.FamLetTbl['Type']
Fam Phone Number	FamPhone['Fam Phone Number'] PDSDMData.FamPhoneTbl['Number']
Fam Phone Type	FamPhone['Fam Phone Type'] PDSDMData.FamPhoneTbl['PhoneType']
Fam Phone Unlisted	FamPhone['Fam Phone Unlisted'] PDSDMData.FamPhoneTbl['Unlisted']
Fund Fund Keyword 1	Fund['Fund Fund Keyword 1'] PDSDMData.FamFundTbl['Keyword1']
Fund Fund Keyword 2	Fund['Fund Fund Keyword 2'] PDSDMData.FamFundTbl['Keyword2']
Fund Fund Number	Fund['Fund Fund Number'] PDSDMData.FamFundTbl['FDFund']
Fund Fund Year	Fund['Fund Fund Year'] PDSDMData.FamFundTbl['FDYear']
Fund Fund Identifier	Fund['Fund Fund Identifier'] PDSDMData.FamFundTbl['FundIdentifier']
Fund Bill Address Line 1	FundBill['Fund Bill Address Line 1'] PDSDMData.FamFundBillTbl['Address1']
Fund Bill Address Line 2	FundBill['Fund Bill Address Line 2'] PDSDMData.FamFundBillTbl['Address2']
Fund Bill Address Block *	FundBill['Fund Bill Address Block'] PDSDMData.FamFundBillTbl['AddressBlock']
Fund Bill C/O a Member	FundBill['Fund Bill C/O a Member'] PDSDMData.FamFundBillTbl['COMember']
Fund Bill Carrier Route	FundBill['Fund Bill Carrier Route'] PDSDMData.FamFundBillTbl['CarrierRoute']
Fund Bill Certified	FundBill['Fund Bill Certified'] PDSDMData.FamFundBillTbl['Certified']
Fund Bill Changed	FundBill['Fund Bill Changed'] PDSDMData.FamFundBillTbl['Changed']
Fund Bill City/State	FundBill['Fund Bill City/State'] PDSDMData.FamFundBillTbl['CityState']
Fund Bill Date Certified	FundBill['Fund Bill Date Certified'] PDSDMData.FamFundBillTbl['CertDate']
Fund Bill DP	FundBill['Fund Bill DP'] PDSDMData.FamFundBillTbl['DP']
Fund Bill E Mail	FundBill['Fund Bill E Mail'] PDSDMData.FamFundBillTbl['Email']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fund Bill End Day	FundBill['Fund Bill End Day'] PDSDMData.FamFundBillTbl['EndDay']
Fund Bill End Month	FundBill['Fund Bill End Month'] PDSDMData.FamFundBillTbl['EndMonth']
Fund Bill Formal Sal	FundBill['Fund Bill Formal Sal'] PDSDMData.FamFundBillTbl['FormalSal']
Fund Bill Informal Sal	FundBill['Fund Bill Informal Sal'] PDSDMData.FamFundBillTbl['InformalSal']
Fund Bill Mailing Name	FundBill['Fund Bill Mailing Name'] PDSDMData.FamFundBillTbl['MailingName']
Fund Bill Member	FundBill['Fund Bill Member'] PDSDMData.FamFundBillTbl['Member']
Fund Bill Name	FundBill['Fund Bill Name'] PDSDMData.FamFundBillTbl['Name']
Fund Bill Phone	FundBill['Fund Bill Phone'] PDSDMData.FamFundBillTbl['Phone']
Fund Bill Prefers E Mail	FundBill['Fund Bill Prefers E Mail'] PDSDMData.FamFundBillTbl['EMailOverMail']
Fund Bill Relationship	FundBill['Fund Bill Relationship'] PDSDMData.FamFundBillTbl['Relationship']
Fund Bill Start Day	FundBill['Fund Bill Start Day'] PDSDMData.FamFundBillTbl['StartDay']
Fund Bill Start Month	FundBill['Fund Bill Start Month'] PDSDMData.FamFundBillTbl['StartMonth']
Fund Bill Statements	FundBill['Fund Bill Statements'] PDSDMData.FamFundBillTbl['BillingTypeName']
Fund Bill Zip	FundBill['Fund Bill Zip'] PDSDMData.FamFundBillTbl['Zip']
Fund Bill Zip Prefix	FundBill['Fund Bill Zip Prefix'] PDSDMData.FamFundBillTbl['ZipPrefix']
Fund EFT Activity	FundEFT['Fund EFT Activity'] PDSDMData.FamFundEFTTbl['Activity']
Fund EFT CC Type Name	FundEFT['Fund EFT CC Type Name'] PDSDMData.FamFundEFTTbl['CCTypeName']
Fund EFT EFT Type	FundEFT['Fund EFT EFT Type'] PDSDMData.FamFundEFTTbl['EFTType']
Fund EFT EFT Type Name	FundEFT['Fund EFT EFT Type Name'] PDSDMData.FamFundEFTTbl['EFTTypeName']
Fund Hist Activity	FundHist['Fund Hist Activity'] PDSDMData.FamFundHistTbl['FEType']
Fund Hist Amount	FundHist['Fund Hist Amount'] PDSDMData.FamFundHistTbl['FEAmt']
Fund Hist Batch Number	FundHist['Fund Hist Batch Number'] PDSDMData.FamFundHistTbl['FEBatch']
Fund Hist Check Number	FundHist['Fund Hist Check Number'] PDSDMData.FamFundHistTbl['FEChk']
Fund Hist Comment	FundHist['Fund Hist Comment'] PDSDMData.FamFundHistTbl['FEComment']
Fund Hist Date	FundHist['Fund Hist Date'] PDSDMData.FamFundHistTbl['FEDate']
Fund Hist Member	FundHist['Fund Hist Member'] PDSDMData.FamFundHistTbl['Member']
Fund Rate Activity	FundRate['Fund Rate Activity'] PDSDMData.FamFundRateTbl['Activity']
Fund Rate Associate with Member	FundRate['Fund Rate Associate with Member'] PDSDMData.FamFundRateTbl['FDUseMem']
Fund Rate End Date	FundRate['Fund Rate End Date'] PDSDMData.FamFundRateTbl['FDEndDate']
Fund Rate Last Rate	FundRate['Fund Rate Last Rate'] PDSDMData.FamFundRateTbl['LastRate']
Fund Rate Last Total	FundRate['Fund Rate Last Total'] PDSDMData.FamFundRateTbl['LastTotal']
Fund Rate Member Name	FundRate['Fund Rate Member Name'] PDSDMData.FamFundRateTbl['MemberName']
Fund Rate Rate	FundRate['Fund Rate Rate'] PDSDMData.FamFundRateTbl['FDRate']

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fund Rate Start Date	FundRate['Fund Rate Start Date'] PDSDMData.FamFundRateTbl['FDStartDate']
Fund Rate Terms	FundRate['Fund Rate Terms'] PDSDMData.FamFundRateTbl['Terms']
Fund Rate Terms Period	FundRate['Fund Rate Terms Period'] PDSDMData.FamFundRateTbl['FDPeriod']
Fund Rate Total	FundRate['Fund Rate Total'] PDSDMData.FamFundRateTbl['FDTotal']
Fund Rate Use EFT	FundRate['Fund Rate Use EFT'] PDSDMData.FamFundRateTbl['UseEFT']
Fund Totals Accum Delinq Amt 1 (31 or More Days)	FundTotals['Fund Totals Accum Delinq Amt 1'] PDSDMCalc.FundTotalsTbl['AccumDelAmt1']
Fund Totals Accum Delinq Amt 2 (61 or More Days)	FundTotals['Fund Totals Accum Delinq Amt 2'] PDSDMCalc.FundTotalsTbl['AccumDelAmt2']
Fund Totals Accum Delinq Amt 3 (91 or More Days)	FundTotals['Fund Totals Accum Delinq Amt 3'] PDSDMCalc.FundTotalsTbl['AccumDelAmt3']
Fund Totals Delinq Amt 1 (31 to 60 Days)	FundTotals['Fund Totals Delinq Amt 1'] PDSDMCalc.FundTotalsTbl['DelAmt1']
Fund Totals Delinq Amt 2 (61 to 90 Days)	FundTotals['Fund Totals Delinq Amt 2'] PDSDMCalc.FundTotalsTbl['DelAmt2']
Fund Totals Delinq Amt 3 (91 or More Days)	FundTotals['Fund Totals Delinq Amt 3'] PDSDMCalc.FundTotalsTbl['DelAmt3']
Fund Totals Grand Total Balance	FundTotals['Fund Totals Grand Total Balance'] PDSDMCalc.FundTotalsTbl['GrandTotalBalance']
Fund Totals Grand Total Credit	FundTotals['Fund Totals Grand Total Credit'] PDSDMCalc.FundTotalsTbl['GrandTotalCredit']
Fund Totals Grand Total Deductible	FundTotals['Fund Totals Grand Total Deductible'] PDSDMCalc.FundTotalsTbl['GrandTotalDeductible']
Fund Totals Grand Total Due	FundTotals['Fund Totals Grand Total Due'] PDSDMCalc.FundTotalsTbl['GrandTotalDue']
Fund Totals Grand Total Gifts	FundTotals['Fund Totals Grand Total Gifts'] PDSDMCalc.FundTotalsTbl['GrandTotalGifts']
Fund Totals Grand Total Non Cash	FundTotals['Fund Totals Grand Total Non Cash'] PDSDMCalc.FundTotalsTbl['GrandTotalNonCash']
Fund Totals Grand Total Non Ded	FundTotals['Fund Totals Grand Total Non Ded'] PDSDMCalc.FundTotalsTbl['GrandTotalNonDed']
Fund Totals Grand Total Paid	FundTotals['Fund Totals Grand Total Paid'] PDSDMCalc.FundTotalsTbl['GrandTotalPaid']
Fund Totals Grand Total Pledged	FundTotals['Fund Totals Grand Total Pledged'] PDSDMCalc.FundTotalsTbl['GrandTotalPledged']
Fund Totals Grand Total Refund	FundTotals['Fund Totals Grand Total Refund'] PDSDMCalc.FundTotalsTbl['GrandTotalRefund']
Fund Totals Grand Total Total Paid	FundTotals['Fund Totals Grand Total Total Paid'] PDSDMCalc.FundTotalsTbl['GrandTotalTotalPaid']
Fund Totals Has Quid Pro Quo	FundTotals['Fund Totals Has Quid Pro Quo'] PDSDMCalc.FundTotalsTbl['HasQuidProQuo']
Fund Totals Lastest Payment Date	FundTotals['Fund Totals Lastest Payment Date'] PDSDMCalc.FundTotalsTbl['LastestPaymentDate']
Fund Totals Payment Percentage	FundTotals['Fund Totals Payment Percentage'] PDSDMCalc.FundTotalsTbl['PaymentPercentage']
Fund Totals Progress Index	FundTotals['Fund Totals Progress Index'] PDSDMCalc.FundTotalsTbl['ProgressIndex']
Fund Totals Progress Pct Amt	FundTotals['Fund Totals Progress Pct Amt'] PDSDMCalc.FundTotalsTbl['ProgressPctAmt']
Fund Totals Progress Pct Time	FundTotals['Fund Totals Progress Pct Time'] PDSDMCalc.FundTotalsTbl['ProgressPctTime']
Fund Totals Recap Balance	FundTotals['Fund Totals Recap Balance'] PDSDMCalc.FundTotalsTbl['RecapBalance']
Fund Totals Recap Credit	FundTotals['Fund Totals Recap Credit'] PDSDMCalc.FundTotalsTbl['RecapCredit']
Fund Totals Recap Deductible	FundTotals['Fund Totals Recap Deductible'] PDSDMCalc.FundTotalsTbl['RecapDeductible']
Fund Totals Recap Due	FundTotals['Fund Totals Recap Due'] PDSDMCalc.FundTotalsTbl['RecapDue']
Fund Totals Recap Gifts	FundTotals['Fund Totals Recap Gifts'] PDSDMCalc.FundTotalsTbl['RecapGifts']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fund Totals Recap Non Cash	FundTotals['Fund Totals Recap Non Cash'] PDSDMCalc.FundTotalsTbl['RecapNonCash']
Fund Totals Recap Non Ded	FundTotals['Fund Totals Recap Non Ded'] PDSDMCalc.FundTotalsTbl['RecapNonDed']
Fund Totals Recap Paid	FundTotals['Fund Totals Recap Paid'] PDSDMCalc.FundTotalsTbl['RecapPaid']
Fund Totals Recap Refund	FundTotals['Fund Totals Recap Refund'] PDSDMCalc.FundTotalsTbl['RecapRefund']
Fund Totals Recap Total Paid	FundTotals['Fund Totals Recap Total Paid'] PDSDMCalc.FundTotalsTbl['RecapTotalPaid']
Fund Totals Tax Statement Required	FundTotals['Fund Totals Tax Statement Required'] PDSDMCalc.FundTotalsTbl['TaxStatementRequired']
Keywords Description	Keywords['Keywords Description'] PDSDMData.MemKWTbl['Description']
Letter Date	Letter['Letter Date'] PDSDMData.MemLetTbl['Date']
Letter Description	Letter['Letter Description'] PDSDMData.MemLetTbl['Name']
Letter Note	Letter['Letter Note'] PDSDMData.MemLetTbl['Note']
Letter Type	Letter['Letter Type'] PDSDMData.MemLetTbl['Type']
Mem Age	Mem['Mem Age'] PDSDMData.MemTbl['Age']
Mem Confidential Remarks	Mem['Mem Confidential Remarks'] PDSDMData.MemTbl['ConfRemarks']
Mem Current Informal Sal	Mem['Mem Current Informal Sal'] PDSDMData.MemTbl['CurrentInformalSal']
Mem Date Changed	Mem['Mem Date Changed'] PDSDMData.MemTbl['DateChanged']
Mem Date Created	Mem['Mem Date Created'] PDSDMData.MemTbl['DateCreated']
Mem Date of Birth	Mem['Mem Date of Birth'] PDSDMData.MemTbl['DateOfBirth']
Mem Day of Birth	Mem['Mem Day of Birth'] PDSDMData.MemTbl['DayOfBirth']
Mem Deceased	Mem['Mem Deceased'] PDSDMData.MemTbl['Deceased']
Mem Different Last Name	Mem['Mem Different Last Name'] PDSDMData.MemTbl['DifLastName']
Mem Disability	Mem['Mem Disability'] PDSDMData.MemTbl['UserKW2']
Mem Ethnicity	Mem['Mem Ethnicity'] PDSDMData.MemTbl['Ethnicity']
Mem Family Name	Mem['Mem Family Name'] PDSDMData.MemTbl['FamilyName']
Mem Fam Unique ID *	Mem['Mem Fam Unique ID'] PDSDMData.MemTbl['FamRecNum']
Mem First Name	Mem['Mem First Name'] PDSDMData.MemTbl['FirstName']
Mem Formal Sal	Mem['Mem Formal Sal'] PDSDMData.MemTbl['FormalSal']
Mem Formal Sal	Mem['Mem Formal Sal'] PDSDMData.MemTbl['CurrentFormalSal']
Mem Full Maiden Name *	Mem['Mem Full Maiden Name'] PDSDMData.MemTbl['FullMaidenName']
Mem Gender	Mem['Mem Gender'] PDSDMData.MemTbl['Gender']
Mem General Remarks	Mem['Mem General Remarks'] PDSDMData.MemTbl['GenRemarks']
Mem Grade	Mem['Mem Grade'] PDSDMData.MemTbl['Grade']
Mem ID/Env Number	Mem['Mem ID/Env Number'] PDSDMData.MemTbl['MemIDNumber']
Mem Inactive	Mem['Mem Inactive'] PDSDMData.MemTbl['Inactive']

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Mem Informal Sal	Mem['Mem Informal Sal'] PDSMDData.MemTbl['InformalSal']
Mem Language	Mem['Mem Language'] PDSMDData.MemTbl['Language']
Mem Last Name	Mem['Mem Last Name'] PDSMDData.MemTbl['LastName']
Mem Letter Language	Mem['Mem Letter Language'] PDSMDData.MemTbl['LetterLanguage']
Mem Location	Mem['Mem Location'] PDSMDData.MemTbl['Location']
Mem Maiden Name	Mem['Mem Maiden Name'] PDSMDData.MemTbl['MaidenName']
Mem Mailing Name	Mem['Mem Mailing Name'] PDSMDData.MemTbl['CurrentMailingName']
Mem Marital Status	Mem['Mem Marital Status'] PDSMDData.MemTbl['MaritalStatus']
Mem Month Num of Birth	Mem['Mem Month Num of Birth'] PDSMDData.MemTbl['MonthNumofBirth']
Mem Month of Birth	Mem['Mem Month of Birth'] PDSMDData.MemTbl['MonthOfBirth']
Mem Name Format 1 ( Mr. John Smith, Jr.)	Mem['Mem Name Format 1'] PDSMDData.MemTbl['NameFormat1']
Mem Name Format 10 (maiden name)	Mem['Mem Name Format 10'] PDSMDData.MemTbl['NameFormat10']
Mem Name Format 11 ( Jack's)	Mem['Mem Name Format 11'] PDSMDData.MemTbl['NameFormat11']
Mem Name Format 2 ( Mr. Jack Smith, Jr.)	Mem['Mem Name Format 2'] PDSMDData.MemTbl['NameFormat2']
Mem Name Format 3 ( SMITH, JR., John)	Mem['Mem Name Format 3'] PDSMDData.MemTbl['NameFormat3']
Mem Name Format 4 ( Mr. Smith)	Mem['Mem Name Format 4'] PDSMDData.MemTbl['NameFormat4']
Mem Name Format 5 ( John)	Mem['Mem Name Format 5'] PDSMDData.MemTbl['NameFormat5']
Mem Name Format 6 (nickname)	Mem['Mem Name Format 6'] PDSMDData.MemTbl['NameFormat6']
Mem Name Format 7 ( Smith, Jr.)	Mem['Mem Name Format 7'] PDSMDData.MemTbl['NameFormat7']
Mem Name Format 8 ( Mr.)	Mem['Mem Name Format 8'] PDSMDData.MemTbl['NameFormat8']
Mem Name Format 9 ( Jack Smith)	Mem['Mem Name Format 9'] PDSMDData.MemTbl['NameFormat9']
Mem Name	Mem['Mem Name'] PDSMDData.MemTbl['Name']
Mem Nickname	Mem['Mem Nickname'] PDSMDData.MemTbl['Nickname']
Mem Occupation	Mem['Mem Occupation'] PDSMDData.MemTbl['UserKW3']
Mem Oldest Child	Mem['Mem Oldest Child'] PDSMDData.MemTbl['OldestChild']
Mem Phone List *	Mem['Mem Phone List'] PDSMDData.MemTbl['PhoneList']
Mem Picture	Mem['Mem Picture'] PDSMDData.MemTbl['PictureFile']
Mem Raw Title *	Mem['Mem Raw Title'] PDSMDData.MemTbl['RawTitle']
Mem Relationship	Mem['Mem Relationship'] PDSMDData.MemTbl['Relationship']
Mem Religion	Mem['Mem Religion'] PDSMDData.MemTbl['UserKW1']
Mem Separate Statement	Mem['Mem Separate Statement'] PDSMDData.MemTbl['SeparateStatement']
Mem Suffix	Mem['Mem Suffix'] PDSMDData.MemTbl['Suffix']
Mem Title	Mem['Mem Title'] PDSMDData.MemTbl['Title']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Mem Type Number *	Mem['Mem Type Number'] PDSDMData.MemTbl['MemberType']
Mem Type	Mem['Mem Type'] PDSDMData.MemTbl['Type']
Mem Unique ID *	Mem['Mem Unique ID'] PDSDMData.MemTbl['MemRecNum']
Mem Volunteer List	Mem['Mem Volunteer List'] PDSDMData.MemTbl['VolunteerList']
Mem X Phone List *	Mem['Mem X Phone List'] PDSDMData.MemTbl['XPhoneList']
Mem Year of Birth	Mem['Mem Year of Birth'] PDSDMData.MemTbl['YearOfBirth']
Mem Youngest Child	Mem['Mem Youngest Child'] PDSDMData.MemTbl['YoungestChild']
Ministries End Date	Ministries['Ministries End Date'] PDSDMData.MemMinTbl['EndDate']
Ministries Ministry	Ministries['Ministries Ministry'] PDSDMData.MemMinTbl['MinName']
Ministries Start Date	Ministries['Ministries Start Date'] PDSDMData.MemMinTbl['StartDate']
Ministries Status	Ministries['Ministries Status'] PDSDMData.MemMinTbl['MinStatusName']
Other Req Date	OtherReq['Other Req Date'] PDSDMData.MemReqTbl['ReqDate']
Other Req Description	OtherReq['Other Req Description'] PDSDMData.MemReqTbl['Description']
Other Req Note	OtherReq['Other Req Note'] PDSDMData.MemReqTbl['ReqNote']
Other Req Result Name	OtherReq['Other Req Result Name'] PDSDMData.MemReqTbl['ResultName']
Phone Number	Phone['Phone Number'] PDSDMData.MemPhoneTbl['Number']
Phone Phone Type	Phone['Phone Phone Type'] PDSDMData.MemPhoneTbl['PhoneType']
Phone Unlisted	Phone['Phone Unlisted'] PDSDMData.MemPhoneTbl['Unlisted']
Sac 1st Comm Addl Status	Sac['Sac 1st Comm Addl Status'] PDSDMData.MemSacLookTbl['Date4Addl']
Sac 1st Comm Date	Sac['Sac 1st Comm Date'] PDSDMData.MemSacLookTbl['Date4Date']
Sac 1st Comm Day of 1st Comm	Sac['Sac 1st Comm Day of 1st Comm'] PDSDMData.MemSacLookTbl['Date4DayofDate4']
Sac 1st Comm Extra Info	Sac['Sac 1st Comm Extra Info'] PDSDMData.MemSacLookTbl['Date4Extra']
Sac 1st Comm Month Num of 1st Comm	Sac['Sac 1st Comm Month Num of 1st Comm'] PDSDMData.MemSacLookTbl['Date4MonthNumofDate4']
Sac 1st Comm Month of 1st Comm	Sac['Sac 1st Comm Month of 1st Comm'] PDSDMData.MemSacLookTbl['Date4MonthofDate4']
Sac 1st Comm Notes	Sac['Sac 1st Comm Notes'] PDSDMData.MemSacLookTbl['Date4Notes']
Sac 1st Comm Performed By	Sac['Sac 1st Comm Performed By'] PDSDMData.MemSacLookTbl['Date4PerformedBy']
Sac 1st Comm Performed	Sac['Sac 1st Comm Performed'] PDSDMData.MemSacLookTbl['Date4Perf']
Sac 1st Comm Place	Sac['Sac 1st Comm Place'] PDSDMData.MemSacLookTbl['Date4PlaceBlock']
Sac 1st Comm Sponsor List	Sac['Sac 1st Comm Sponsor List'] PDSDMData.MemSacLookTbl['Date4SponsorList']
Sac 1st Comm Status	Sac['Sac 1st Comm Status'] PDSDMData.MemSacLookTbl['Date4Status']
Sac 1st Comm Year of 1st Comm	Sac['Sac 1st Comm Year of 1st Comm'] PDSDMData.MemSacLookTbl['Date4YearofDate4']
Sac Baptism Addl Status	Sac['Sac Baptism Addl Status'] PDSDMData.MemSacLookTbl['Date1Addl']
Sac Baptism Baptismal Name	Sac['Sac Baptism Baptismal Name'] PDSDMData.MemSacLookTbl['Date1Extra']



## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Sac Baptism Date	Sac['Sac Baptism Date'] PDSDMData.MemSacLookTbl['Date1Date']
Sac Baptism Day of Baptism	Sac['Sac Baptism Day of Baptism'] PDSDMData.MemSacLookTbl['Date1DayOfDate1']
Sac Baptism Month Num of Baptism	Sac['Sac Baptism Month Num of Baptism'] PDSDMData.MemSacLookTbl['Date1MonthNumofDate1']
Sac Baptism Month of Baptism	Sac['Sac Baptism Month of Baptism'] PDSDMData.MemSacLookTbl['Date1MonthofDate1']
Sac Baptism Notes	Sac['Sac Baptism Notes'] PDSDMData.MemSacLookTbl['Date1Notes']
Sac Baptism Performed By	Sac['Sac Baptism Performed By'] PDSDMData.MemSacLookTbl['Date1PerformedBy']
Sac Baptism Performed	Sac['Sac Baptism Performed'] PDSDMData.MemSacLookTbl['Date1Perf']
Sac Baptism Place	Sac['Sac Baptism Place'] PDSDMData.MemSacLookTbl['Date1PlaceBlock']
Sac Baptism Sponsor List	Sac['Sac Baptism Sponsor List'] PDSDMData.MemSacLookTbl['Date1SponsorList']
Sac Baptism Status	Sac['Sac Baptism Status'] PDSDMData.MemSacLookTbl['Date1Status']
Sac Baptism Year of Baptism	Sac['Sac Baptism Year of Baptism'] PDSDMData.MemSacLookTbl['Date1YearofDate1']
Sac Confirm Addl Status	Sac['Sac Confirm Addl Status'] PDSDMData.MemSacLookTbl['Date2Addl']
Sac Confirm Confirmation Name	Sac['Sac Confirm Confirmation Name'] PDSDMData.MemSacLookTbl['Date2Extra']
Sac Confirm Date	Sac['Sac Confirm Date'] PDSDMData.MemSacLookTbl['Date2Date']
Sac Confirm Day of Confirm	Sac['Sac Confirm Day of Confirm'] PDSDMData.MemSacLookTbl['Date2DayofDate2']
Sac Confirm Month Num of Confirm	Sac['Sac Confirm Month Num of Confirm'] PDSDMData.MemSacLookTbl['Date2MonthNumofDate2']
Sac Confirm Month of Confirm	Sac['Sac Confirm Month of Confirm'] PDSDMData.MemSacLookTbl['Date2MonthofDate2']
Sac Confirm Notes	Sac['Sac Confirm Notes'] PDSDMData.MemSacLookTbl['Date2Notes']
Sac Confirm Performed By	Sac['Sac Confirm Performed By'] PDSDMData.MemSacLookTbl['Date2PerformedBy']
Sac Confirm Performed	Sac['Sac Confirm Performed'] PDSDMData.MemSacLookTbl['Date2Perf']
Sac Confirm Place	Sac['Sac Confirm Place'] PDSDMData.MemSacLookTbl['Date2PlaceBlock']
Sac Confirm Sponsor List	Sac['Sac Confirm Sponsor List'] PDSDMData.MemSacLookTbl['Date2SponsorList']
Sac Confirm Status	Sac['Sac Confirm Status'] PDSDMData.MemSacLookTbl['Date2Status']
Sac Confirm Year of Confirm	Sac['Sac Confirm Year of Confirm'] PDSDMData.MemSacLookTbl['Date2YearofDate2']
Sac Marriage Addl Status	Sac['Sac Marriage Addl Status'] PDSDMData.MemSacLookTbl['Date3Addl']
Sac Marriage Date	Sac['Sac Marriage Date'] PDSDMData.MemSacLookTbl['Date3Date']
Sac Marriage Day of Marriage	Sac['Sac Marriage Day of Marriage'] PDSDMData.MemSacLookTbl['Date3DayofDate3']
Sac Marriage Extra Info	Sac['Sac Marriage Extra Info'] PDSDMData.MemSacLookTbl['Date3Extra']
Sac Marriage Month Num of Marriage	Sac['Sac Marriage Month Num of Marriage'] PDSDMData.MemSacLookTbl['Date3MonthNumofDate3']
Sac Marriage Month of Marriage	Sac['Sac Marriage Month of Marriage'] PDSDMData.MemSacLookTbl['Date3MonthofDate3']
Sac Marriage Notes	Sac['Sac Marriage Notes'] PDSDMData.MemSacLookTbl['Date3Notes']
Sac Marriage Performed By	Sac['Sac Marriage Performed By'] PDSDMData.MemSacLookTbl['Date3PerformedBy']
Sac Marriage Performed	Sac['Sac Marriage Performed'] PDSDMData.MemSacLookTbl['Date3Perf']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Sac Marriage Place	Sac['Sac Marriage Place'] PDSDMData.MemSacLookTbl['Date3PlaceBlock']
Sac Marriage Sponsor List	Sac['Sac Marriage Sponsor List'] PDSDMData.MemSacLookTbl['Date3SponsorList']
Sac Marriage Status	Sac['Sac Marriage Status'] PDSDMData.MemSacLookTbl['Date3Status']
Sac Marriage Year of Marriage	Sac['Sac Marriage Year of Marriage'] PDSDMData.MemSacLookTbl['Date3YearofDate3']
Sac Penance Addl Status	Sac['Sac Penance Addl Status'] PDSDMData.MemSacLookTbl['Date5Addl']
Sac Penance Date	Sac['Sac Penance Date'] PDSDMData.MemSacLookTbl['Date5Date']
Sac Penance Day of Penance	Sac['Sac Penance Day of Penance'] PDSDMData.MemSacLookTbl['Date5DayofDate5']
Sac Penance Extra Info	Sac['Sac Penance Extra Info'] PDSDMData.MemSacLookTbl['Date5Extra']
Sac Penance Month Num of Penance	Sac['Sac Penance Month Num of Penance'] PDSDMData.MemSacLookTbl['Date5MonthNumofDate5']
Sac Penance Month of Penance	Sac['Sac Penance Month of Penance'] PDSDMData.MemSacLookTbl['Date5MonthofDate5']
Sac Penance Notes	Sac['Sac Penance Notes'] PDSDMData.MemSacLookTbl['Date5Notes']
Sac Penance Performed By	Sac['Sac Penance Performed By'] PDSDMData.MemSacLookTbl['Date5PerformedBy']
Sac Penance Performed	Sac['Sac Penance Performed'] PDSDMData.MemSacLookTbl['Date5Perf']
Sac Penance Place	Sac['Sac Penance Place'] PDSDMData.MemSacLookTbl['Date5PlaceBlock']
Sac Penance Sponsor List	Sac['Sac Penance Sponsor List'] PDSDMData.MemSacLookTbl['Date5SponsorList']
Sac Penance Status	Sac['Sac Penance Status'] PDSDMData.MemSacLookTbl['Date5Status']
Sac Penance Year of Penance	Sac['Sac Penance Year of Penance'] PDSDMData.MemSacLookTbl['Date5YearofDate5']
Sac Extra Confidential Notes	SacExtra['Sac Extra Confidential Notes'] PDSDMData.MemSacTbl['ConfNotes']
Sac Extra Extra Information	SacExtra['Sac Extra Extra Information'] PDSDMData.MemSacTbl['ExtraInfo']
Sac Extra General Notes	SacExtra['Sac Extra General Notes'] PDSDMData.MemSacTbl['GenNotes']
Sac Extra Performed By	SacExtra['Sac Extra Performed By'] PDSDMData.MemSacTbl['PerformedBy']
Sac General Birth Place	SacGeneral['Sac General Birth Place'] PDSDMData.AskTbl['BirthPlace']
Sac General Date Changed	SacGeneral['Sac General Date Changed'] PDSDMData.AskTbl['DateChanged']
Sac General Date Created	SacGeneral['Sac General Date Created'] PDSDMData.AskTbl['DateCreated']
Sac General Father's Name	SacGeneral['Sac General Father's Name'] PDSDMData.AskTbl['FatherName']
Sac General Mother's Maiden Name	SacGeneral['Sac General Mother's Maiden Name'] PDSDMData.AskTbl['MothersMaiden']
Sac General Mother's Name	SacGeneral['Sac General Mother's Name'] PDSDMData.AskTbl['MotherName']
Sac List Addl Name	SacList['Sac List Addl Name'] PDSDMData.MemDatesTbl['AddlName']
Sac List Addl Status	SacList['Sac List Addl Status'] PDSDMData.MemDatesTbl['AddlStatus']
Sac List Address	SacList['Sac List Address'] PDSDMData.MemDatesTbl['DateAddress']
Sac List City State	SacList['Sac List City State'] PDSDMData.MemDatesTbl['DateCityState']
Sac List Date	SacList['Sac List Date'] PDSDMData.MemDatesTbl['Date']
Sac List Day	SacList['Sac List Day'] PDSDMData.MemDatesTbl['Day']

## Church Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Sac List E Mail	SacList['Sac List E Mail'] PDSMDData.MemDatesTbl['DateEMail']
Sac List Month Num	SacList['Sac List Month Num'] PDSMDData.MemDatesTbl['MonthNum']
Sac List Month	SacList['Sac List Month'] PDSMDData.MemDatesTbl['Month']
Sac List Name	SacList['Sac List Name'] PDSMDData.MemDatesTbl['DateName']
Sac List Order Num	SacList['Sac List Order Num'] PDSMDData.MemDatesTbl['OrderNum']
Sac List Place	SacList['Sac List Place'] PDSMDData.MemDatesTbl['DatePlace']
Sac List Status	SacList['Sac List Status'] PDSMDData.MemDatesTbl['Status']
Sac List Status Name	SacList['Sac List Status Name'] PDSMDData.MemDatesTbl['StatusName']
Sac List Use E Mail	SacList['Sac List Use E Mail'] PDSMDData.MemDatesTbl['DateUseEMail']
Sac List Year	SacList['Sac List Year'] PDSMDData.MemDatesTbl['Year']
Sac List Zip	SacList['Sac List Zip'] PDSMDData.MemDatesTbl['DateZip']
Sac Sponsor Address	SacSponsor['Sac Sponsor Address'] PDSMDData.MemSponsTbl['Address']
Sac Sponsor City State	SacSponsor['Sac Sponsor City State'] PDSMDData.MemSponsTbl['CityState']
Sac Sponsor Formal Sal	SacSponsor['Sac Sponsor Formal Sal'] PDSMDData.MemSponsTbl['FormalSal']
Sac Sponsor Informal Sal	SacSponsor['Sac Sponsor Informal Sal'] PDSMDData.MemSponsTbl['InformalSal']
Sac Sponsor Mailing Name	SacSponsor['Sac Sponsor Mailing Name'] PDSMDData.MemSponsTbl['MailingName']
Sac Sponsor Name	SacSponsor['Sac Sponsor Name'] PDSMDData.MemSponsTbl['Name']
Sac Sponsor Phone	SacSponsor['Sac Sponsor Phone'] PDSMDData.MemSponsTbl['Phone']
Sac Sponsor Sponsor Type	SacSponsor['Sac Sponsor Sponsor Type'] PDSMDData.MemSponsTbl['SponsorType']
Sac Sponsor Unlisted	SacSponsor['Sac Sponsor Unlisted'] PDSMDData.MemSponsTbl['Unl']
Sac Sponsor Zip Code	SacSponsor['Sac Sponsor Zip Code'] PDSMDData.MemSponsTbl['ZipCode']
Talents End Date	Talents['Talents End Date'] PDSMDData.MemTalTbl['EndDate']
Talents Start Date	Talents['Talents Start Date'] PDSMDData.MemTalTbl['StartDate']
Talents Status	Talents['Talents Status'] PDSMDData.MemTalTbl['TalStatusName']
Talents Talent	Talents['Talents Talent'] PDSMDData.MemTalTbl['TalName']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Attendance Attn For	Attendance['Attendance Attn For'] PDSDMData.MemAttnTbl['AttnFor']
Attendance Date	Attendance['Attendance Date'] PDSDMData.MemAttnTbl['Date']
Attendance Date	Attendance['Attendance Date'] PDSDMData.MemREAttnTbl['Date']
Attendance Hours	Attendance['Attendance Hours'] PDSDMData.MemAttnTbl['Hours']
Attendance Hours	Attendance['Attendance Hours'] PDSDMData.MemREAttnTbl['Hours']
Attendance Reason	Attendance['Attendance Reason'] PDSDMData.MemAttnTbl['Reason']
Attendance Reason	Attendance['Attendance Reason'] PDSDMData.MemREAttnTbl['Reason']
Attendance Session Name	Attendance['Attendance Session Name'] PDSDMData.MemREAttnTbl['SessionName']
Attendance Type	Attendance['Attendance Type'] PDSDMData.MemAttnTbl['Type']
Attendance Type	Attendance['Attendance Type'] PDSDMData.MemREAttnTbl['Type']
Background Chk Date	BackgroundChk['Background Chk Date'] PDSDMData.MemBGTbl['BGDate']
Background Chk Description	BackgroundChk['Background Chk Description'] PDSDMData.MemBGTbl['Description']
Background Chk Note	BackgroundChk['Background Chk Note'] PDSDMData.MemBGTbl['BGNote']
Background Chk Result Name	BackgroundChk['Background Chk Result Name'] PDSDMData.MemBGTbl['ResultName']
Calc Fund Fund Name	CalcFund['Calc Fund Fund Name'] PDSDMCalc.CalcFundTbl['FundName']
Calc Fund Fund Number	CalcFund['Calc Fund Fund Number'] PDSDMCalc.CalcFundTbl['FundNumber']
Calc Fund Goal *	CalcFund['Calc Fund Goal'] PDSDMCalc.CalcFundTbl['Goal']
Calc Fund Has Month Totals	CalcFund['Calc Fund Has Month Totals'] PDSDMCalc.CalcFundTbl['HasMonthTotals']
Calc Fund Order *	CalcFund['Calc Fund Order'] PDSDMCalc.CalcFundTbl['Order']
Calc Fund Total Pledged	CalcFund['Calc Fund Total Pledged'] PDSDMCalc.CalcFundTbl['TotalPledged']
Calc Fund Unique Fund ID *	CalcFund['Calc Fund Unique Fund ID'] PDSDMCalc.CalcFundTbl['FundRecNum']
Calc Fund Act Activity	CalcFundAct['Calc Fund Act Activity'] PDSDMCalc.CalcFundActTbl['Activity']
Calc Fund Act Amount	CalcFundAct['Calc Fund Act Amount'] PDSDMCalc.CalcFundActTbl['Amount']
Calc Fund Act Delinq Amt 1 (31 to 60 Days)	CalcFundAct['Calc Fund Act Delinq Amt 1'] PDSDMCalc.CalcFundActTbl['DelAmt1']
Calc Fund Act Delinq Amt 2 (61 to 90 Days)	CalcFundAct['Calc Fund Act Delinq Amt 2'] PDSDMCalc.CalcFundActTbl['DelAmt2']
Calc Fund Act Delinq Amt 3 (91 or More Days)	CalcFundAct['Calc Fund Act Delinq Amt 3'] PDSDMCalc.CalcFundActTbl['DelAmt3']
Calc Fund Act Frequency	CalcFundAct['Calc Fund Act Frequency'] PDSDMCalc.CalcFundActTbl['Frequency']
Calc Fund Act Function	CalcFundAct['Calc Fund Act Function'] PDSDMCalc.CalcFundActTbl['Function']
Calc Fund Act Fund Number	CalcFundAct['Calc Fund Act Fund Number'] PDSDMCalc.CalcFundActTbl['FundNumber']
Calc Fund Act Group Name	CalcFundAct['Calc Fund Act Group Name'] PDSDMCalc.CalcFundActTbl['GroupName']
Calc Fund Act Group Order *	CalcFundAct['Calc Fund Act Group Order'] PDSDMCalc.CalcFundActTbl['GroupOrder']
Calc Fund Act Order *	CalcFundAct['Calc Fund Act Order'] PDSDMCalc.CalcFundActTbl['Order']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Calc Fund Act Recap Amount	CalcFundAct['Calc Fund Act Recap Amount'] PDSDMCalc.CalcFundActTbl['RecapAmount']
Calc Fund Act Recur Amount	CalcFundAct['Calc Fund Act Recur Amount'] PDSDMCalc.CalcFundActTbl['RecurAmount']
Calc Fund Act Recur Desc	CalcFundAct['Calc Fund Act Recur Desc'] PDSDMCalc.CalcFundActTbl['RecurDesc']
Calc Fund Act Recur Period	CalcFundAct['Calc Fund Act Recur Period'] PDSDMCalc.CalcFundActTbl['RecurPeriod']
Calc Fund Act Rep Func *	CalcFundAct['Calc Fund Act Rep Func'] PDSDMCalc.CalcFundActTbl['RepFunc']
Calc Fund Act Unique Fam ID *	CalcFundAct['Calc Fund Act Unique Fam ID'] PDSDMCalc.CalcFundActTbl['FamRecNum']
Calc Fund Hist Act Func	CalcFundHist['Calc Fund Hist Act Func'] PDSDMCalc.CalcFundHistTbl['ActFunc']
Calc Fund Hist Activity Name	CalcFundHist['Calc Fund Hist Activity Name'] PDSDMCalc.CalcFundHistTbl['ActivityName']
Calc Fund Hist Amount	CalcFundHist['Calc Fund Hist Amount'] PDSDMCalc.CalcFundHistTbl['FEAmt']
Calc Fund Hist Batch	CalcFundHist['Calc Fund Hist Batch'] PDSDMCalc.CalcFundHistTbl['FEBatch']
Calc Fund Hist Check Number	CalcFundHist['Calc Fund Hist Check Number'] PDSDMCalc.CalcFundHistTbl['FEChk']
Calc Fund Hist Comments	CalcFundHist['Calc Fund Hist Comments'] PDSDMCalc.CalcFundHistTbl['Comments']
Calc Fund Hist Date	CalcFundHist['Calc Fund Hist Date'] PDSDMCalc.CalcFundHistTbl['FEDate']
Calc Fund Hist Date To Apply	CalcFundHist['Calc Fund Hist Date To Apply'] PDSDMCalc.CalcFundHistTbl['DateToApply']
Calc Fund Hist Fund Number	CalcFundHist['Calc Fund Hist Fund Number'] PDSDMCalc.CalcFundHistTbl['FundNumber']
Calc Fund Hist Fund Year	CalcFundHist['Calc Fund Hist Fund Year'] PDSDMCalc.CalcFundHistTbl['FundYear']
Calc Fund Hist Has Comments	CalcFundHist['Calc Fund Hist Has Comments'] PDSDMCalc.CalcFundHistTbl['HasComments']
Calc Fund Hist Member	CalcFundHist['Calc Fund Hist Member'] PDSDMCalc.CalcFundHistTbl['Member']
Calc Fund Hist Recur Amount	CalcFundHist['Calc Fund Hist Recur Amount'] PDSDMCalc.CalcFundHistTbl['RecurAmount']
Calc Fund Hist Unique Act ID *	CalcFundHist['Calc Fund Hist Unique Act ID'] PDSDMCalc.CalcFundHistTbl['ActRecNum']
Calc Fund Hist Unique Fam ID *	CalcFundHist['Calc Fund Hist Unique Fam ID'] PDSDMCalc.CalcFundHistTbl['FEFamRec']
Calc Fund Hist Unique Fund ID *	CalcFundHist['Calc Fund Hist Unique Fund ID'] PDSDMCalc.CalcFundHistTbl['FEFundRec']
Calc Fund Hist Unique ID *	CalcFundHist['Calc Fund Hist Unique ID'] PDSDMCalc.CalcFundHistTbl['FERecNum']
Calc Fund Hist Unique Mem ID *	CalcFundHist['Calc Fund Hist Unique Mem ID'] PDSDMCalc.CalcFundHistTbl['MemRecNum']
Calc Fund Info End Date	CalcFundInfo['Calc Fund Info End Date'] PDSDMCalc.CalcFundInfoTbl['EndDate']
Calc Fund Info Has Balances	CalcFundInfo['Calc Fund Info Has Balances'] PDSDMCalc.CalcFundInfoTbl['HasBalances']
Calc Fund Info Has Hours	CalcFundInfo['Calc Fund Info Has Hours'] PDSDMCalc.CalcFundInfoTbl['HasHours']
Calc Fund Info Has Month Totals	CalcFundInfo['Calc Fund Info Has Month Totals'] PDSDMCalc.CalcFundInfoTbl['HasMonthTotals']
Calc Fund Info Start Date	CalcFundInfo['Calc Fund Info Start Date'] PDSDMCalc.CalcFundInfoTbl['StartDate']
Calc Fund Info Unique Fam ID *	CalcFundInfo['Calc Fund Info Unique Fam ID'] PDSDMCalc.CalcFundInfoTbl['FamRecNum']
Calc Fund Month Balance	CalcFundMonth['Calc Fund Month Balance'] PDSDMCalc.CalcFundMonthTbl['Balance']
Calc Fund Month Credit	CalcFundMonth['Calc Fund Month Credit'] PDSDMCalc.CalcFundMonthTbl['Credit']
Calc Fund Month Disp Date	CalcFundMonth['Calc Fund Month Disp Date'] PDSDMCalc.CalcFundMonthTbl['DispDate']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Calc Fund Month Due	CalcFundMonth['Calc Fund Month Due'] PDSDMCalc.CalcFundMonthTbl['Due']
Calc Fund Month Fund Name	CalcFundMonth['Calc Fund Month Fund Name'] PDSDMCalc.CalcFundMonthTbl['FundName']
Calc Fund Month Fund Number	CalcFundMonth['Calc Fund Month Fund Number'] PDSDMCalc.CalcFundMonthTbl['FundNumber']
Calc Fund Month Gifts	CalcFundMonth['Calc Fund Month Gifts'] PDSDMCalc.CalcFundMonthTbl['Gifts']
Calc Fund Month Hours	CalcFundMonth['Calc Fund Month Hours'] PDSDMCalc.CalcFundMonthTbl['Hours']
Calc Fund Month Hours Due	CalcFundMonth['Calc Fund Month Hours Due'] PDSDMCalc.CalcFundMonthTbl['HoursDue']
Calc Fund Month Ignore	CalcFundMonth['Calc Fund Month Ignore'] PDSDMCalc.CalcFundMonthTbl['Ignore']
Calc Fund Month Month	CalcFundMonth['Calc Fund Month Month'] PDSDMCalc.CalcFundMonthTbl['Month']
Calc Fund Month Month Name	CalcFundMonth['Calc Fund Month Month Name'] PDSDMCalc.CalcFundMonthTbl['MonthName']
Calc Fund Month Paid	CalcFundMonth['Calc Fund Month Paid'] PDSDMCalc.CalcFundMonthTbl['Paid']
Calc Fund Month Paid And Credit	CalcFundMonth['Calc Fund Month Paid And Credit'] PDSDMCalc.CalcFundMonthTbl['PaidAndCredit']
Calc Fund Month Rec Num *	CalcFundMonth['Calc Fund Month Rec Num'] PDSDMCalc.CalcFundMonthTbl['TotRecNum']
Calc Fund Month Refund	CalcFundMonth['Calc Fund Month Refund'] PDSDMCalc.CalcFundMonthTbl['Refund']
Calc Fund Month Tot Date	CalcFundMonth['Calc Fund Month Tot Date'] PDSDMCalc.CalcFundMonthTbl['TotDate']
Calc Fund Month Unique Fam ID *	CalcFundMonth['Calc Fund Month Unique Fam ID'] PDSDMCalc.CalcFundMonthTbl['FamRecNum']
Calc Grp List Activity	CalcGrpList['Calc Grp List Activity'] PDSDMCalc.CalcGrpListTbl['Activity']
Calc Grp List Act Order *	CalcGrpList['Calc Grp List Act Order'] PDSDMCalc.CalcGrpListTbl['ActOrder']
Calc Grp List Delinq Amt 1 (31 to 60 Days)	CalcGrpList['Calc Grp List Delinq Amt 1'] PDSDMCalc.CalcGrpListTbl['DelAmt1']
Calc Grp List Delinq Amt 2 (61 to 90 Days)	CalcGrpList['Calc Grp List Delinq Amt 2'] PDSDMCalc.CalcGrpListTbl['DelAmt2']
Calc Grp List Delinq Amt 3 (91 or More Days)	CalcGrpList['Calc Grp List Delinq Amt 3'] PDSDMCalc.CalcGrpListTbl['DelAmt3']
Calc Grp List Fund Number	CalcGrpList['Calc Grp List Fund Number'] PDSDMCalc.CalcGrpListTbl['FundNumber']
Calc Grp List Hours Completed	CalcGrpList['Calc Grp List Hours Completed'] PDSDMCalc.CalcGrpListTbl['HoursCompleted']
Calc Grp List Hours Pledged	CalcGrpList['Calc Grp List Hours Pledged'] PDSDMCalc.CalcGrpListTbl['HoursPledged']
Calc Grp List Hours Remaining	CalcGrpList['Calc Grp List Hours Remaining'] PDSDMCalc.CalcGrpListTbl['HoursRemaining']
Calc Grp List Is Hours	CalcGrpList['Calc Grp List Is Hours'] PDSDMCalc.CalcGrpListTbl['IsHours']
Calc Grp List Months Delinquent	CalcGrpList['Calc Grp List Months Delinquent'] PDSDMCalc.CalcGrpListTbl['MonthsDelinquent']
Calc Grp List Order *	CalcGrpList['Calc Grp List Order'] PDSDMCalc.CalcGrpListTbl['Order']
Calc Grp List Pmt Freq	CalcGrpList['Calc Grp List Pmt Freq'] PDSDMCalc.CalcGrpListTbl['PmtFreq']
Calc Grp List Recap Bal	CalcGrpList['Calc Grp List Recap Bal'] PDSDMCalc.CalcGrpListTbl['RecapBal']
Calc Grp List Recap Credits	CalcGrpList['Calc Grp List Recap Credits'] PDSDMCalc.CalcGrpListTbl['RecapCredits']
Calc Grp List Recap Due	CalcGrpList['Calc Grp List Recap Due'] PDSDMCalc.CalcGrpListTbl['RecapDue']
Calc Grp List Recap Gifts	CalcGrpList['Calc Grp List Recap Gifts'] PDSDMCalc.CalcGrpListTbl['RecapGifts']
Calc Grp List Recap Paid	CalcGrpList['Calc Grp List Recap Paid'] PDSDMCalc.CalcGrpListTbl['RecapPaid']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Calc Grp List Recur Amount	CalcGrpList['Calc Grp List Recur Amount'] PDSDMCalc.CalcGrpListTbl['RecurAmount']
Calc Grp List Recur Desc	CalcGrpList['Calc Grp List Recur Desc'] PDSDMCalc.CalcGrpListTbl['RecurDesc']
Calc Grp List Recur Period	CalcGrpList['Calc Grp List Recur Period'] PDSDMCalc.CalcGrpListTbl['RecurPeriod']
Calc Grp List Total Bal	CalcGrpList['Calc Grp List Total Bal'] PDSDMCalc.CalcGrpListTbl['TotalBal']
Calc Grp List Total Credits	CalcGrpList['Calc Grp List Total Credits'] PDSDMCalc.CalcGrpListTbl['TotalCredits']
Calc Grp List Total Due	CalcGrpList['Calc Grp List Total Due'] PDSDMCalc.CalcGrpListTbl['TotalDue']
Calc Grp List Total Gifts	CalcGrpList['Calc Grp List Total Gifts'] PDSDMCalc.CalcGrpListTbl['TotalGifts']
Calc Grp List Total Paid	CalcGrpList['Calc Grp List Total Paid'] PDSDMCalc.CalcGrpListTbl['TotalPaid']
Calc Grp List Unique Fam ID *	CalcGrpList['Calc Grp List Unique Fam ID'] PDSDMCalc.CalcGrpListTbl['FamRecNum']
Calc Rate Desc	CalcRate['Calc Rate Desc'] PDSDMCalc.CalcRateTbl['Desc']
Calc Rate Last Total	CalcRate['Calc Rate Last Total'] PDSDMCalc.CalcRateTbl['LastTotal']
Calc Rate Member	CalcRate['Calc Rate Member'] PDSDMCalc.CalcRateTbl['Member']
Calc Rate Order	CalcRate['Calc Rate Order'] PDSDMCalc.CalcRateTbl['Order']
Calc Rate Period	CalcRate['Calc Rate Period'] PDSDMCalc.CalcRateTbl['Period']
Calc Rate Rate	CalcRate['Calc Rate Rate'] PDSDMCalc.CalcRateTbl['Rate']
Calc Rate Terms	CalcRate['Calc Rate Terms'] PDSDMCalc.CalcRateTbl['Terms']
Calc Rate Unique Act ID	CalcRate['Calc Rate Unique Act ID'] PDSDMCalc.CalcRateTbl['ActRecNum']
Calc Rate Unique Fam ID	CalcRate['Calc Rate Unique Fam ID'] PDSDMCalc.CalcRateTbl['FamRecNum']
Calc Rate Unique Mem ID	CalcRate['Calc Rate Unique Mem ID'] PDSDMCalc.CalcRateTbl['MemRecNum']
Class Calculate Days Present	Class['Class Calculate Days Present'] PDSMDMData.REClassTbl['CalcPres']
Class Class Frequency	Class['Class Class Frequency'] PDSMDMData.REClassTbl['FreqStr']
Class Date Changed	Class['Class Date Changed'] PDSMDMData.REClassTbl['DateChanged']
Class Date of First Class	Class['Class Date of First Class'] PDSMDMData.REClassTbl['StartDate']
Class Date of Last Class	Class['Class Date of Last Class'] PDSMDMData.REClassTbl['EndDate']
Class Dates Altered	Class['Class Dates Altered'] PDSMDMData.REClassTbl['DatesAltered']
Class Grade	Class['Class Grade'] PDSMDMData.REClassTbl['Grade']
Class Inactive	Class['Class Inactive'] PDSMDMData.REClassTbl['Inactive']
Class Minimum Attendance	Class['Class Minimum Attendance'] PDSMDMData.REClassTbl['MinAttend']
Class Name	Class['Class Name'] PDSMDMData.REClassTbl['Name']
Class Room	Class['Class Room'] PDSMDMData.REClassTbl['Room']
Class Time	Class['Class Time'] PDSMDMData.REClassTbl['Time']
Class Unique ID	Class['Class Unique ID'] PDSMDMData.REClassTbl['REClassRecNum']
Class Ctchst Name	ClassCtchst['Class Ctchst Name'] PDSMDMData.REClassTeaTbl['TeacherName']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code
Class Ctchst Position	ClassCtchst['Class Ctchst Position'] PDSMDData.REClassTeaTbl['Position']
Class Date Attendance Required	ClassDate['Class Date Attendance Required'] PDSMDData.REClassDateTbl['AttnRequired']
Class Date Class Meets	ClassDate['Class Date Class Meets'] PDSMDData.REClassDateTbl['ClassMeets']
Class Date Date	ClassDate['Class Date Date'] PDSMDData.REClassDateTbl['ClassDate']
Class Date Note	ClassDate['Class Date Note'] PDSMDData.REClassDateTbl['Note']
Contact Address is Certified	Contact['Contact Address is Certified'] PDSMDData.MemREContTbl['AddressCertified']
Contact Address	Contact['Contact Address'] PDSMDData.MemREContTbl['Address']
Contact Address Block	Contact['Contact Address Block'] PDSMDData.MemREContTbl['AddressBlock']
Contact Carrier Route	Contact['Contact Carrier Route'] PDSMDData.MemREContTbl['CarrierRoute']
Contact City	Contact['Contact City'] PDSMDData.MemREContTbl['City']
Contact Date Address Certified	Contact['Contact Date Address Certified'] PDSMDData.MemREContTbl['AddressCertDate']
Contact Date Address Changed	Contact['Contact Date Address Changed'] PDSMDData.MemREContTbl['AddressChanged']
Contact DP	Contact['Contact DP'] PDSMDData.MemREContTbl['DP']
Contact E Mail	Contact['Contact E Mail'] PDSMDData.MemREContTbl['EMail']
Contact Formal Sal	Contact['Contact Formal Sal'] PDSMDData.MemREContTbl['FormalSal']
Contact Informal Sal	Contact['Contact Informal Sal'] PDSMDData.MemREContTbl['InformalSal']
Contact Mailing Name	Contact['Contact Mailing Name'] PDSMDData.MemREContTbl['MailingName']
Contact Marital Status	Contact['Contact Marital Status'] PDSMDData.MemREContTbl['MaritalStatus']
Contact Name	Contact['Contact Name'] PDSMDData.MemREContTbl['Name']
Contact Phone List	Contact['Contact Phone List'] PDSMDData.MemREContTbl['PhoneList']
Contact Prefers E Mail	Contact['Contact Prefers E Mail'] PDSMDData.MemREContTbl['EMailOverMail']
Contact Relationship	Contact['Contact Relationship'] PDSMDData.MemREContTbl['Relationship']
Contact Religion	Contact['Contact Religion'] PDSMDData.MemREContTbl['User1']
Contact Remarks	Contact['Contact Remarks'] PDSMDData.MemREContTbl['Remarks']
Contact Send Courtesy Copy	Contact['Contact Send Courtesy Copy'] PDSMDData.MemREContTbl['SendCourtesyCopy']
Contact X Phone List	Contact['Contact X Phone List'] PDSMDData.MemREContTbl['XPhoneList']
Contact Zip	Contact['Contact Zip'] PDSMDData.MemREContTbl['Zip']
Contact Zip Prefix	Contact['Contact Zip Prefix'] PDSMDData.MemREContTbl['ZipPrefix']
Contact Phone Number	ContactPhone['Contact Phone Number'] PDSMDData.MemREContPhTbl['Number']
Contact Phone Phone Type	ContactPhone['Contact Phone Phone Type'] PDSMDData.MemREContPhTbl['PhoneType']
Contact Phone Unlisted	ContactPhone['Contact Phone Unlisted'] PDSMDData.MemREContPhTbl['Unlisted']
Ctchst Age	Ctchst['Ctchst Age'] PDSMDData.MemTbl['Age']
Ctchst Confidential Remarks	Ctchst['Ctchst Confidential Remarks'] PDSMDData.MemTbl['ConfRemarks']



## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ctchst Current Informal Sal	Ctchst['Ctchst Current Informal Sal'] PDSMDData.MemTbl['CurrentInformalSal']
Ctchst Date Changed	Ctchst['Ctchst Date Changed'] PDSMDData.MemTbl['DateChanged']
Ctchst Date Changed	Ctchst['Ctchst Date Changed'] PDSMDData.MemTbl['REDateChanged']
Ctchst Date Created	Ctchst['Ctchst Date Created'] PDSMDData.MemTbl['DateCreated']
Ctchst Date of Birth	Ctchst['Ctchst Date of Birth'] PDSMDData.MemTbl['DateOfBirth']
Ctchst Day of Birth	Ctchst['Ctchst Day of Birth'] PDSMDData.MemTbl['DayOfBirth']
Ctchst Deceased	Ctchst['Ctchst Deceased'] PDSMDData.MemTbl['Deceased']
Ctchst Different Last Name	Ctchst['Ctchst Different Last Name'] PDSMDData.MemTbl['DifLastName']
Ctchst Ethnicity	Ctchst['Ctchst Ethnicity'] PDSMDData.MemTbl['Ethnicity']
Ctchst Family Name	Ctchst['Ctchst Family Name'] PDSMDData.MemTbl['FamilyName']
Ctchst Fam Unique ID *	Ctchst['Ctchst Fam Unique ID'] PDSMDData.MemTbl['FamRecNum']
Ctchst First Name	Ctchst['Ctchst First Name'] PDSMDData.MemTbl['FirstName']
Ctchst Formal Sal	Ctchst['Ctchst Formal Sal'] PDSMDData.MemTbl['FormalSal']
Ctchst Formal Sal	Ctchst['Ctchst Formal Sal'] PDSMDData.MemTbl['CurrentFormalSal']
Ctchst Formation Member	Ctchst['Ctchst Formation Member'] PDSMDData.MemTbl['REMember1']
Ctchst Full Maiden Name *	Ctchst['Ctchst Full Maiden Name'] PDSMDData.MemTbl['FullMaidenName']
Ctchst Gender	Ctchst['Ctchst Gender'] PDSMDData.MemTbl['Gender']
Ctchst General Remarks	Ctchst['Ctchst General Remarks'] PDSMDData.MemTbl['GenRemarks']
Ctchst Grade	Ctchst['Ctchst Grade'] PDSMDData.MemTbl['Grade']
Ctchst ID Number	Ctchst['Ctchst ID Number'] PDSMDData.MemTbl['REIDNumber']
Ctchst Inactive	Ctchst['Ctchst Inactive'] PDSMDData.MemTbl['Inactive']
Ctchst Informal Sal	Ctchst['Ctchst Informal Sal'] PDSMDData.MemTbl['InformalSal']
Ctchst Is Catechist	Ctchst['Ctchst Is Catechist'] PDSMDData.MemTbl['Teacher']
Ctchst Is Parent	Ctchst['Ctchst Is Parent'] PDSMDData.MemTbl['Parent']
Ctchst Is Student	Ctchst['Ctchst Is Student'] PDSMDData.MemTbl['Student']
Ctchst Language	Ctchst['Ctchst Language'] PDSMDData.MemTbl['Language']
Ctchst Last Name	Ctchst['Ctchst Last Name'] PDSMDData.MemTbl['LastName']
Ctchst Letter Language	Ctchst['Ctchst Letter Language'] PDSMDData.MemTbl['LetterLanguage']
Ctchst Location	Ctchst['Ctchst Location'] PDSMDData.MemTbl['Location']
Ctchst Maiden Name	Ctchst['Ctchst Maiden Name'] PDSMDData.MemTbl['MaidenName']
Ctchst Mailing Name	Ctchst['Ctchst Mailing Name'] PDSMDData.MemTbl['CurrentMailingName']
Ctchst Marital Status	Ctchst['Ctchst Marital Status'] PDSMDData.MemTbl['MaritalStatus']
Ctchst Month Num of Birth	Ctchst['Ctchst Month Num of Birth'] PDSMDData.MemTbl['MonthNumofBirth']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ctchst Month of Birth	Ctchst['Ctchst Month of Birth'] PDSMDData.MemTbl['MonthOfBirth']
Ctchst Name Format 1 ( Mr. John Smith, Jr.)	Ctchst['Ctchst Name Format 1'] PDSMDData.MemTbl['NameFormat1']
Ctchst Name Format 10 (maiden name)	Ctchst['Ctchst Name Format 10'] PDSMDData.MemTbl['NameFormat10']
Ctchst Name Format 11 ( Jack's)	Ctchst['Ctchst Name Format 11'] PDSMDData.MemTbl['NameFormat11']
Ctchst Name Format 2 ( Mr. Jack Smith, Jr.)	Ctchst['Ctchst Name Format 2'] PDSMDData.MemTbl['NameFormat2']
Ctchst Name Format 3 ( SMITH, JR., John)	Ctchst['Ctchst Name Format 3'] PDSMDData.MemTbl['NameFormat3']
Ctchst Name Format 4 ( Mr. Smith)	Ctchst['Ctchst Name Format 4'] PDSMDData.MemTbl['NameFormat4']
Ctchst Name Format 5 ( John)	Ctchst['Ctchst Name Format 5'] PDSMDData.MemTbl['NameFormat5']
Ctchst Name Format 6 (nickname)	Ctchst['Ctchst Name Format 6'] PDSMDData.MemTbl['NameFormat6']
Ctchst Name Format 7 ( Smith, Jr.)	Ctchst['Ctchst Name Format 7'] PDSMDData.MemTbl['NameFormat7']
Ctchst Name Format 8 ( Mr.)	Ctchst['Ctchst Name Format 8'] PDSMDData.MemTbl['NameFormat8']
Ctchst Name Format 9 ( Jack Smith)	Ctchst['Ctchst Name Format 9'] PDSMDData.MemTbl['NameFormat9']
Ctchst Name	Ctchst['Ctchst Name'] PDSMDData.MemTbl['Name']
Ctchst Nickname	Ctchst['Ctchst Nickname'] PDSMDData.MemTbl['Nickname']
Ctchst Occupation	Ctchst['Ctchst Occupation'] PDSMDData.MemTbl['UserKW3']
Ctchst Oldest Child	Ctchst['Ctchst Oldest Child'] PDSMDData.MemTbl['OldestChild']
Ctchst Phone List *	Ctchst['Ctchst Phone List'] PDSMDData.MemTbl['PhoneList']
Ctchst Picture	Ctchst['Ctchst Picture'] PDSMDData.MemTbl['PictureFile']
Ctchst Raw Title *	Ctchst['Ctchst Raw Title'] PDSMDData.MemTbl['RawTitle']
Ctchst Relationship	Ctchst['Ctchst Relationship'] PDSMDData.MemTbl['Relationship']
Ctchst Religion	Ctchst['Ctchst Religion'] PDSMDData.MemTbl['UserKW1']
Ctchst School	Ctchst['Ctchst School'] PDSMDData.MemTbl['UserKW4']
Ctchst Suffix	Ctchst['Ctchst Suffix'] PDSMDData.MemTbl['Suffix']
Ctchst Title	Ctchst['Ctchst Title'] PDSMDData.MemTbl['Title']
Ctchst Type Number *	Ctchst['Ctchst Type Number'] PDSMDData.MemTbl['MemberType']
Ctchst Type	Ctchst['Ctchst Type'] PDSMDData.MemTbl['Type']
Ctchst Unique ID *	Ctchst['Ctchst Unique ID'] PDSMDData.MemTbl['MemRecNum']
Ctchst Volunteer List	Ctchst['Ctchst Volunteer List'] PDSMDData.MemTbl['VolunteerList']
Ctchst X Phone List *	Ctchst['Ctchst X Phone List'] PDSMDData.MemTbl['XPhoneList']
Ctchst Year of Birth	Ctchst['Ctchst Year of Birth'] PDSMDData.MemTbl['YearOfBirth']
Ctchst Youngest Child	Ctchst['Ctchst Youngest Child'] PDSMDData.MemTbl['YoungestChild']
Ct Detail College Area	CtDetail['Ct Detail College Area'] PDSMDData.RETeaTbl['CollegeArea']
Ct Detail College Degree	CtDetail['Ct Detail College Degree'] PDSMDData.RETeaTbl['CollegeDegree']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ct Detail College Name	CtDetail['Ct Detail College Name'] PDSMDData.RETeaTbl['CollegeName']
Ct Detail College Year Completed	CtDetail['Ct Detail College Year Completed'] PDSMDData.RETeaTbl['CollegeYearComp']
Ct Detail College Years	CtDetail['Ct Detail College Years'] PDSMDData.RETeaTbl['CollegeYears']
Ct Detail Date Started	CtDetail['Ct Detail Date Started'] PDSMDData.RETeaTbl['DateStarted']
Ct Detail Experience	CtDetail['Ct Detail Experience'] PDSMDData.RETeaTbl['Experience']
Ct Detail Grad School Area	CtDetail['Ct Detail Grad School Area'] PDSMDData.RETeaTbl['GradArea']
Ct Detail Grad School Degree	CtDetail['Ct Detail Grad School Degree'] PDSMDData.RETeaTbl['GradDegree']
Ct Detail Grad School Name	CtDetail['Ct Detail Grad School Name'] PDSMDData.RETeaTbl['GradName']
Ct Detail Grad School Year Completed	CtDetail['Ct Detail Grad School Year Completed'] PDSMDData.RETeaTbl['GradYearComp']
Ct Detail Grad School Years	CtDetail['Ct Detail Grad School Years'] PDSMDData.RETeaTbl['GradYears']
Ct Detail High School Area of Study	CtDetail['Ct Detail High School Area of Study'] PDSMDData.RETeaTbl['HighSchArea']
Ct Detail High School Location	CtDetail['Ct Detail High School Location'] PDSMDData.RETeaTbl['HighSchLocation']
Ct Detail High School Name	CtDetail['Ct Detail High School Name'] PDSMDData.RETeaTbl['HighSchName']
Ct Detail High School Years	CtDetail['Ct Detail High School Years'] PDSMDData.RETeaTbl['HighSchYears']
Ct Detail Hight School Degree	CtDetail['Ct Detail Hight School Degree'] PDSMDData.RETeaTbl['HighSchDegree']
Ct Detail Hight School Year Completed	CtDetail['Ct Detail Hight School Year Completed'] PDSMDData.RETeaTbl['HighSchYearComp']
Ct Detail Level 1 Calc Cert	CtDetail['Ct Detail Level 1 Calc Cert'] PDSMDData.RETeaTbl['CalcCertLevel1']
Ct Detail Level 1 Certified	CtDetail['Ct Detail Level 1 Certified'] PDSMDData.RETeaTbl['RCertLevel1']
Ct Detail Level 1 Date Certified	CtDetail['Ct Detail Level 1 Date Certified'] PDSMDData.RETeaTbl['RCertDateLevel1']
Ct Detail Level 1 Expired	CtDetail['Ct Detail Level 1 Expired'] PDSMDData.RETeaTbl['RExpiredLevel1']
Ct Detail Level 1 Remaining	CtDetail['Ct Detail Level 1 Remaining'] PDSMDData.RETeaTbl['RRemainingLevel1']
Ct Detail Level 2 Calc Cert	CtDetail['Ct Detail Level 2 Calc Cert'] PDSMDData.RETeaTbl['CalcCertLevel2']
Ct Detail Level 2 Certified	CtDetail['Ct Detail Level 2 Certified'] PDSMDData.RETeaTbl['RCertLevel2']
Ct Detail Level 2 Date Certified	CtDetail['Ct Detail Level 2 Date Certified'] PDSMDData.RETeaTbl['RCertDateLevel2']
Ct Detail Level 2 Expired	CtDetail['Ct Detail Level 2 Expired'] PDSMDData.RETeaTbl['RExpiredLevel2']
Ct Detail Level 2 Remaining	CtDetail['Ct Detail Level 2 Remaining'] PDSMDData.RETeaTbl['RRemainingLevel2']
Ct Detail Level 3 Calc Cert	CtDetail['Ct Detail Level 3 Calc Cert'] PDSMDData.RETeaTbl['CalcCertLevel3']
Ct Detail Level 3 Certified	CtDetail['Ct Detail Level 3 Certified'] PDSMDData.RETeaTbl['RCertLevel3']
Ct Detail Level 3 Date Certified	CtDetail['Ct Detail Level 3 Date Certified'] PDSMDData.RETeaTbl['RCertDateLevel3']
Ct Detail Level 3 Expired	CtDetail['Ct Detail Level 3 Expired'] PDSMDData.RETeaTbl['RExpiredLevel3']
Ct Detail Level 3 Remaining	CtDetail['Ct Detail Level 3 Remaining'] PDSMDData.RETeaTbl['RRemainingLevel3']
Ct Detail Level 4 Calc Cert	CtDetail['Ct Detail Level 4 Calc Cert'] PDSMDData.RETeaTbl['CalcCertLevel4']
Ct Detail Level 4 Certified	CtDetail['Ct Detail Level 4 Certified'] PDSMDData.RETeaTbl['RCertLevel4']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ct Detail Level 4 Date Certified	CtDetail['Ct Detail Level 4 Date Certified'] PDSMDData.RETeaTbl['RCertDateLevel4']
Ct Detail Level 4 Expired	CtDetail['Ct Detail Level 4 Expired'] PDSMDData.RETeaTbl['REExpiredLevel4']
Ct Detail Level 4 Remaining	CtDetail['Ct Detail Level 4 Remaining'] PDSMDData.RETeaTbl['RRemainingLevel4']
Ct Detail Other Area	CtDetail['Ct Detail Other Area'] PDSMDData.RETeaTbl['OtherArea']
Ct Detail Other Degree	CtDetail['Ct Detail Other Degree'] PDSMDData.RETeaTbl['OtherDegree']
Ct Detail Other Name	CtDetail['Ct Detail Other Name'] PDSMDData.RETeaTbl['OtherName']
Ct Detail Other Year Completed	CtDetail['Ct Detail Other Year Completed'] PDSMDData.RETeaTbl['OtherYearComp']
Ct Detail Other Years	CtDetail['Ct Detail Other Years'] PDSMDData.RETeaTbl['OtherYears']
Ct Detail Position	CtDetail['Ct Detail Position'] PDSMDData.RETeaTbl['Position']
Ct Detail Remarks	CtDetail['Ct Detail Remarks'] PDSMDData.RETeaTbl['Remarks']
Ct Detail Renew Calc Cert	CtDetail['Ct Detail Renew Calc Cert'] PDSMDData.RETeaTbl['CalcCertRenew']
Ct Detail Renewed Certification	CtDetail['Ct Detail Renewed Certification'] PDSMDData.RETeaTbl['RCertRenew']
Ct Detail Renewed Date Certification	CtDetail['Ct Detail Renewed Date Certification'] PDSMDData.RETeaTbl['RCertDateRenew']
Ct Detail Renewed Remaining	CtDetail['Ct Detail Renewed Remaining'] PDSMDData.RETeaTbl['RRemainingRenew']
Ct Further Ed Alternate Course	CtFurtherEd['Ct Further Ed Alternate Course'] PDSMDData.RETeaFedTbl['AltCourse']
Ct Further Ed Certification Level	CtFurtherEd['Ct Further Ed Certification Level'] PDSMDData.RETeaFedTbl['CertLevel']
Ct Further Ed Course	CtFurtherEd['Ct Further Ed Course'] PDSMDData.RETeaFedTbl['Course']
Ct Further Ed Date	CtFurtherEd['Ct Further Ed Date'] PDSMDData.RETeaFedTbl['FedDate']
Ct Further Ed Instructor	CtFurtherEd['Ct Further Ed Instructor'] PDSMDData.RETeaFedTbl['Instructor']
Ct Further Ed Note	CtFurtherEd['Ct Further Ed Note'] PDSMDData.RETeaFedTbl['Note']
Ct Further Ed Place	CtFurtherEd['Ct Further Ed Place'] PDSMDData.RETeaFedTbl['Place']
Ct Further Ed Req Elect Other	CtFurtherEd['Ct Further Ed Req Elect Other'] PDSMDData.RETeaFedTbl['ReqElectOther']
Ct Further Ed Requirement Met	CtFurtherEd['Ct Further Ed Requirement Met'] PDSMDData.RETeaFedTbl['ReqMet']
Ct Further Ed Units	CtFurtherEd['Ct Further Ed Units'] PDSMDData.RETeaFedTbl['Units']
Ct Perm Rec Date	CtPermRec['Ct Perm Rec Date'] PDSMDData.RETeaPRTbl['PRDate']
Ct Perm Rec Grade	CtPermRec['Ct Perm Rec Grade'] PDSMDData.RETeaPRTbl['Grade']
Ct Perm Rec Position	CtPermRec['Ct Perm Rec Position'] PDSMDData.RETeaPRTbl['Position']
Ct Perm Rec Session Name	CtPermRec['Ct Perm Rec Session Name'] PDSMDData.RETeaPRTbl['SessionName']
Ct Sched Days Absent	CtSched['Ct Sched Days Absent'] PDSMDData.RETeaSchedTbl['DaysAbsent']
Ct Sched Days Present	CtSched['Ct Sched Days Present'] PDSMDData.RETeaSchedTbl['DaysPresent']
Ct Sched Order	CtSched['Ct Sched Order'] PDSMDData.RETeaSchedTbl['Order']
Ct Sched Position	CtSched['Ct Sched Position'] PDSMDData.RETeaSchedTbl['Position']
Ct Sched Session Name	CtSched['Ct Sched Session Name'] PDSMDData.RETeaSchedTbl['SessionName']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ct Sched Unique ID	CtSched['Ct Sched Unique ID'] PDSMDData.RETeaSchedTbl['SchedRecNum']
E Mail Description	EMail['E Mail Description'] PDSMDData.MemEMailTbl['EMailDesc']
E Mail E Mail Address	EMail['E Mail E Mail Address'] PDSMDData.MemEMailTbl['EMailAddress']
E Mail Prefers E Mail	EMail['E Mail Prefers E Mail'] PDSMDData.MemEMailTbl['EMailOverMail']
Enrollment Enrollment Date	Enrollment['Enrollment Enrollment Date'] PDSMDData.MemREENrTbl['REENrDate']
Enrollment Enrollment Year	Enrollment['Enrollment Enrollment Year'] PDSMDData.MemREENrTbl['REENrYear']
Enrollment Grade	Enrollment['Enrollment Grade'] PDSMDData.MemREENrTbl['EnrGrade']
Fam Address 1	Fam['Fam Address 1'] PDSMDData.FamTbl['Address1']
Fam Address 2	Fam['Fam Address 2'] PDSMDData.FamTbl['Address2']
Fam Address Remarks	Fam['Fam Address Remarks'] PDSMDData.FamTbl['AddressRemarks']
Fam Address Block *	Fam['Fam Address Block'] PDSMDData.FamTbl['AddressBlock']
Fam Address Changed	Fam['Fam Address Changed'] PDSMDData.FamTbl['AddressChanged']
Fam All Member Names	Fam['Fam All Member Names'] PDSMDData.FamTbl['AllMemberNames']
Fam Alt. Address Day to End	Fam['Fam Alt. Address Day to End'] PDSMDData.FamTbl['AlternateEndDay']
Fam Alt. Address Day to Start	Fam['Fam Alt. Address Day to Start'] PDSMDData.FamTbl['AlternateStartDay']
Fam Alt. Address Month to End	Fam['Fam Alt. Address Month to End'] PDSMDData.FamTbl['AlternateEndMonth']
Fam Alt. Address Month to Start	Fam['Fam Alt. Address Month to Start'] PDSMDData.FamTbl['AlternateStartMonth']
Fam Alternate Address is Certified	Fam['Fam Alternate Address is Certified'] PDSMDData.FamTbl['AlternateCertified']
Fam Alternate Address Line 1	Fam['Fam Alternate Address Line 1'] PDSMDData.FamTbl['AlternateAddress1']
Fam Alternate Address Line 2	Fam['Fam Alternate Address Line 2'] PDSMDData.FamTbl['AlternateAddress2']
Fam Alternate Carrier Route	Fam['Fam Alternate Carrier Route'] PDSMDData.FamTbl['AlternateCarrierRoute']
Fam Alternate DP	Fam['Fam Alternate DP'] PDSMDData.FamTbl['AlternateDP']
Fam Alternate E Mail	Fam['Fam Alternate E Mail'] PDSMDData.FamTbl['AlternateEMail']
Fam Alternate Zip	Fam['Fam Alternate Zip'] PDSMDData.FamTbl['AlternateZip']
Fam Alternate Address Block *	Fam['Fam Alternate Address Block'] PDSMDData.FamTbl['AlternateAddressBlock']
Fam Alternate City	Fam['Fam Alternate City'] PDSMDData.FamTbl['AlternateCity']
Fam Alternate Period	Fam['Fam Alternate Period'] PDSMDData.FamTbl['AlternatePeriod']
Fam Alt Prefers E Mail	Fam['Fam Alt Prefers E Mail'] PDSMDData.FamTbl['AltEMailOverMail']
Fam Blank1 *	Fam['Fam Blank1'] PDSMDData.FamTbl['Blank1']
Fam Blank2 *	Fam['Fam Blank2'] PDSMDData.FamTbl['Blank2']
Fam City	Fam['Fam City'] PDSMDData.FamTbl['City']
Fam City Only	Fam['Fam City Only'] PDSMDData.FamTbl['CityOnly']
Fam Confidential Remarks	Fam['Fam Confidential Remarks'] PDSMDData.FamTbl['ConfRemarks']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Date Alt. Addr Changed	Fam['Fam Date Alt. Addr Changed'] PDSMDData.FamTbl['AlternateChanged']
Fam Date Alternate Addr Certified	Fam['Fam Date Alternate Addr Certified'] PDSMDData.FamTbl['AlternateCertDate']
Fam Date Changed	Fam['Fam Date Changed'] PDSMDData.FamTbl['DateChanged']
Fam Date Created	Fam['Fam Date Created'] PDSMDData.FamTbl['DateCreated']
Fam Date Funds Changed	Fam['Fam Date Funds Changed'] PDSMDData.FamTbl['FundDateChanged']
Fam Date Mailing Addr Certified	Fam['Fam Date Mailing Addr Certified'] PDSMDData.FamTbl['MailingCertDate']
Fam Date Mailing Addr Changed	Fam['Fam Date Mailing Addr Changed'] PDSMDData.FamTbl['MailingChanged']
Fam Date Name Changed	Fam['Fam Date Name Changed'] PDSMDData.FamTbl['NameChanged']
Fam Date of Formation Registration	Fam['Fam Date of Formation Registration'] PDSMDData.FamTbl['REDateRegistered']
Fam Date Phone Changed	Fam['Fam Date Phone Changed'] PDSMDData.FamTbl['PhoneChanged']
Fam Date Street Addr Certified	Fam['Fam Date Street Addr Certified'] PDSMDData.FamTbl['StreetCertDate']
Fam Date Street Addr Changed	Fam['Fam Date Street Addr Changed'] PDSMDData.FamTbl['StreetChanged']
Fam E Mail	Fam['Fam E Mail'] PDSMDData.FamTbl['CurrentEMail']
Fam Family Status	Fam['Fam Family Status'] PDSMDData.FamTbl['FamilyStatus']
Fam First Name	Fam['Fam First Name'] PDSMDData.FamTbl['FirstName']
Fam Formal Sal	Fam['Fam Formal Sal'] PDSMDData.FamTbl['CurrentFormalSal']
Fam General Remarks	Fam['Fam General Remarks'] PDSMDData.FamTbl['GenRemarks']
Fam Geog. Area	Fam['Fam Geog. Area'] PDSMDData.FamTbl['AreaNumber']
Fam ID Number	Fam['Fam ID Number'] PDSMDData.FamTbl['CurrentIDNumber']
Fam Inactive	Fam['Fam Inactive'] PDSMDData.FamTbl['Inactive']
Fam Inactive Members	Fam['Fam Inactive Members'] PDSMDData.FamTbl['InactiveMembers']
Fam Informal Sal	Fam['Fam Informal Sal'] PDSMDData.FamTbl['CurrentInformalSal']
Fam Last Name	Fam['Fam Last Name'] PDSMDData.FamTbl['LastName']
Fam Letter Language	Fam['Fam Letter Language'] PDSMDData.FamTbl['LetterLanguage']
Fam License Name	Fam['Fam License Name'] PDSMDData.FamTbl['LicenseName']
Fam Mailing Address is Certified	Fam['Fam Mailing Address is Certified'] PDSMDData.FamTbl['MailingCertified']
Fam Mailing Address Line 1	Fam['Fam Mailing Address Line 1'] PDSMDData.FamTbl['MailingAddress1']
Fam Mailing Address Line 2	Fam['Fam Mailing Address Line 2'] PDSMDData.FamTbl['MailingAddress2']
Fam Mailing C/O Name	Fam['Fam Mailing C/O Name'] PDSMDData.FamTbl['MailingNameCO']
Fam Mailing Carrier Route	Fam['Fam Mailing Carrier Route'] PDSMDData.FamTbl['MailingCarrierRoute']
Fam Mailing DP	Fam['Fam Mailing DP'] PDSMDData.FamTbl['MailingDP']
Fam Mailing Zip	Fam['Fam Mailing Zip'] PDSMDData.FamTbl['MailingZip']
Fam Mailing Address Block *	Fam['Fam Mailing Address Block'] PDSMDData.FamTbl['MailingAddressBlock']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Mailing City	Fam['Fam Mailing City'] PDSDMData.FamTbl['MailingCity']
Fam Mailing Name	Fam['Fam Mailing Name'] PDSDMData.FamTbl['CurrentMailingName']
Fam Mailing Name Block *	Fam['Fam Mailing Name Block'] PDSDMData.FamTbl['MailingNameBlock']
Fam Member Names	Fam['Fam Member Names'] PDSDMData.FamTbl['MemberNames']
Fam Name Format 1 ( Mr. & Mrs. John Smith, Jr.)	Fam['Fam Name Format 1'] PDSDMData.FamTbl['NameFormat1']
Fam Name Format 10 (nickname of head)	Fam['Fam Name Format 10'] PDSDMData.FamTbl['NameFormat10']
Fam Name Format 11 (nickname of spouse)	Fam['Fam Name Format 11'] PDSDMData.FamTbl['NameFormat11']
Fam Name Format 12 ( Jack & Dot)	Fam['Fam Name Format 12'] PDSDMData.FamTbl['NameFormat12']
Fam Name Format 13 ( John & Mary Smith, Jr.)	Fam['Fam Name Format 13'] PDSDMData.FamTbl['NameFormat13']
Fam Name Format 2 ( Mr. & Mrs. John & Mary Smith, Jr.)	Fam['Fam Name Format 2'] PDSDMData.FamTbl['NameFormat2']
Fam Name Format 3 ( SMITH, JR., John & Mary)	Fam['Fam Name Format 3'] PDSDMData.FamTbl['NameFormat3']
Fam Name Format 4 ( Mr. & Mrs. Smith)	Fam['Fam Name Format 4'] PDSDMData.FamTbl['NameFormat4']
Fam Name Format 5 (first name)	Fam['Fam Name Format 5'] PDSDMData.FamTbl['NameFormat5']
Fam Name Format 6 (spouse name)	Fam['Fam Name Format 6'] PDSDMData.FamTbl['NameFormat6']
Fam Name Format 7 ( Smith, Jr.)	Fam['Fam Name Format 7'] PDSDMData.FamTbl['NameFormat7']
Fam Name Format 8 ( Mr. & Mrs.)	Fam['Fam Name Format 8'] PDSDMData.FamTbl['NameFormat8']
Fam Name Format 9 ( John & Mary)	Fam['Fam Name Format 9'] PDSDMData.FamTbl['NameFormat9']
Fam Name	Fam['Fam Name'] PDSDMData.FamTbl['CurrentName']
Fam Num All Children	Fam['Fam Num All Children'] PDSDMData.FamTbl['NumAllChild']
Fam Num All Members	Fam['Fam Num All Members'] PDSDMData.FamTbl['NumAllMem']
Fam Number Of Children	Fam['Fam Number Of Children'] PDSDMData.FamTbl['NumOfChildren']
Fam Number Of Students	Fam['Fam Number Of Students'] PDSDMData.FamTbl['NumOfStudents']
Fam Num Children	Fam['Fam Num Children'] PDSDMData.FamTbl['NumChild']
Fam Num Members	Fam['Fam Num Members'] PDSDMData.FamTbl['NumMem']
Fam Orig ID Number *	Fam['Fam Orig ID Number'] PDSDMData.FamTbl['IDNumber']
Fam Orig Mailing Name *	Fam['Fam Orig Mailing Name'] PDSDMData.FamTbl['MailingName']
Fam Orig Formal Sal *	Fam['Fam Orig Formal Sal'] PDSDMData.FamTbl['FormalSal']
Fam Orig Informal Sal *	Fam['Fam Orig Informal Sal'] PDSDMData.FamTbl['InformalSal']
Fam Orig Name *	Fam['Fam Orig Name'] PDSDMData.FamTbl['Name']
Fam Parish	Fam['Fam Parish'] PDSDMData.FamTbl['Church']
Fam Phone List *	Fam['Fam Phone List'] PDSDMData.FamTbl['PhoneList']
Fam Picture	Fam['Fam Picture'] PDSDMData.FamTbl['PictureFile']
Fam Prefers E Mail	Fam['Fam Prefers E Mail'] PDSDMData.FamTbl['CurrentEMailOverMail']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Raw Title *	Fam['Fam Raw Title'] PDSDMData.FamTbl['RawTitle']
Fam Send Mail to Alt. Address	Fam['Fam Send Mail to Alt. Address'] PDSDMData.FamTbl['SendMailToAlternate']
Fam Spouse First	Fam['Fam Spouse First'] PDSDMData.FamTbl['SpouseFirst']
Fam Spouse Last	Fam['Fam Spouse Last'] PDSDMData.FamTbl['SpouseLast']
Fam Spouse Raw Title *	Fam['Fam Spouse Raw Title'] PDSDMData.FamTbl['SpouseRawTitle']
Fam Spouse Title	Fam['Fam Spouse Title'] PDSDMData.FamTbl['SpouseTitle']
Fam State Only	Fam['Fam State Only'] PDSDMData.FamTbl['StateOnly']
Fam Street Address is Certified	Fam['Fam Street Address is Certified'] PDSDMData.FamTbl['StreetCertified']
Fam Street Address Line 1	Fam['Fam Street Address Line 1'] PDSDMData.FamTbl['StreetAddress1']
Fam Street Address Line 2	Fam['Fam Street Address Line 2'] PDSDMData.FamTbl['StreetAddress2']
Fam Street Carrier Route	Fam['Fam Street Carrier Route'] PDSDMData.FamTbl['StreetCarrierRoute']
Fam Street City	Fam['Fam Street City'] PDSDMData.FamTbl['StreetCity']
Fam Street DP	Fam['Fam Street DP'] PDSDMData.FamTbl['StreetDP']
Fam Street E Mail	Fam['Fam Street E Mail'] PDSDMData.FamTbl['EMail']
Fam Street Zip	Fam['Fam Street Zip'] PDSDMData.FamTbl['StreetZip']
Fam Street Address Block *	Fam['Fam Street Address Block'] PDSDMData.FamTbl['StreetAddressBlock']
Fam Street Prefers E Mail	Fam['Fam Street Prefers E Mail'] PDSDMData.FamTbl['EMailOverMail']
Fam Suffix	Fam['Fam Suffix'] PDSDMData.FamTbl['Suffix']
Fam Title	Fam['Fam Title'] PDSDMData.FamTbl['Title']
Fam Unique ID *	Fam['Fam Unique ID'] PDSDMData.FamTbl['FamRecNum']
Fam Use Alt	Fam['Fam Use Alt'] PDSDMData.FamTbl['UseAlt']
Fam Use E Mail	Fam['Fam Use E Mail'] PDSDMData.FamTbl['UseEMail']
Fam X Phone List *	Fam['Fam X Phone List'] PDSDMData.FamTbl['XPhoneList']
Fam Zip	Fam['Fam Zip'] PDSDMData.FamTbl['Zip']
Fam Zip DP	Fam['Fam Zip DP'] PDSDMData.FamTbl['ZipDP']
Fam Zip Prefix	Fam['Fam Zip Prefix'] PDSDMData.FamTbl['ZipPrefix']
Fam Zip Route	Fam['Fam Zip Route'] PDSDMData.FamTbl['ZipRoute']
Fam Keyword Description	FamKeyword['Fam Keyword Description'] PDSDMData.REFamKWTbl['Description']
Fam Letters Date	FamLetters['Fam Letters Date'] PDSDMData.FamLetTbl['Date']
Fam Letters Description	FamLetters['Fam Letters Description'] PDSDMData.FamLetTbl['Name']
Fam Letters Note	FamLetters['Fam Letters Note'] PDSDMData.FamLetTbl['Note']
Fam Letters Type	FamLetters['Fam Letters Type'] PDSDMData.FamLetTbl['Type']
Fam Phone Number	FamPhone['Fam Phone Number'] PDSDMData.FamPhoneTbl['Number']



## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fam Phone Type	FamPhone['Fam Phone Type'] PDSDMData.FamPhoneTbl['PhoneType']
Fam Phone Unlisted	FamPhone['Fam Phone Unlisted'] PDSDMData.FamPhoneTbl['Unlisted']
Fund Fund Keyword 1	Fund['Fund Fund Keyword 1'] PDSDMData.FamFundTbl['Keyword1']
Fund Fund Keyword 2	Fund['Fund Fund Keyword 2'] PDSDMData.FamFundTbl['Keyword2']
Fund Fund Number	Fund['Fund Fund Number'] PDSDMData.FamFundTbl['FDFund']
Fund Fund Year	Fund['Fund Fund Year'] PDSDMData.FamFundTbl['FDYear']
Fund Fund Identifier	Fund['Fund Fund Identifier'] PDSDMData.FamFundTbl['FundIdentifier']
Fund Bill Address Line 1	FundBill['Fund Bill Address Line 1'] PDSDMData.FamFundBillTbl['Address1']
Fund Bill Address Line 2	FundBill['Fund Bill Address Line 2'] PDSDMData.FamFundBillTbl['Address2']
Fund Bill Address Block *	FundBill['Fund Bill Address Block'] PDSDMData.FamFundBillTbl['AddressBlock']
Fund Bill C/O a Member	FundBill['Fund Bill C/O a Member'] PDSDMData.FamFundBillTbl['COMember']
Fund Bill Carrier Route	FundBill['Fund Bill Carrier Route'] PDSDMData.FamFundBillTbl['CarrierRoute']
Fund Bill Certified	FundBill['Fund Bill Certified'] PDSDMData.FamFundBillTbl['Certified']
Fund Bill Changed	FundBill['Fund Bill Changed'] PDSDMData.FamFundBillTbl['Changed']
Fund Bill City/State	FundBill['Fund Bill City/State'] PDSDMData.FamFundBillTbl['CityState']
Fund Bill Date Certified	FundBill['Fund Bill Date Certified'] PDSDMData.FamFundBillTbl['CertDate']
Fund Bill DP	FundBill['Fund Bill DP'] PDSDMData.FamFundBillTbl['DP']
Fund Bill E Mail	FundBill['Fund Bill E Mail'] PDSDMData.FamFundBillTbl['EMail']
Fund Bill End Day	FundBill['Fund Bill End Day'] PDSDMData.FamFundBillTbl['EndDay']
Fund Bill End Month	FundBill['Fund Bill End Month'] PDSDMData.FamFundBillTbl['EndMonth']
Fund Bill Formal Sal	FundBill['Fund Bill Formal Sal'] PDSDMData.FamFundBillTbl['FormalSal']
Fund Bill Informal Sal	FundBill['Fund Bill Informal Sal'] PDSDMData.FamFundBillTbl['InformalSal']
Fund Bill Mailing Name	FundBill['Fund Bill Mailing Name'] PDSDMData.FamFundBillTbl['MailingName']
Fund Bill Member	FundBill['Fund Bill Member'] PDSDMData.FamFundBillTbl['Member']
Fund Bill Name	FundBill['Fund Bill Name'] PDSDMData.FamFundBillTbl['Name']
Fund Bill Phone	FundBill['Fund Bill Phone'] PDSDMData.FamFundBillTbl['Phone']
Fund Bill Prefers E Mail	FundBill['Fund Bill Prefers E Mail'] PDSDMData.FamFundBillTbl['EMailOverMail']
Fund Bill Relationship	FundBill['Fund Bill Relationship'] PDSDMData.FamFundBillTbl['Relationship']
Fund Bill Start Day	FundBill['Fund Bill Start Day'] PDSDMData.FamFundBillTbl['StartDay']
Fund Bill Start Month	FundBill['Fund Bill Start Month'] PDSDMData.FamFundBillTbl['StartMonth']
Fund Bill Statements	FundBill['Fund Bill Statements'] PDSDMData.FamFundBillTbl['BillingTypeName']
Fund Bill Zip	FundBill['Fund Bill Zip'] PDSDMData.FamFundBillTbl['Zip']
Fund Bill Zip Prefix	FundBill['Fund Bill Zip Prefix'] PDSDMData.FamFundBillTbl['ZipPrefix']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fund EFT Activity	FundEFT['Fund EFT Activity'] PDSDMData.FamFundEFTTbl['Activity']
Fund EFT CC Type Name	FundEFT['Fund EFT CC Type Name'] PDSDMData.FamFundEFTTbl['CCTypeName']
Fund EFT EFT Type	FundEFT['Fund EFT EFT Type'] PDSDMData.FamFundEFTTbl['EFTType']
Fund EFT EFT Type Name	FundEFT['Fund EFT EFT Type Name'] PDSDMData.FamFundEFTTbl['EFTTypeName']
Fund Hist Activity	FundHist['Fund Hist Activity'] PDSDMData.FamFundHistTbl['FEType']
Fund Hist Amount	FundHist['Fund Hist Amount'] PDSDMData.FamFundHistTbl['FEAmt']
Fund Hist Batch Number	FundHist['Fund Hist Batch Number'] PDSDMData.FamFundHistTbl['FEBatch']
Fund Hist Check Number	FundHist['Fund Hist Check Number'] PDSDMData.FamFundHistTbl['FEChk']
Fund Hist Comment	FundHist['Fund Hist Comment'] PDSDMData.FamFundHistTbl['FEComment']
Fund Hist Date	FundHist['Fund Hist Date'] PDSDMData.FamFundHistTbl['FEDate']
Fund Hist Member	FundHist['Fund Hist Member'] PDSDMData.FamFundHistTbl['Member']
Fund Rate Activity	FundRate['Fund Rate Activity'] PDSDMData.FamFundRateTbl['Activity']
Fund Rate Associate with Member	FundRate['Fund Rate Associate with Member'] PDSDMData.FamFundRateTbl['FDUseMem']
Fund Rate End Date	FundRate['Fund Rate End Date'] PDSDMData.FamFundRateTbl['FDEndDate']
Fund Rate Last Rate	FundRate['Fund Rate Last Rate'] PDSDMData.FamFundRateTbl['LastRate']
Fund Rate Last Total	FundRate['Fund Rate Last Total'] PDSDMData.FamFundRateTbl['LastTotal']
Fund Rate Member Name	FundRate['Fund Rate Member Name'] PDSDMData.FamFundRateTbl['MemberName']
Fund Rate Rate	FundRate['Fund Rate Rate'] PDSDMData.FamFundRateTbl['FDRate']
Fund Rate Start Date	FundRate['Fund Rate Start Date'] PDSDMData.FamFundRateTbl['FDStartDate']
Fund Rate Terms	FundRate['Fund Rate Terms'] PDSDMData.FamFundRateTbl['Terms']
Fund Rate Terms Period	FundRate['Fund Rate Terms Period'] PDSDMData.FamFundRateTbl['FDPeriod']
Fund Rate Total	FundRate['Fund Rate Total'] PDSDMData.FamFundRateTbl['FDTTotal']
Fund Rate Use EFT	FundRate['Fund Rate Use EFT'] PDSDMData.FamFundRateTbl['UseEFT']
Fund Totals Accum Delinq Amt 1 (31 or More Days)	FundTotals['Fund Totals Accum Delinq Amt 1'] PDSDMCalc.FundTotalsTbl['AccumDelAmt1']
Fund Totals Accum Delinq Amt 2 (61 or More Days)	FundTotals['Fund Totals Accum Delinq Amt 2'] PDSDMCalc.FundTotalsTbl['AccumDelAmt2']
Fund Totals Accum Delinq Amt 3 (91 or More Days)	FundTotals['Fund Totals Accum Delinq Amt 3'] PDSDMCalc.FundTotalsTbl['AccumDelAmt3']
Fund Totals Delinq Amt 1 (31 to 60 Days)	FundTotals['Fund Totals Delinq Amt 1'] PDSDMCalc.FundTotalsTbl['DelAmt1']
Fund Totals Delinq Amt 2 (61 to 90 Days)	FundTotals['Fund Totals Delinq Amt 2'] PDSDMCalc.FundTotalsTbl['DelAmt2']
Fund Totals Delinq Amt 3 (91 or More Days)	FundTotals['Fund Totals Delinq Amt 3'] PDSDMCalc.FundTotalsTbl['DelAmt3']
Fund Totals Grand Total Balance	FundTotals['Fund Totals Grand Total Balance'] PDSDMCalc.FundTotalsTbl['GrandTotalBalance']
Fund Totals Grand Total Credit	FundTotals['Fund Totals Grand Total Credit'] PDSDMCalc.FundTotalsTbl['GrandTotalCredit']
Fund Totals Grand Total Deductible	FundTotals['Fund Totals Grand Total Deductible'] PDSDMCalc.FundTotalsTbl['GrandTotalDeductible']
Fund Totals Grand Total Due	FundTotals['Fund Totals Grand Total Due'] PDSDMCalc.FundTotalsTbl['GrandTotalDue']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fund Totals Grand Total Gifts	FundTotals['Fund Totals Grand Total Gifts'] PDSDMCalc.FundTotalsTbl['GrandTotalGifts']
Fund Totals Grand Total Non Cash	FundTotals['Fund Totals Grand Total Non Cash'] PDSDMCalc.FundTotalsTbl['GrandTotalNonCash']
Fund Totals Grand Total Non Ded	FundTotals['Fund Totals Grand Total Non Ded'] PDSDMCalc.FundTotalsTbl['GrandTotalNonDed']
Fund Totals Grand Total Paid	FundTotals['Fund Totals Grand Total Paid'] PDSDMCalc.FundTotalsTbl['GrandTotalPaid']
Fund Totals Grand Total Pledged	FundTotals['Fund Totals Grand Total Pledged'] PDSDMCalc.FundTotalsTbl['GrandTotalPledged']
Fund Totals Grand Total Refund	FundTotals['Fund Totals Grand Total Refund'] PDSDMCalc.FundTotalsTbl['GrandTotalRefund']
Fund Totals Grand Total Total Paid	FundTotals['Fund Totals Grand Total Total Paid'] PDSDMCalc.FundTotalsTbl['GrandTotalTotalPaid']
Fund Totals Has Quid Pro Quo	FundTotals['Fund Totals Has Quid Pro Quo'] PDSDMCalc.FundTotalsTbl['HasQuidProQuo']
Fund Totals Lastest Payment Date	FundTotals['Fund Totals Lastest Payment Date'] PDSDMCalc.FundTotalsTbl['LastestPaymentDate']
Fund Totals Payment Percentage	FundTotals['Fund Totals Payment Percentage'] PDSDMCalc.FundTotalsTbl['PaymentPercentage']
Fund Totals Progress Index	FundTotals['Fund Totals Progress Index'] PDSDMCalc.FundTotalsTbl['ProgressIndex']
Fund Totals Progress Pct Amt	FundTotals['Fund Totals Progress Pct Amt'] PDSDMCalc.FundTotalsTbl['ProgressPctAmt']
Fund Totals Progress Pct Time	FundTotals['Fund Totals Progress Pct Time'] PDSDMCalc.FundTotalsTbl['ProgressPctTime']
Fund Totals Recap Balance	FundTotals['Fund Totals Recap Balance'] PDSDMCalc.FundTotalsTbl['RecapBalance']
Fund Totals Recap Credit	FundTotals['Fund Totals Recap Credit'] PDSDMCalc.FundTotalsTbl['RecapCredit']
Fund Totals Recap Deductible	FundTotals['Fund Totals Recap Deductible'] PDSDMCalc.FundTotalsTbl['RecapDeductible']
Fund Totals Recap Due	FundTotals['Fund Totals Recap Due'] PDSDMCalc.FundTotalsTbl['RecapDue']
Fund Totals Recap Gifts	FundTotals['Fund Totals Recap Gifts'] PDSDMCalc.FundTotalsTbl['RecapGifts']
Fund Totals Recap Non Cash	FundTotals['Fund Totals Recap Non Cash'] PDSDMCalc.FundTotalsTbl['RecapNonCash']
Fund Totals Recap Non Ded	FundTotals['Fund Totals Recap Non Ded'] PDSDMCalc.FundTotalsTbl['RecapNonDed']
Fund Totals Recap Paid	FundTotals['Fund Totals Recap Paid'] PDSDMCalc.FundTotalsTbl['RecapPaid']
Fund Totals Recap Refund	FundTotals['Fund Totals Recap Refund'] PDSDMCalc.FundTotalsTbl['RecapRefund']
Fund Totals Recap Total Paid	FundTotals['Fund Totals Recap Total Paid'] PDSDMCalc.FundTotalsTbl['RecapTotalPaid']
Fund Totals Tax Statement Required	FundTotals['Fund Totals Tax Statement Required'] PDSDMCalc.FundTotalsTbl['TaxStatementRequired']
Keywords Description	Keywords['Keywords Description'] PDSDMData.MemKWTbl['Description']
Keywords Description	Keywords['Keywords Description'] PDSDMData.REMemKWTbl['Description']
Letter Date	Letter['Letter Date'] PDSDMData.MemLetTbl['Date']
Letter Description	Letter['Letter Description'] PDSDMData.MemLetTbl['Name']
Letter Note	Letter['Letter Note'] PDSDMData.MemLetTbl['Note']
Letter Type	Letter['Letter Type'] PDSDMData.MemLetTbl['Type']
Mem Age	Mem['Mem Age'] PDSDMData.MemTbl['Age']
Mem Confidential Remarks	Mem['Mem Confidential Remarks'] PDSDMData.MemTbl['ConfRemarks']
Mem Current Informal Sal	Mem['Mem Current Informal Sal'] PDSDMData.MemTbl['CurrentInformalSal']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Mem Date Changed	Mem['Mem Date Changed'] PDSMDData.MemTbl['DateChanged']
Mem Date Changed	Mem['Mem Date Changed'] PDSMDData.MemTbl['REDateChanged']
Mem Date Created	Mem['Mem Date Created'] PDSMDData.MemTbl['DateCreated']
Mem Date of Birth	Mem['Mem Date of Birth'] PDSMDData.MemTbl['DateOfBirth']
Mem Day of Birth	Mem['Mem Day of Birth'] PDSMDData.MemTbl['DayOfBirth']
Mem Deceased	Mem['Mem Deceased'] PDSMDData.MemTbl['Deceased']
Mem Different Last Name	Mem['Mem Different Last Name'] PDSMDData.MemTbl['DifLastName']
Mem Ethnicity	Mem['Mem Ethnicity'] PDSMDData.MemTbl['Ethnicity']
Mem Family Name	Mem['Mem Family Name'] PDSMDData.MemTbl['FamilyName']
Mem Fam Unique ID *	Mem['Mem Fam Unique ID'] PDSMDData.MemTbl['FamRecNum']
Mem First Name	Mem['Mem First Name'] PDSMDData.MemTbl['FirstName']
Mem Formal Sal	Mem['Mem Formal Sal'] PDSMDData.MemTbl['FormalSal']
Mem Formal Sal	Mem['Mem Formal Sal'] PDSMDData.MemTbl['CurrentFormalSal']
Mem Formation Member	Mem['Mem Formation Member'] PDSMDData.MemTbl['REMember1']
Mem Full Maiden Name *	Mem['Mem Full Maiden Name'] PDSMDData.MemTbl['FullMaidenName']
Mem Gender	Mem['Mem Gender'] PDSMDData.MemTbl['Gender']
Mem General Remarks	Mem['Mem General Remarks'] PDSMDData.MemTbl['GenRemarks']
Mem Grade	Mem['Mem Grade'] PDSMDData.MemTbl['Grade']
Mem ID Number	Mem['Mem ID Number'] PDSMDData.MemTbl['REIDNumber']
Mem Inactive	Mem['Mem Inactive'] PDSMDData.MemTbl['Inactive']
Mem Informal Sal	Mem['Mem Informal Sal'] PDSMDData.MemTbl['InformalSal']
Mem Is Catechist	Mem['Mem Is Catechist'] PDSMDData.MemTbl['Teacher']
Mem Is Parent	Mem['Mem Is Parent'] PDSMDData.MemTbl['Parent']
Mem Is Student	Mem['Mem Is Student'] PDSMDData.MemTbl['Student']
Mem Language	Mem['Mem Language'] PDSMDData.MemTbl['Language']
Mem Last Name	Mem['Mem Last Name'] PDSMDData.MemTbl['LastName']
Mem Letter Language	Mem['Mem Letter Language'] PDSMDData.MemTbl['LetterLanguage']
Mem Location	Mem['Mem Location'] PDSMDData.MemTbl['Location']
Mem Maiden Name	Mem['Mem Maiden Name'] PDSMDData.MemTbl['MaidenName']
Mem Mailing Name	Mem['Mem Mailing Name'] PDSMDData.MemTbl['CurrentMailingName']
Mem Marital Status	Mem['Mem Marital Status'] PDSMDData.MemTbl['MaritalStatus']
Mem Month Num of Birth	Mem['Mem Month Num of Birth'] PDSMDData.MemTbl['MonthNumofBirth']
Mem Month of Birth	Mem['Mem Month of Birth'] PDSMDData.MemTbl['MonthOfBirth']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Mem Name Format 1 ( Mr. John Smith, Jr.)	Mem['Mem Name Format 1'] PDSMDData.MemTbl['NameFormat1']
Mem Name Format 10 (maiden name)	Mem['Mem Name Format 10'] PDSMDData.MemTbl['NameFormat10']
Mem Name Format 11 ( Jack's)	Mem['Mem Name Format 11'] PDSMDData.MemTbl['NameFormat11']
Mem Name Format 2 ( Mr. Jack Smith, Jr.)	Mem['Mem Name Format 2'] PDSMDData.MemTbl['NameFormat2']
Mem Name Format 3 ( SMITH, JR., John)	Mem['Mem Name Format 3'] PDSMDData.MemTbl['NameFormat3']
Mem Name Format 4 ( Mr. Smith)	Mem['Mem Name Format 4'] PDSMDData.MemTbl['NameFormat4']
Mem Name Format 5 ( John)	Mem['Mem Name Format 5'] PDSMDData.MemTbl['NameFormat5']
Mem Name Format 6 (nickname)	Mem['Mem Name Format 6'] PDSMDData.MemTbl['NameFormat6']
Mem Name Format 7 ( Smith, Jr.)	Mem['Mem Name Format 7'] PDSMDData.MemTbl['NameFormat7']
Mem Name Format 8 ( Mr.)	Mem['Mem Name Format 8'] PDSMDData.MemTbl['NameFormat8']
Mem Name Format 9 ( Jack Smith)	Mem['Mem Name Format 9'] PDSMDData.MemTbl['NameFormat9']
Mem Name	Mem['Mem Name'] PDSMDData.MemTbl['Name']
Mem Nickname	Mem['Mem Nickname'] PDSMDData.MemTbl['Nickname']
Mem Occupation	Mem['Mem Occupation'] PDSMDData.MemTbl['UserKW3']
Mem Oldest Child	Mem['Mem Oldest Child'] PDSMDData.MemTbl['OldestChild']
Mem Phone List *	Mem['Mem Phone List'] PDSMDData.MemTbl['PhoneList']
Mem Picture	Mem['Mem Picture'] PDSMDData.MemTbl['PictureFile']
Mem Raw Title *	Mem['Mem Raw Title'] PDSMDData.MemTbl['RawTitle']
Mem Relationship	Mem['Mem Relationship'] PDSMDData.MemTbl['Relationship']
Mem Religion	Mem['Mem Religion'] PDSMDData.MemTbl['UserKW1']
Mem School	Mem['Mem School'] PDSMDData.MemTbl['UserKW4']
Mem Suffix	Mem['Mem Suffix'] PDSMDData.MemTbl['Suffix']
Mem Title	Mem['Mem Title'] PDSMDData.MemTbl['Title']
Mem Type Number *	Mem['Mem Type Number'] PDSMDData.MemTbl['MemberType']
Mem Type	Mem['Mem Type'] PDSMDData.MemTbl['Type']
Mem Unique ID *	Mem['Mem Unique ID'] PDSMDData.MemTbl['MemRecNum']
Mem Volunteer List	Mem['Mem Volunteer List'] PDSMDData.MemTbl['VolunteerList']
Mem X Phone List *	Mem['Mem X Phone List'] PDSMDData.MemTbl['XPhoneList']
Mem Year of Birth	Mem['Mem Year of Birth'] PDSMDData.MemTbl['YearOfBirth']
Mem Youngest Child	Mem['Mem Youngest Child'] PDSMDData.MemTbl['YoungestChild']
Ministries End Date	Ministries['Ministries End Date'] PDSMDData.MemMinTbl['EndDate']
Ministries Ministry	Ministries['Ministries Ministry'] PDSMDData.MemMinTbl['MinName']
Ministries Start Date	Ministries['Ministries Start Date'] PDSMDData.MemMinTbl['StartDate']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ministries Status	Ministries['Ministries Status'] PDSDMData.MemMinTbl['MinStatusName']
Other Req Date	OtherReq['Other Req Date'] PDSDMData.MemReqTbl['ReqDate']
Other Req Description	OtherReq['Other Req Description'] PDSDMData.MemReqTbl['Description']
Other Req Note	OtherReq['Other Req Note'] PDSDMData.MemReqTbl['ReqNote']
Other Req Result Name	OtherReq['Other Req Result Name'] PDSDMData.MemReqTbl['ResultName']
Perm Rec Absent	PermRec['Perm Rec Absent'] PDSDMData.MemREPRTbl['PRAbsent']
Perm Rec Comment	PermRec['Perm Rec Comment'] PDSDMData.MemREPRTbl['PRComment']
Perm Rec Grade	PermRec['Perm Rec Grade'] PDSDMData.MemREPRTbl['Grade']
Perm Rec Pass	PermRec['Perm Rec Pass'] PDSDMData.MemREPRTbl['PRPass']
Perm Rec Present	PermRec['Perm Rec Present'] PDSDMData.MemREPRTbl['PRPresent']
Perm Rec Session Name	PermRec['Perm Rec Session Name'] PDSDMData.MemREPRTbl['SessionName']
Perm Rec Teacher	PermRec['Perm Rec Teacher'] PDSDMData.MemREPRTbl['PRTeacher']
Perm Rec Year	PermRec['Perm Rec Year'] PDSDMData.MemREPRTbl['PRYear']
Phone Number	Phone['Phone Number'] PDSDMData.MemPhoneTbl['Number']
Phone Phone Type	Phone['Phone Phone Type'] PDSDMData.MemPhoneTbl['PhoneType']
Phone Unlisted	Phone['Phone Unlisted'] PDSDMData.MemPhoneTbl['Unlisted']
Prep Class Name	Prep['Prep Class Name'] PDSDMData.MemREPrepTbl['ClassName']
Prep Complete	Prep['Prep Complete'] PDSDMData.MemREPrepTbl['Complete']
Prep Date	Prep['Prep Date'] PDSDMData.MemREPrepTbl['ClassDate']
Prep Notes	Prep['Prep Notes'] PDSDMData.MemREPrepTbl['Notes']
Prep Num. Sessions Attended	Prep['Prep Num. Sessions Attended'] PDSDMData.MemREPrepTbl['SessionsAttended']
Sac Baptism Addl Status	Sac['Sac Baptism Addl Status'] PDSDMData.MemSacLookTbl['Date1Addl']
Sac Baptism Baptismal Name	Sac['Sac Baptism Baptismal Name'] PDSDMData.MemSacLookTbl['Date1Extra']
Sac Baptism Date	Sac['Sac Baptism Date'] PDSDMData.MemSacLookTbl['Date1Date']
Sac Baptism Day of Baptism	Sac['Sac Baptism Day of Baptism'] PDSDMData.MemSacLookTbl['Date1DayOfDate1']
Sac Baptism Month Num of Baptism	Sac['Sac Baptism Month Num of Baptism'] PDSDMData.MemSacLookTbl['Date1MonthNumofDate1']
Sac Baptism Month of Baptism	Sac['Sac Baptism Month of Baptism'] PDSDMData.MemSacLookTbl['Date1MonthofDate1']
Sac Baptism Notes	Sac['Sac Baptism Notes'] PDSDMData.MemSacLookTbl['Date1Notes']
Sac Baptism Performed By	Sac['Sac Baptism Performed By'] PDSDMData.MemSacLookTbl['Date1PerformedBy']
Sac Baptism Performed	Sac['Sac Baptism Performed'] PDSDMData.MemSacLookTbl['Date1Perf']
Sac Baptism Place	Sac['Sac Baptism Place'] PDSDMData.MemSacLookTbl['Date1PlaceBlock']
Sac Baptism Sponsor List	Sac['Sac Baptism Sponsor List'] PDSDMData.MemSacLookTbl['Date1SponsorList']
Sac Baptism Status	Sac['Sac Baptism Status'] PDSDMData.MemSacLookTbl['Date1Status']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Sac Baptism Year of Baptism	Sac['Sac Baptism Year of Baptism'] PDSDMData.MemSacLookTbl['Date1YearofDate1']
Sac Confirm Addl Status	Sac['Sac Confirm Addl Status'] PDSDMData.MemSacLookTbl['Date2Addl']
Sac Confirm Confirmation Name	Sac['Sac Confirm Confirmation Name'] PDSDMData.MemSacLookTbl['Date2Extra']
Sac Confirm Date	Sac['Sac Confirm Date'] PDSDMData.MemSacLookTbl['Date2Date']
Sac Confirm Day of Confirm	Sac['Sac Confirm Day of Confirm'] PDSDMData.MemSacLookTbl['Date2DayofDate2']
Sac Confirm Month Num of Confirm	Sac['Sac Confirm Month Num of Confirm'] PDSDMData.MemSacLookTbl['Date2MonthNumofDate2']
Sac Confirm Month of Confirm	Sac['Sac Confirm Month of Confirm'] PDSDMData.MemSacLookTbl['Date2MonthofDate2']
Sac Confirm Notes	Sac['Sac Confirm Notes'] PDSDMData.MemSacLookTbl['Date2Notes']
Sac Confirm Performed By	Sac['Sac Confirm Performed By'] PDSDMData.MemSacLookTbl['Date2PerformedBy']
Sac Confirm Performed	Sac['Sac Confirm Performed'] PDSDMData.MemSacLookTbl['Date2Perf']
Sac Confirm Place	Sac['Sac Confirm Place'] PDSDMData.MemSacLookTbl['Date2PlaceBlock']
Sac Confirm Sponsor List	Sac['Sac Confirm Sponsor List'] PDSDMData.MemSacLookTbl['Date2SponsorList']
Sac Confirm Status	Sac['Sac Confirm Status'] PDSDMData.MemSacLookTbl['Date2Status']
Sac Confirm Year of Confirm	Sac['Sac Confirm Year of Confirm'] PDSDMData.MemSacLookTbl['Date2YearofDate2']
Sac First Comm Addl Status	Sac['Sac First Comm Addl Status'] PDSDMData.MemSacLookTbl['Date4Addl']
Sac First Comm Date	Sac['Sac First Comm Date'] PDSDMData.MemSacLookTbl['Date4Date']
Sac First Comm Day of First Comm	Sac['Sac First Comm Day of First Comm'] PDSDMData.MemSacLookTbl['Date4DayofDate4']
Sac First Comm Extra Info	Sac['Sac First Comm Extra Info'] PDSDMData.MemSacLookTbl['Date4Extra']
Sac First Comm Month Num of First Comm	Sac['Sac First Comm Month Num of First Comm'] PDSDMData.MemSacLookTbl['Date4MonthNumofDate4']
Sac First Comm Month of First Comm	Sac['Sac First Comm Month of First Comm'] PDSDMData.MemSacLookTbl['Date4MonthofDate4']
Sac First Comm Notes	Sac['Sac First Comm Notes'] PDSDMData.MemSacLookTbl['Date4Notes']
Sac First Comm Performed By	Sac['Sac First Comm Performed By'] PDSDMData.MemSacLookTbl['Date4PerformedBy']
Sac First Comm Performed	Sac['Sac First Comm Performed'] PDSDMData.MemSacLookTbl['Date4Perf']
Sac First Comm Place	Sac['Sac First Comm Place'] PDSDMData.MemSacLookTbl['Date4PlaceBlock']
Sac First Comm Sponsor List	Sac['Sac First Comm Sponsor List'] PDSDMData.MemSacLookTbl['Date4SponsorList']
Sac First Comm Status	Sac['Sac First Comm Status'] PDSDMData.MemSacLookTbl['Date4Status']
Sac First Comm Year of First Comm	Sac['Sac First Comm Year of First Comm'] PDSDMData.MemSacLookTbl['Date4YearofDate4']
Sac Marriage Addl Status	Sac['Sac Marriage Addl Status'] PDSDMData.MemSacLookTbl['Date5Addl']
Sac Marriage Date	Sac['Sac Marriage Date'] PDSDMData.MemSacLookTbl['Date5Date']
Sac Marriage Day of Marriage	Sac['Sac Marriage Day of Marriage'] PDSDMData.MemSacLookTbl['Date5DayofDate5']
Sac Marriage Month Num of Marriage	Sac['Sac Marriage Month Num of Marriage'] PDSDMData.MemSacLookTbl['Date5MonthNumofDate5']
Sac Marriage Month of Marriage	Sac['Sac Marriage Month of Marriage'] PDSDMData.MemSacLookTbl['Date5MonthofDate5']
Sac Marriage Notes	Sac['Sac Marriage Notes'] PDSDMData.MemSacLookTbl['Date5Notes']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Sac Marriage Performed By	Sac['Sac Marriage Performed By'] PDSDMData.MemSacLookTbl['Date5PerformedBy']
Sac Marriage Performed	Sac['Sac Marriage Performed'] PDSDMData.MemSacLookTbl['Date5Perf']
Sac Marriage Place	Sac['Sac Marriage Place'] PDSDMData.MemSacLookTbl['Date5PlaceBlock']
Sac Marriage Sponsor List	Sac['Sac Marriage Sponsor List'] PDSDMData.MemSacLookTbl['Date5SponsorList']
Sac Marriage Spouse Name	Sac['Sac Marriage Spouse Name'] PDSDMData.MemSacLookTbl['Date5Extra']
Sac Marriage Status	Sac['Sac Marriage Status'] PDSDMData.MemSacLookTbl['Date5Status']
Sac Marriage Year of Marriage	Sac['Sac Marriage Year of Marriage'] PDSDMData.MemSacLookTbl['Date5YearofDate5']
Sac Reconciliati Addl Status	Sac['Sac Reconciliati Addl Status'] PDSDMData.MemSacLookTbl['Date3Addl']
Sac Reconciliati Date	Sac['Sac Reconciliati Date'] PDSDMData.MemSacLookTbl['Date3Date']
Sac Reconciliati Day of Reconciliati	Sac['Sac Reconciliati Day of Reconciliati'] PDSDMData.MemSacLookTbl['Date3DayofDate3']
Sac Reconciliati Extra Info	Sac['Sac Reconciliati Extra Info'] PDSDMData.MemSacLookTbl['Date3Extra']
Sac Reconciliati Month Num of Reconciliati	Sac['Sac Reconciliati Month Num of Reconciliati'] PDSDMData.MemSacLookTbl['Date3MonthNumofDate3']
Sac Reconciliati Month of Reconciliati	Sac['Sac Reconciliati Month of Reconciliati'] PDSDMData.MemSacLookTbl['Date3MonthofDate3']
Sac Reconciliati Notes	Sac['Sac Reconciliati Notes'] PDSDMData.MemSacLookTbl['Date3Notes']
Sac Reconciliati Performed By	Sac['Sac Reconciliati Performed By'] PDSDMData.MemSacLookTbl['Date3PerformedBy']
Sac Reconciliati Performed	Sac['Sac Reconciliati Performed'] PDSDMData.MemSacLookTbl['Date3Perf']
Sac Reconciliati Place	Sac['Sac Reconciliati Place'] PDSDMData.MemSacLookTbl['Date3PlaceBlock']
Sac Reconciliati Sponsor List	Sac['Sac Reconciliati Sponsor List'] PDSDMData.MemSacLookTbl['Date3SponsorList']
Sac Reconciliati Status	Sac['Sac Reconciliati Status'] PDSDMData.MemSacLookTbl['Date3Status']
Sac Reconciliati Year of Reconciliati	Sac['Sac Reconciliati Year of Reconciliati'] PDSDMData.MemSacLookTbl['Date3YearofDate3']
Sac Extra Confidential Notes	SacExtra['Sac Extra Confidential Notes'] PDSDMData.MemSacTbl['ConfNotes']
Sac Extra Extra Information	SacExtra['Sac Extra Extra Information'] PDSDMData.MemSacTbl['ExtraInfo']
Sac Extra General Notes	SacExtra['Sac Extra General Notes'] PDSDMData.MemSacTbl['GenNotes']
Sac Extra Performed By	SacExtra['Sac Extra Performed By'] PDSDMData.MemSacTbl['PerformedBy']
Sac General Birth Place	SacGeneral['Sac General Birth Place'] PDSDMData.AskTbl['BirthPlace']
Sac General Date Changed	SacGeneral['Sac General Date Changed'] PDSDMData.AskTbl['DateChanged']
Sac General Date Created	SacGeneral['Sac General Date Created'] PDSDMData.AskTbl['DateCreated']
Sac General Father's Name	SacGeneral['Sac General Father's Name'] PDSDMData.AskTbl['FatherName']
Sac General Mother's Maiden Name	SacGeneral['Sac General Mother's Maiden Name'] PDSDMData.AskTbl['MothersMaiden']
Sac General Mother's Name	SacGeneral['Sac General Mother's Name'] PDSDMData.AskTbl['MotherName']
Sac List Addl Name	SacList['Sac List Addl Name'] PDSDMData.MemDatesTbl['AddlName']
Sac List Addl Status	SacList['Sac List Addl Status'] PDSDMData.MemDatesTbl['AddlStatus']
Sac List Address	SacList['Sac List Address'] PDSDMData.MemDatesTbl['DateAddress']



## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code
Sac List City State	SacList['Sac List City State'] PDSDMData.MemDatesTbl['DateCityState']
Sac List Date	SacList['Sac List Date'] PDSDMData.MemDatesTbl['Date']
Sac List Day	SacList['Sac List Day'] PDSDMData.MemDatesTbl['Day']
Sac List E Mail	SacList['Sac List E Mail'] PDSDMData.MemDatesTbl['DateEMail']
Sac List Month Num	SacList['Sac List Month Num'] PDSDMData.MemDatesTbl['MonthNum']
Sac List Month	SacList['Sac List Month'] PDSDMData.MemDatesTbl['Month']
Sac List Name	SacList['Sac List Name'] PDSDMData.MemDatesTbl['DateName']
Sac List Order Num	SacList['Sac List Order Num'] PDSDMData.MemDatesTbl['OrderNum']
Sac List Place	SacList['Sac List Place'] PDSDMData.MemDatesTbl['DatePlace']
Sac List Status	SacList['Sac List Status'] PDSDMData.MemDatesTbl['Status']
Sac List Status Name	SacList['Sac List Status Name'] PDSDMData.MemDatesTbl['StatusName']
Sac List Use E Mail	SacList['Sac List Use E Mail'] PDSDMData.MemDatesTbl['DateUseEMail']
Sac List Year	SacList['Sac List Year'] PDSDMData.MemDatesTbl['Year']
Sac List Zip	SacList['Sac List Zip'] PDSDMData.MemDatesTbl['DateZip']
Sac Sponsor Address	SacSponsor['Sac Sponsor Address'] PDSDMData.MemSponsTbl['Address']
Sac Sponsor City State	SacSponsor['Sac Sponsor City State'] PDSDMData.MemSponsTbl['CityState']
Sac Sponsor Formal Sal	SacSponsor['Sac Sponsor Formal Sal'] PDSDMData.MemSponsTbl['FormalSal']
Sac Sponsor Informal Sal	SacSponsor['Sac Sponsor Informal Sal'] PDSDMData.MemSponsTbl['InformalSal']
Sac Sponsor Mailing Name	SacSponsor['Sac Sponsor Mailing Name'] PDSDMData.MemSponsTbl['MailingName']
Sac Sponsor Name	SacSponsor['Sac Sponsor Name'] PDSDMData.MemSponsTbl['Name']
Sac Sponsor Phone	SacSponsor['Sac Sponsor Phone'] PDSDMData.MemSponsTbl['Phone']
Sac Sponsor Sponsor Type	SacSponsor['Sac Sponsor Sponsor Type'] PDSDMData.MemSponsTbl['SponsorType']
Sac Sponsor Unlisted	SacSponsor['Sac Sponsor Unlisted'] PDSDMData.MemSponsTbl['Unl']
Sac Sponsor Zip Code	SacSponsor['Sac Sponsor Zip Code'] PDSDMData.MemSponsTbl['ZipCode']
Sched Days Absent	Sched['Sched Days Absent'] PDSDMData.MemRESHdTbl['DaysAbsent']
Sched Days Present	Sched['Sched Days Present'] PDSDMData.MemRESHdTbl['DaysPresent']
Sched Passing	Sched['Sched Passing'] PDSDMData.MemRESHdTbl['Passing']
Sched Session Name	Sched['Sched Session Name'] PDSDMData.MemRESHdTbl['SessionName']
Sched Unique ID	Sched['Sched Unique ID'] PDSDMData.MemRESHdTbl['MemRESHdRec']
Serv Ret Date	ServRet['Serv Ret Date'] PDSDMData.MemREServRetTbl['Date']
Serv Ret Hours	ServRet['Serv Ret Hours'] PDSDMData.MemREServRetTbl['Hours']
Serv Ret Service/Retreat	ServRet['Serv Ret Service/Retreat'] PDSDMData.MemREServRetTbl['ServiceRetreat']
Serv Ret Rem Health Problems	ServRetRem['Serv Ret Rem Health Problems'] PDSDMData.MemREtbl['UserRemark1']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code
Serv Ret Rem Misc. Info	ServRetRem['Serv Ret Rem Misc. Info'] PDSDMData.MemREtbl['UserRemark3']
Serv Ret Rem Other Conditions	ServRetRem['Serv Ret Rem Other Conditions'] PDSDMData.MemREtbl['UserRemark2']
Serv Ret Rem Remarks	ServRetRem['Serv Ret Rem Remarks'] PDSDMData.MemREtbl['UserRemark4']
Student Age	Student['Student Age'] PDSDMData.MemTbl['Age']
Student Confidential Remarks	Student['Student Confidential Remarks'] PDSDMData.MemTbl['ConfRemarks']
Student Current Informal Sal	Student['Student Current Informal Sal'] PDSDMData.MemTbl['CurrentInformalSal']
Student Date Changed	Student['Student Date Changed'] PDSDMData.MemTbl['DateChanged']
Student Date Changed	Student['Student Date Changed'] PDSDMData.MemTbl['REDateChanged']
Student Date Created	Student['Student Date Created'] PDSDMData.MemTbl['DateCreated']
Student Date of Birth	Student['Student Date of Birth'] PDSDMData.MemTbl['DateOfBirth']
Student Day of Birth	Student['Student Day of Birth'] PDSDMData.MemTbl['DayOfBirth']
Student Deceased	Student['Student Deceased'] PDSDMData.MemTbl['Deceased']
Student Different Last Name	Student['Student Different Last Name'] PDSDMData.MemTbl['DifLastName']
Student Ethnicity	Student['Student Ethnicity'] PDSDMData.MemTbl['Ethnicity']
Student Family Name	Student['Student Family Name'] PDSDMData.MemTbl['FamilyName']
Student Fam Unique ID *	Student['Student Fam Unique ID'] PDSDMData.MemTbl['FamRecNum']
Student First Name	Student['Student First Name'] PDSDMData.MemTbl['FirstName']
Student Formal Sal	Student['Student Formal Sal'] PDSDMData.MemTbl['FormalSal']
Student Formal Sal	Student['Student Formal Sal'] PDSDMData.MemTbl['CurrentFormalSal']
Student Formation Member	Student['Student Formation Member'] PDSDMData.MemTbl['REMember1']
Student Full Maiden Name *	Student['Student Full Maiden Name'] PDSDMData.MemTbl['FullMaidenName']
Student Gender	Student['Student Gender'] PDSDMData.MemTbl['Gender']
Student General Remarks	Student['Student General Remarks'] PDSDMData.MemTbl['GenRemarks']
Student Grade	Student['Student Grade'] PDSDMData.MemTbl['Grade']
Student ID Number	Student['Student ID Number'] PDSDMData.MemTbl['REIDNumber']
Student Inactive	Student['Student Inactive'] PDSDMData.MemTbl['Inactive']
Student Informal Sal	Student['Student Informal Sal'] PDSDMData.MemTbl['InformalSal']
Student Is Catechist	Student['Student Is Catechist'] PDSDMData.MemTbl['Teacher']
Student Is Parent	Student['Student Is Parent'] PDSDMData.MemTbl['Parent']
Student Is Student	Student['Student Is Student'] PDSDMData.MemTbl['Student']
Student Language	Student['Student Language'] PDSDMData.MemTbl['Language']
Student Last Name	Student['Student Last Name'] PDSDMData.MemTbl['LastName']
Student Letter Language	Student['Student Letter Language'] PDSDMData.MemTbl['LetterLanguage']

## Formation Office Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code
Student Location	Student['Student Location'] PDSMDData.MemTbl['Location']
Student Maiden Name	Student['Student Maiden Name'] PDSMDData.MemTbl['MaidenName']
Student Mailing Name	Student['Student Mailing Name'] PDSMDData.MemTbl['CurrentMailingName']
Student Marital Status	Student['Student Marital Status'] PDSMDData.MemTbl['MaritalStatus']
Student Month Num of Birth	Student['Student Month Num of Birth'] PDSMDData.MemTbl['MonthNumofBirth']
Student Month of Birth	Student['Student Month of Birth'] PDSMDData.MemTbl['MonthOfBirth']
Student Name Format 1 ( Mr. John Smith, Jr.)	Student['Student Name Format 1'] PDSMDData.MemTbl['NameFormat1']
Student Name Format 10 (maiden name)	Student['Student Name Format 10'] PDSMDData.MemTbl['NameFormat10']
Student Name Format 11 ( Jack's)	Student['Student Name Format 11'] PDSMDData.MemTbl['NameFormat11']
Student Name Format 2 ( Mr. Jack Smith, Jr.)	Student['Student Name Format 2'] PDSMDData.MemTbl['NameFormat2']
Student Name Format 3 ( SMITH, JR., John)	Student['Student Name Format 3'] PDSMDData.MemTbl['NameFormat3']
Student Name Format 4 ( Mr. Smith)	Student['Student Name Format 4'] PDSMDData.MemTbl['NameFormat4']
Student Name Format 5 ( John)	Student['Student Name Format 5'] PDSMDData.MemTbl['NameFormat5']
Student Name Format 6 (nickname)	Student['Student Name Format 6'] PDSMDData.MemTbl['NameFormat6']
Student Name Format 7 ( Smith, Jr.)	Student['Student Name Format 7'] PDSMDData.MemTbl['NameFormat7']
Student Name Format 8 ( Mr.)	Student['Student Name Format 8'] PDSMDData.MemTbl['NameFormat8']
Student Name Format 9 ( Jack Smith)	Student['Student Name Format 9'] PDSMDData.MemTbl['NameFormat9']
Student Name	Student['Student Name'] PDSMDData.MemTbl['Name']
Student Nickname	Student['Student Nickname'] PDSMDData.MemTbl['Nickname']
Student Occupation	Student['Student Occupation'] PDSMDData.MemTbl['UserKW3']
Student Oldest Child	Student['Student Oldest Child'] PDSMDData.MemTbl['OldestChild']
Student Phone List *	Student['Student Phone List'] PDSMDData.MemTbl['PhoneList']
Student Picture	Student['Student Picture'] PDSMDData.MemTbl['PictureFile']
Student Raw Title *	Student['Student Raw Title'] PDSMDData.MemTbl['RawTitle']
Student Relationship	Student['Student Relationship'] PDSMDData.MemTbl['Relationship']
Student Religion	Student['Student Religion'] PDSMDData.MemTbl['UserKW1']
Student School	Student['Student School'] PDSMDData.MemTbl['UserKW4']
Student Suffix	Student['Student Suffix'] PDSMDData.MemTbl['Suffix']
Student Title	Student['Student Title'] PDSMDData.MemTbl['Title']
Student Type Number *	Student['Student Type Number'] PDSMDData.MemTbl['MemberType']
Student Type	Student['Student Type'] PDSMDData.MemTbl['Type']
Student Unique ID *	Student['Student Unique ID'] PDSMDData.MemTbl['MemRecNum']
Student Volunteer List	Student['Student Volunteer List'] PDSMDData.MemTbl['VolunteerList']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code
Student X Phone List *	Student['Student X Phone List'] PDSMDData.MemTbl['XPhoneList']
Student Year of Birth	Student['Student Year of Birth'] PDSMDData.MemTbl['YearOfBirth']
Student Youngest Child	Student['Student Youngest Child'] PDSMDData.MemTbl['YoungestChild']
Talents End Date	Talents['Talents End Date'] PDSMDData.MemTalTbl['EndDate']
Talents Start Date	Talents['Talents Start Date'] PDSMDData.MemTalTbl['StartDate']
Talents Status	Talents['Talents Status'] PDSMDData.MemTalTbl['TalStatusName']
Talents Talent	Talents['Talents Talent'] PDSMDData.MemTalTbl['TalName']
Volunteer Note	Volunteer['Volunteer Note'] PDSMDData.MemREVolTbl['VolNote']
Volunteer Volunteer Area	Volunteer['Volunteer Volunteer Area'] PDSMDData.MemREVolTbl['VolunteerArea']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Aged 30 Days Amount	Aged['Aged 30 Days Amount'] DM.SourceAgedTbl['Aged30DaysAmt']
Aged 30 Days Amount	Aged['Aged 30 Days Amount'] DM.VenAgedTbl['Aged30DaysAmt']
Aged 30 Days Number	Aged['Aged 30 Days Number'] DM.SourceAgedTbl['Aged30DaysNum']
Aged 30 Days Number	Aged['Aged 30 Days Number'] DM.VenAgedTbl['Aged30DaysNum']
Aged 60 Days Amount	Aged['Aged 60 Days Amount'] DM.SourceAgedTbl['Aged60DaysAmt']
Aged 60 Days Amount	Aged['Aged 60 Days Amount'] DM.VenAgedTbl['Aged60DaysAmt']
Aged 60 Days Number	Aged['Aged 60 Days Number'] DM.SourceAgedTbl['Aged60DaysNum']
Aged 60 Days Number	Aged['Aged 60 Days Number'] DM.VenAgedTbl['Aged60DaysNum']
Aged 90 Days Amount	Aged['Aged 90 Days Amount'] DM.SourceAgedTbl['Aged90DaysAmt']
Aged 90 Days Amount	Aged['Aged 90 Days Amount'] DM.VenAgedTbl['Aged90DaysAmt']
Aged 90 Days Number	Aged['Aged 90 Days Number'] DM.SourceAgedTbl['Aged90DaysNum']
Aged 90 Days Number	Aged['Aged 90 Days Number'] DM.VenAgedTbl['Aged90DaysNum']
Aged Amount of Current	Aged['Aged Amount of Current'] DM.SourceAgedTbl['CurrentAmt']
Aged Amount of Current	Aged['Aged Amount of Current'] DM.VenAgedTbl['CurrentAmt']
Aged Number of Current	Aged['Aged Number of Current'] DM.SourceAgedTbl['CurrentNum']
Aged Number of Current	Aged['Aged Number of Current'] DM.VenAgedTbl['CurrentNum']
Aged Over 90 Days Amount	Aged['Aged Over 90 Days Amount'] DM.SourceAgedTbl['AgedOver90DaysAmt']
Aged Over 90 Days Amount	Aged['Aged Over 90 Days Amount'] DM.VenAgedTbl['AgedOver90DaysAmt']
Aged Over 90 Days Number	Aged['Aged Over 90 Days Number'] DM.SourceAgedTbl['AgedOver90DaysNum']
Aged Over 90 Days Number	Aged['Aged Over 90 Days Number'] DM.VenAgedTbl['AgedOver90DaysNum']
Aged Total Amount	Aged['Aged Total Amount'] DM.SourceAgedTbl['AgedTotalAmt']
Aged Total Amount	Aged['Aged Total Amount'] DM.VenAgedTbl['AgedTotalAmt']
Aged Total Number	Aged['Aged Total Number'] DM.SourceAgedTbl['AgedTotalNum']
Aged Total Number	Aged['Aged Total Number'] DM.VenAgedTbl['AgedTotalNum']
Asset Accumulated Depreciation Account Name	Asset['Asset Accumulated Depreciation Account Name'] DM.AssetTbl['AccAcctName']
Asset Accumulated Depreciation Account Number	Asset['Asset Accumulated Depreciation Account Number'] DM.AssetTbl['AccAcctNum']
Asset Category	Asset['Asset Category'] DM.AssetTbl['Category']
Asset Current Value	Asset['Asset Current Value'] DM.AssetTbl['CurrentValue']
Asset Date Changed	Asset['Asset Date Changed'] DM.AssetTbl['DateChanged']
Asset Date Created	Asset['Asset Date Created'] DM.AssetTbl['DateCreated']
Asset Depreciation Method	Asset['Asset Depreciation Method'] DM.AssetTbl['DeprMethod']
Asset Depr Next Amount	Asset['Asset Depr Next Amount'] DM.AssetTbl['DeprNewAmount']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Asset Depr Next Date	Asset['Asset Depr Next Date'] DM.AssetTbl['DeprNextDate']
Asset First Month Larger	Asset['Asset First Month Larger'] DM.AssetTbl['FirstMonthLarger']
Asset Fixed Asset Account Name	Asset['Asset Fixed Asset Account Name'] DM.AssetTbl['AssetAcctName']
Asset Fixed Asset Account Number	Asset['Asset Fixed Asset Account Number'] DM.AssetTbl['AssetAcctNum']
Asset Inactive	Asset['Asset Inactive'] DM.AssetTbl['Inactive']
Asset Last Depreciation Amount	Asset['Asset Last Depreciation Amount'] DM.AssetTbl['DeprLastAmount']
Asset Last Depreciation Date	Asset['Asset Last Depreciation Date'] DM.AssetTbl['DeprLastDate']
Asset Life Expectancy	Asset['Asset Life Expectancy'] DM.AssetTbl['LifeExpectance']
Asset Name of Fixed Asset	Asset['Asset Name of Fixed Asset'] DM.AssetTbl['Name']
Asset Purchase Date	Asset['Asset Purchase Date'] DM.AssetTbl['PurchaseDate']
Asset Purchase Price	Asset['Asset Purchase Price'] DM.AssetTbl['PurchasePrice']
Asset Remarks	Asset['Asset Remarks'] DM.AssetTbl['Remarks']
Asset Salvage Value	Asset['Asset Salvage Value'] DM.AssetTbl['SalvageValue']
Asset Serial Number	Asset['Asset Serial Number'] DM.AssetTbl['SerialNumber']
Asset Take Yearly	Asset['Asset Take Yearly'] DM.AssetTbl['TakeYearly']
Asset Total to Date Depreciation	Asset['Asset Total to Date Depreciation'] DM.AssetTbl['DeprTTD']
Asset Voided	Asset['Asset Voided'] DM.AssetTbl['Voided']
Asset Voided Or Reversed	Asset['Asset Voided Or Reversed'] DM.AssetTbl['VoidedOrReversed']
Asset Year to Date Depreciation	Asset['Asset Year to Date Depreciation'] DM.AssetTbl['DeprYTD']
COA Account Code ID	Coa['COA Account Code ID'] DM.COATbl['AcctCodeId']
COA Account Name	Coa['COA Account Name'] DM.COATbl['Name']
COA Account Number	Coa['COA Account Number'] DM.COATbl['Number']
COA Alternate Number	Coa['COA Alternate Number'] DM.COATbl['AlternateNumber']
COA Balance Sheet Column	Coa['COA Balance Sheet Column'] DM.COATbl['BalanceSheetColumn']
COA Beginning Balance	Coa['COA Beginning Balance'] DM.COATbl['BeginningBalance']
COA Blank Lines After	Coa['COA Blank Lines After'] DM.COATbl['BlankLines']
COA Category Name	Coa['COA Category Name'] DM.COATbl['Category']
COA Comments	Coa['COA Comments'] DM.COATbl['Comments']
COA Date Changed	Coa['COA Date Changed'] DM.COATbl['DateChanged']
COA Date Created	Coa['COA Date Created'] DM.COATbl['DateCreated']
COA Department ID	Coa['COA Department ID'] DM.COATbl['DeptId']
COA Department Name	Coa['COA Department Name'] DM.COATbl['DeptName']
COA Fiscal Date	Coa['COA Fiscal Date'] DM.COATbl['FiscalDate']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
COA Has Distribution	Coa['COA Has Distribution'] DM.COATbl['HasDistribution']
COA Inactive	Coa['COA Inactive'] DM.COATbl['Inactive']
COA Is Asset	Coa['COA Is Asset'] DM.COATbl['IsAsset']
COA Is Balance Sheet	Coa['COA Is Balance Sheet'] DM.COATbl['IsBalanceSheet']
COA Is Credit	Coa['COA Is Credit'] DM.COATbl['IsCredit']
COA Is Debit	Coa['COA Is Debit'] DM.COATbl['IsDebit']
COA Is Detail	Coa['COA Is Detail'] DM.COATbl['IsDetail']
COA Is Expense	Coa['COA Is Expense'] DM.COATbl['IsExpense']
COA Is Heading	Coa['COA Is Heading'] DM.COATbl['IsHeading']
COA Is Income Statement	Coa['COA Is Income Statement'] DM.COATbl['IsIncomeStatement']
COA Is Income	Coa['COA Is Income'] DM.COATbl['IsIncome']
COA Is Liability	Coa['COA Is Liability'] DM.COATbl['IsLiability']
COA Is Net	Coa['COA Is Net'] DM.COATbl['IsNet']
COA Is Prior Year Fund Bal	Coa['COA Is Prior Year Fund Bal'] DM.COATbl['IsPriorYearFundBal']
COA Is Retained Earnings	Coa['COA Is Retained Earnings'] DM.COATbl['IsRetainedEarnings']
COA Is Total	Coa['COA Is Total'] DM.COATbl['IsTotal']
COA Is Cash	Coa['COA Is Cash'] DM.COATbl['IsCash']
COA Master Account	Coa['COA Master Account'] DM.COATbl['MasterAccount']
COA New Page After	Coa['COA New Page After'] DM.COATbl['BlankPage']
COA Normal Balance	Coa['COA Normal Balance'] DM.COATbl['NormalBalance']
COA Report Type	Coa['COA Report Type'] DM.COATbl['ReportType']
COA Set Of Books	Coa['COA Set Of Books'] DM.COATbl['SetOfBooks']
COA Sub Account	Coa['COA Sub Account'] DM.COATbl['SubAccount']
COA Total Level	Coa['COA Total Level'] DM.COATbl['TotalLevel']
COA Type	Coa['COA Type'] DM.COATbl['Type']
COA Unique ID	Coa['COA Unique ID'] DM.COATbl['CoaRec']
COA Sub Trn Amount	CoaSubTrn['COA Sub Trn Amount'] DM.CoaTrnTbl['Amount']
COA Sub Trn Credit	CoaSubTrn['COA Sub Trn Credit'] DM.CoaTrnTbl['Credit']
COA Sub Trn Date	CoaSubTrn['COA Sub Trn Date'] DM.CoaTrnTbl['TDate']
COA Sub Trn Debit	CoaSubTrn['COA Sub Trn Debit'] DM.CoaTrnTbl['Debit']
COA Sub Trn Description	CoaSubTrn['COA Sub Trn Description'] DM.CoaTrnTbl['Description']
COA Sub Trn Voided	CoaSubTrn['COA Sub Trn Voided'] DM.CoaTrnTbl['Voided']
COA Sub Trn Voided Or Reversed	CoaSubTrn['COA Sub Trn Voided Or Reversed'] DM.CoaTrnTbl['VoidedOrReversed']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Dist Description	Dist['Dist Description'] DM.DistTbl['Description']
Dist Name	Dist['Dist Name'] DM.DistTbl['COAName']
Dist Number	Dist['Dist Number'] DM.DistTbl['COANumber']
Dist Percentage	Dist['Dist Percentage'] DM.DistTbl['Percentage']
Emp Date Birth	Emp['Emp Date Birth'] DMEmployee.tblEmployee['Birthday']
Emp Date Changed	Emp['Emp Date Changed'] DMEmployee.tblEmployee['DateChanged']
Emp Date Created	Emp['Emp Date Created'] DMEmployee.tblEmployee['DateCreated']
Emp Date First Hired	Emp['Emp Date First Hired'] DMEmployee.tblEmployee['HireDate']
Emp E Mail	Emp['Emp E Mail'] DMEmployee.tblEmployee['Email']
Emp First Name	Emp['Emp First Name'] DMEmployee.tblEmployee['FirstName']
Emp Gender	Emp['Emp Gender'] DMEmployee.tblEmployee['Gender']
Emp ID Number	Emp['Emp ID Number'] DMEmployee.tblEmployee['ID']
Emp Inactive	Emp['Emp Inactive'] DMEmployee.tblEmployee['Inactive']
Emp Last Name	Emp['Emp Last Name'] DMEmployee.tblEmployee['LastName']
Emp Mailing Name	Emp['Emp Mailing Name'] DMEmployee.tblEmployee['Mailing']
Emp Name	Emp['Emp Name'] DMEmployee.tblEmployee['Name']
Emp Phone List	Emp['Emp Phone List'] DMEmployee.tblEmployee['PhoneList']
Emp Picture	Emp['Emp Picture'] DMEmployee.tblEmployee['Picture']
Emp Remarks	Emp['Emp Remarks'] DMEmployee.tblEmployee['Remarks']
Emp Salutation	Emp['Emp Salutation'] DMEmployee.tblEmployee['Salutation']
Emp SSN	Emp['Emp SSN'] DMEmployee.tblEmployee['SSN']
Emp Suffix	Emp['Emp Suffix'] DMEmployee.tblEmployee['Suffix']
Emp Title	Emp['Emp Title'] DMEmployee.tblEmployee['Title']
Emp Web Page	Emp['Emp Web Page'] DMEmployee.tblEmployee['WebPage']
Emp1099 Income QTD	Emp1099['Emp1099 Income QTD'] DMEmployee.tblEmp1099['IncomeQTD']
Emp1099 Income YTD	Emp1099['Emp1099 Income YTD'] DMEmployee.tblEmp1099['IncomeYTD']
Emp1099 Is Contract	Emp1099['Emp1099 Is Contract'] DMEmployee.tblEmp1099['IsContract']
Emp1099 State 1 Abbr	Emp1099['Emp1099 State 1 Abbr'] DMEmployee.tblEmp1099['State1Abbr']
Emp1099 State 1 ID	Emp1099['Emp1099 State 1 ID'] DMEmployee.tblEmp1099['State1ID']
Emp1099 State 1 Income YTD	Emp1099['Emp1099 State 1 Income YTD'] DMEmployee.tblEmp1099['State1IncomeYTD']
Emp1099 State 1 Withholding YTD	Emp1099['Emp1099 State 1 Withholding YTD'] DMEmployee.tblEmp1099['State1WHYTD']
Emp1099 State 2 Abbr	Emp1099['Emp1099 State 2 Abbr'] DMEmployee.tblEmp1099['State2Abbr']
Emp1099 State 2 ID	Emp1099['Emp1099 State 2 ID'] DMEmployee.tblEmp1099['State2ID']



## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp1099 State 2 Income YTD	Emp1099['Emp1099 State 2 Income YTD'] DMEmployee.tblEmp1099['State2IncomeYTD']
Emp1099 State 2 Withholding YTD	Emp1099['Emp1099 State 2 Withholding YTD'] DMEmployee.tblEmp1099['State2WHYTD']
Emp1099 Withholding QTD	Emp1099['Emp1099 Withholding QTD'] DMEmployee.tblEmp1099['WithholdingQTD']
Emp1099 Withholding YTD	Emp1099['Emp1099 Withholding YTD'] DMEmployee.tblEmp1099['WithholdingYTD']
Emp Act Hist Comments	EmpActHist['Emp Act Hist Comments'] DMEmployee.tblActivity['Comments']
Emp Act Hist Date From	EmpActHist['Emp Act Hist Date From'] DMEmployee.tblActivity['From']
Emp Act Hist Date To	EmpActHist['Emp Act Hist Date To'] DMEmployee.tblActivity['To']
Emp Act Hist Name	EmpActHist['Emp Act Hist Name'] DMEmployee.tblActivity['lkActivity']
Emp Act Hist Units	EmpActHist['Emp Act Hist Units'] DMEmployee.tblActivity['Hours']
Emp Act Hist Year	EmpActHist['Emp Act Hist Year'] DMEmployee.tblActivity['Year']
Emp Activity Left	EmpActivity['Emp Activity Left'] DMEmployee.tblDays['calcLeft']
Emp Activity Max	EmpActivity['Emp Activity Max'] DMEmployee.tblDays['Max']
Emp Activity Name	EmpActivity['Emp Activity Name'] DMEmployee.tblDays['lkType']
Emp Activity Used	EmpActivity['Emp Activity Used'] DMEmployee.tblDays['Used']
Emp Activity Year	EmpActivity['Emp Activity Year'] DMEmployee.tblDays['Year']
Emp Address 1	EmpAddress['Emp Address 1'] DMEmployee.tblAddress['Address']
Emp Address 2	EmpAddress['Emp Address 2'] DMEmployee.tblAddress['Address2']
Emp Address Address Block	EmpAddress['Emp Address Address Block'] DMEmployee.tblAddress['AddressBlock']
Emp Address City/State	EmpAddress['Emp Address City/State'] DMEmployee.tblAddress['CityState']
Emp Address City/State/Zip	EmpAddress['Emp Address City/State/Zip'] DMEmployee.tblAddress['CityZip']
Emp Address City Only	EmpAddress['Emp Address City Only'] DMEmployee.tblAddress['CityOnly']
Emp Address State Only	EmpAddress['Emp Address State Only'] DMEmployee.tblAddress['StateOnly']
Emp Address Type	EmpAddress['Emp Address Type'] DMEmployee.tblAddress['AddressType']
Emp Address Zip	EmpAddress['Emp Address Zip'] DMEmployee.tblAddress['Zip']
Emp Benefit Amount	EmpBenefit['Emp Benefit Amount'] DMEmployee.tblBenefits['Amount']
Emp Benefit Comments	EmpBenefit['Emp Benefit Comments'] DMEmployee.tblBenefits['Comments']
Emp Benefit Name	EmpBenefit['Emp Benefit Name'] DMEmployee.tblBenefits['lkBenefit']
Emp Benefit Total Amount	EmpBenefit['Emp Benefit Total Amount'] DMEmployee.tblBenefits['AmountTotal']
Emp Benefit Year	EmpBenefit['Emp Benefit Year'] DMEmployee.tblBenefits['Year']
Emp Chk Amount	EmpChk['Emp Chk Amount'] DMEmployee.tblPrintChecks['Amount']
Emp Chk calc Regular Hours	EmpChk['Emp Chk calc Regular Hours'] DMEmployee.tblPrintChecks['calcRegularHours']
Emp Chk calc Void Amount	EmpChk['Emp Chk calc Void Amount'] DMEmployee.tblPrintChecks['calcVoidAmount']
Emp Chk Check~Date	EmpChk['Emp Chk Check~Date'] DMEmployee.tblPrintChecks['AtDate']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp Chk Deductions	EmpChk['Emp Chk Deductions'] DMEmployee.tblPrintChecks['Deductions']
Emp Chk Direct~Dep.	EmpChk['Emp Chk Direct~Dep.'] DMEmployee.tblPrintChecks['DirectDeposit']
Emp Chk End Period Date	EmpChk['Emp Chk End Period Date'] DMEmployee.tblPrintChecks['EndPeriodDate']
Emp Chk Gross	EmpChk['Emp Chk Gross'] DMEmployee.tblPrintChecks['Gross']
Emp Chk Local COA List	EmpChk['Emp Chk Local COA List'] DMEmployee.tblPrintChecks['LocalCoaList']
Emp Chk Memo	EmpChk['Emp Chk Memo'] DMEmployee.tblPrintChecks['Memo']
Emp Chk Number	EmpChk['Emp Chk Number'] DMEmployee.tblPrintChecks['Number']
Emp Chk Ovr1 Hours	EmpChk['Emp Chk Ovr1 Hours'] DMEmployee.tblPrintChecks['Ovr1Hours']
Emp Chk Ovr2 Hours	EmpChk['Emp Chk Ovr2 Hours'] DMEmployee.tblPrintChecks['Ovr2Hours']
Emp Chk Print~Now	EmpChk['Emp Chk Print~Now'] DMEmployee.tblPrintChecks['Done']
Emp Chk Print~Now	EmpChk['Emp Chk Print~Now'] DMEmployee.tblPrintChecks['PrintNow']
Emp Chk Regular Hours	EmpChk['Emp Chk Regular Hours'] DMEmployee.tblPrintChecks['RegularHours']
Emp Chk Remarks	EmpChk['Emp Chk Remarks'] DMEmployee.tblPrintChecks['Remarks']
Emp Chk Special Hours	EmpChk['Emp Chk Special Hours'] DMEmployee.tblPrintChecks['SpecialHours']
Emp Chk Start Period Date	EmpChk['Emp Chk Start Period Date'] DMEmployee.tblPrintChecks['StartPeriodDate']
Emp Chk State COA List	EmpChk['Emp Chk State COA List'] DMEmployee.tblPrintChecks['StateCoaList']
Emp Chk Total~Hours	EmpChk['Emp Chk Total~Hours'] DMEmployee.tblPrintChecks['TotalHours']
Emp Chk Trn Rec	EmpChk['Emp Chk Trn Rec'] DMEmployee.tblPrintChecks['TrnRec']
Emp Chk Void	EmpChk['Emp Chk Void'] DMEmployee.tblPrintChecks['Void']
Emp Emg Contact Mailing Name	EmpEmgContact['Emp Emg Contact Mailing Name'] DMEmployee.tblEmergency['Mailing']
Emp Emg Contact Name	EmpEmgContact['Emp Emg Contact Name'] DMEmployee.tblEmergency['Name']
Emp Emg Contact Phone List	EmpEmgContact['Emp Emg Contact Phone List'] DMEmployee.tblEmergency['PhoneList']
Emp Emg Contact Relation	EmpEmgContact['Emp Emg Contact Relation'] DMEmployee.tblEmergency['Relation']
Emp Emg Contact Remarks	EmpEmgContact['Emp Emg Contact Remarks'] DMEmployee.tblEmergency['Remarks']
Emp Emg Contact Salutation	EmpEmgContact['Emp Emg Contact Salutation'] DMEmployee.tblEmergency['Salutation']
Emp Emg Phone Number	EmpEmgPhone['Emp Emg Phone Number'] DMEmployee.tblEmgPhone['NNumber']
Emp Emg Phone Type	EmpEmgPhone['Emp Emg Phone Type'] DMEmployee.tblEmgPhone['PhoneType']
Emp Emg Phone Unlisted	EmpEmgPhone['Emp Emg Phone Unlisted'] DMEmployee.tblEmgPhone['Unlisted']
Emp Emg Phone X Number	EmpEmgPhone['Emp Emg Phone X Number'] DMEmployee.tblEmgPhone['XNumber']
Emp Evaluation By	EmpEvaluation['Emp Evaluation By'] DMEmployee.tblEvaluation['EvaluatedBy']
Emp Evaluation Date	EmpEvaluation['Emp Evaluation Date'] DMEmployee.tblEvaluation['Date']
Emp Evaluation Remarks	EmpEvaluation['Emp Evaluation Remarks'] DMEmployee.tblEvaluation['Comments']
Emp Job Annual Salary/Hourly Rate	EmpJob['Emp Job Annual Salary/Hourly Rate'] DMEmployee.tblJobs['Salary']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp Job Date Hired	EmpJob['Emp Job Date Hired'] DMEmployee.tblJobs['HireDate']
Emp Job Date Terminated	EmpJob['Emp Job Date Terminated'] DMEmployee.tblJobs['TerminationDate']
Emp Job Days Per Check	EmpJob['Emp Job Days Per Check'] DMEmployee.tblJobs['lkFreqEquivDays']
Emp Job Department	EmpJob['Emp Job Department'] DMEmployee.tblJobs['lkDepartment']
Emp Job Department/Position	EmpJob['Emp Job Department/Position'] DMEmployee.tblJobs['calcJob']
Emp Job Direct Deposit	EmpJob['Emp Job Direct Deposit'] DMEmployee.tblJobs['DirectDeposit']
Emp Job Employer	EmpJob['Emp Job Employer'] DMEmployee.tblJobs['lkEmployer']
Emp Job Equivalent Annual Salary	EmpJob['Emp Job Equivalent Annual Salary'] DMEmployee.tblJobs['EquivAnnualSalary']
Emp Job Equivalent Daily Rate	EmpJob['Emp Job Equivalent Daily Rate'] DMEmployee.tblJobs['EquivalentDR']
Emp Job Equivalent Days	EmpJob['Emp Job Equivalent Days'] DMEmployee.tblJobs['EquivDaysFloat']
Emp Job Equivalent Hourly Rate	EmpJob['Emp Job Equivalent Hourly Rate'] DMEmployee.tblJobs['EquivalentHR']
Emp Job Equivalent Hours	EmpJob['Emp Job Equivalent Hours'] DMEmployee.tblJobs['EquivHoursFloat']
Emp Job Federal Withholdings	EmpJob['Emp Job Federal Withholdings'] DMEmployee.tblJobs['FedWithholding']
Emp Job Filing Status	EmpJob['Emp Job Filing Status'] DMEmployee.tblJobs['lkFilingStatus']
Emp Job Frequency Description	EmpJob['Emp Job Frequency Description'] DMEmployee.tblJobs['lkFreqDescription']
Emp Job Frequency	EmpJob['Emp Job Frequency'] DMEmployee.tblJobs['lkFrequency']
Emp Job Gross Income	EmpJob['Emp Job Gross Income'] DMEmployee.tblJobs['calcGrossIncome']
Emp Job Hourly	EmpJob['Emp Job Hourly'] DMEmployee.tblJobs['PayTypeHourly']
Emp Job Hours Per Check	EmpJob['Emp Job Hours Per Check'] DMEmployee.tblJobs['lkFreqEquivHours']
Emp Job Inactive	EmpJob['Emp Job Inactive'] DMEmployee.tblJobs['Inactive']
Emp Job Last Check Date	EmpJob['Emp Job Last Check Date'] DMEmployee.tblJobs['LastCheckDate']
Emp Job Last Check Num	EmpJob['Emp Job Last Check Num'] DMEmployee.tblJobs['LastCheckNum']
Emp Job Last Check Value	EmpJob['Emp Job Last Check Value'] DMEmployee.tblJobs['LastCheckValue']
Emp Job Number of Payments	EmpJob['Emp Job Number of Payments'] DMEmployee.tblJobs['PayCounter']
Emp Job Other State Exemptions	EmpJob['Emp Job Other State Exemptions'] DMEmployee.tblJobs['OtherStateExempt']
Emp Job Pay Period	EmpJob['Emp Job Pay Period'] DMEmployee.tblJobs['lkPayPeriod']
Emp Job Personal Exemptions	EmpJob['Emp Job Personal Exemptions'] DMEmployee.tblJobs['PersonalExemptions']
Emp Job Position	EmpJob['Emp Job Position'] DMEmployee.tblJobs['lkPosition']
Emp Job Retirement Plan	EmpJob['Emp Job Retirement Plan'] DMEmployee.tblJobs['RetirementPlan']
Emp Job State Dependents	EmpJob['Emp Job State Dependents'] DMEmployee.tblJobs['StateDependents']
Emp Job Statutory Employee	EmpJob['Emp Job Statutory Employee'] DMEmployee.tblJobs['StatutoryEmployee']
Emp Job Termination Reason	EmpJob['Emp Job Termination Reason'] DMEmployee.tblJobs['TerminationReason']
Emp Job Third Party Sick Pay	EmpJob['Emp Job Third Party Sick Pay'] DMEmployee.tblJobs['ThirdPartySickPay']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp Job Hours 1 QTD	EmpJobHours['Emp Job Hours 1 QTD'] DMEmployee.tblTotalsHourly['QTD1']
Emp Job Hours 2 QTD	EmpJobHours['Emp Job Hours 2 QTD'] DMEmployee.tblTotalsHourly['QTD2']
Emp Job Hours 3 QTD	EmpJobHours['Emp Job Hours 3 QTD'] DMEmployee.tblTotalsHourly['QTD3']
Emp Job Hours 4 QTD	EmpJobHours['Emp Job Hours 4 QTD'] DMEmployee.tblTotalsHourly['QTD4']
Emp Job Hours Type	EmpJobHours['Emp Job Hours Type'] DMEmployee.tblTotalsHourly['Type']
Emp Job Hours Year	EmpJobHours['Emp Job Hours Year'] DMEmployee.tblTotalsHourly['Year']
Emp Job Hours YTD	EmpJobHours['Emp Job Hours YTD'] DMEmployee.tblTotalsHourly['calcYTD']
Emp Job Totals 1 QTD	EmpJobTotals['Emp Job Totals 1 QTD'] DMEmployee.tblTotals['QTD1']
Emp Job Totals 2 QTD	EmpJobTotals['Emp Job Totals 2 QTD'] DMEmployee.tblTotals['QTD2']
Emp Job Totals 3 QTD	EmpJobTotals['Emp Job Totals 3 QTD'] DMEmployee.tblTotals['QTD3']
Emp Job Totals 4 QTD	EmpJobTotals['Emp Job Totals 4 QTD'] DMEmployee.tblTotals['QTD4']
Emp Job Totals COA Name	EmpJobTotals['Emp Job Totals COA Name'] DMEmployee.tblTotals['AccName']
Emp Job Totals COA Number	EmpJobTotals['Emp Job Totals COA Number'] DMEmployee.tblTotals['lkAccNumber']
Emp Job Totals Kind	EmpJobTotals['Emp Job Totals Kind'] DMEmployee.tblTotals['lkKind']
Emp Job Totals Sub Type	EmpJobTotals['Emp Job Totals Sub Type'] DMEmployee.tblTotals['lkSubType']
Emp Job Totals Type	EmpJobTotals['Emp Job Totals Type'] DMEmployee.tblTotals['lkTypeAcc']
Emp Job Totals Year	EmpJobTotals['Emp Job Totals Year'] DMEmployee.tblTotals['Year']
Emp Job Totals YTD	EmpJobTotals['Emp Job Totals YTD'] DMEmployee.tblTotals['calcYTD']
Emp Job Wage COA Name	EmpJobWage['Emp Job Wage COA Name'] DMEmployee.tblDistribution['lkCOAName']
Emp Job Wage COA Number	EmpJobWage['Emp Job Wage COA Number'] DMEmployee.tblDistribution['lkCoaNumber']
Emp Job Wage Fixed Amount	EmpJobWage['Emp Job Wage Fixed Amount'] DMEmployee.tblDistribution['FixedAmt']
Emp Job Wage Frequency	EmpJobWage['Emp Job Wage Frequency'] DMEmployee.tblDistribution['lkFreqName']
Emp Job Wage Percent Amount	EmpJobWage['Emp Job Wage Percent Amount'] DMEmployee.tblDistribution['PctAmt']
Emp Job Wage Type	EmpJobWage['Emp Job Wage Type'] DMEmployee.tblDistribution['lkTypeName']
Employer Address 1	Employer['Employer Address 1'] DMEmployee.tblEmployer['Address']
Employer Address 2	Employer['Employer Address 2'] DMEmployee.tblEmployer['Address2']
Employer Address Block	Employer['Employer Address Block'] DMEmployee.tblEmployer['AddressBlock']
Employer City/State/Zip	Employer['Employer City/State/Zip'] DMEmployee.tblEmployer['CityZip']
Employer City Only	Employer['Employer City Only'] DMEmployee.tblEmployer['CityOnly']
Employer City State	Employer['Employer City State'] DMEmployee.tblEmployer['CityState']
Employer E Mail	Employer['Employer E Mail'] DMEmployee.tblEmployer['EMail']
Employer Federal ID Number	Employer['Employer Federal ID Number'] DMEmployee.tblEmployer['FederalIDNumber']
Employer Local ID Number	Employer['Employer Local ID Number'] DMEmployee.tblEmployer['LocalIDNumber']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Employer Local Name	Employer['Employer Local Name'] DMEmployee.tblEmployer['LocalName']
Employer Name	Employer['Employer Name'] DMEmployee.tblEmployer['Name']
Employer Remarks	Employer['Employer Remarks'] DMEmployee.tblEmployer['Remarks']
Employer SDIF Acc Number	Employer['Employer SDIF Acc Number'] DMEmployee.tblEmployer['SDIFAccNumber']
Employer SDIF Limit	Employer['Employer SDIF Limit'] DMEmployee.tblEmployer['SDIFLimit']
Employer SDIF Rate	Employer['Employer SDIF Rate'] DMEmployee.tblEmployer['SDIFRate']
Employer State ID Number	Employer['Employer State ID Number'] DMEmployee.tblEmployer['StateIDNumber']
Employer State Name	Employer['Employer State Name'] DMEmployee.tblEmployer['StateName']
Employer State Only	Employer['Employer State Only'] DMEmployee.tblEmployer['StateOnly']
Employer SUTA Acc Number	Employer['Employer SUTA Acc Number'] DMEmployee.tblEmployer['SUTAAccNumber']
Employer SUTA Limit	Employer['Employer SUTA Limit'] DMEmployee.tblEmployer['SUTALimit']
Employer SUTA Rate	Employer['Employer SUTA Rate'] DMEmployee.tblEmployer['SUTARate']
Employer Web Page	Employer['Employer Web Page'] DMEmployee.tblEmployer['WebPage']
Employer Zip	Employer['Employer Zip'] DMEmployee.tblEmployer['Zip']
Emp Phone Number	EmpPhone['Emp Phone Number'] DMEmployee.tblEmpPhone['NNumber']
Emp Phone Type	EmpPhone['Emp Phone Type'] DMEmployee.tblEmpPhone['PhoneType']
Emp Phone Unlisted	EmpPhone['Emp Phone Unlisted'] DMEmployee.tblEmpPhone['Unlisted']
Emp Phone X Number	EmpPhone['Emp Phone X Number'] DMEmployee.tblEmpPhone['XNumber']
Emp Sal Hist Date	EmpSalHist['Emp Sal Hist Date'] DMEmployee.tblHistory['Date']
Emp Sal Hist Description	EmpSalHist['Emp Sal Hist Description'] DMEmployee.tblHistory['Activity']
Emp Sal Hist From Amount	EmpSalHist['Emp Sal Hist From Amount'] DMEmployee.tblHistory['FromAmount']
Emp Sal Hist From Freq	EmpSalHist['Emp Sal Hist From Freq'] DMEmployee.tblHistory['FromFreq']
Emp Sal Hist To Freq	EmpSalHist['Emp Sal Hist To Freq'] DMEmployee.tblHistory['ToFreq']
Emp Sal Hist To Amount	EmpSalHist['Emp Sal Hist To Amount'] DMEmployee.tblHistory['ToAmount']
Emp W2 Box 12 A Code	EmpW2['Emp W2 Box 12 A Code'] DMEmployee.tblEmpW2['Box12ACode']
Emp W2 Box 12 A YTD	EmpW2['Emp W2 Box 12 A YTD'] DMEmployee.tblEmpW2['Box12AYTD']
Emp W2 Box 12 B Code	EmpW2['Emp W2 Box 12 B Code'] DMEmployee.tblEmpW2['Box12BCode']
Emp W2 Box 12 B YTD	EmpW2['Emp W2 Box 12 B YTD'] DMEmployee.tblEmpW2['Box12BYTD']
Emp W2 Box 12 C Code	EmpW2['Emp W2 Box 12 C Code'] DMEmployee.tblEmpW2['Box12CCode']
Emp W2 Box 12 C YTD	EmpW2['Emp W2 Box 12 C YTD'] DMEmployee.tblEmpW2['Box12CYTD']
Emp W2 Box 12 D Code	EmpW2['Emp W2 Box 12 D Code'] DMEmployee.tblEmpW2['Box12DCode']
Emp W2 Box 12 D YTD	EmpW2['Emp W2 Box 12 D YTD'] DMEmployee.tblEmpW2['Box12DYTD']
Emp W2 Box 14 A Name	EmpW2['Emp W2 Box 14 A Name'] DMEmployee.tblEmpW2['Box14AName']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp W2 Box 14 A YTD	EmpW2['Emp W2 Box 14 A YTD'] DMEmployee.tblEmpW2['Box14AYTD']
Emp W2 Box 14 B Name	EmpW2['Emp W2 Box 14 B Name'] DMEmployee.tblEmpW2['Box14BName']
Emp W2 Box 14 B YTD	EmpW2['Emp W2 Box 14 B YTD'] DMEmployee.tblEmpW2['Box14BYTD']
Emp W2 Box 14 C Name	EmpW2['Emp W2 Box 14 C Name'] DMEmployee.tblEmpW2['Box14CName']
Emp W2 Box 14 C YTD	EmpW2['Emp W2 Box 14 C YTD'] DMEmployee.tblEmpW2['Box14CYTD']
Emp W2 Box 14 D Name	EmpW2['Emp W2 Box 14 D Name'] DMEmployee.tblEmpW2['Box14DName']
Emp W2 Box 14 D YTD	EmpW2['Emp W2 Box 14 D YTD'] DMEmployee.tblEmpW2['Box14DYTD']
Emp W2 Deductions QTD	EmpW2['Emp W2 Deductions QTD'] DMEmployee.tblEmpW2['DeductionsQTD']
Emp W2 Deductions YTD	EmpW2['Emp W2 Deductions YTD'] DMEmployee.tblEmpW2['DeductionsYTD']
Emp W2 Dependent Care Benefits QTD	EmpW2['Emp W2 Dependent Care Benefits QTD'] DMEmployee.tblEmpW2['DepCareQTD']
Emp W2 Dependent Care Benefits YTD	EmpW2['Emp W2 Dependent Care Benefits YTD'] DMEmployee.tblEmpW2['DepCareYTD']
Emp W2 EIC Payments QTD	EmpW2['Emp W2 EIC Payments QTD'] DMEmployee.tblEmpW2['EICPaymentsQTD']
Emp W2 EIC Payments YTD	EmpW2['Emp W2 EIC Payments YTD'] DMEmployee.tblEmpW2['EICPaymentsYTD']
Emp W2 Federal Wage QTD	EmpW2['Emp W2 Federal Wage QTD'] DMEmployee.tblEmpW2['FederalWageQTD']
Emp W2 Federal Wage YTD	EmpW2['Emp W2 Federal Wage YTD'] DMEmployee.tblEmpW2['FederalWageYTD']
Emp W2 Federal Withholding QTD	EmpW2['Emp W2 Federal Withholding QTD'] DMEmployee.tblEmpW2['FederalWHQTD']
Emp W2 Federal Withholding YTD	EmpW2['Emp W2 Federal Withholding YTD'] DMEmployee.tblEmpW2['FederalWHYTD']
Emp W2 FICA Error QTD	EmpW2['Emp W2 FICA Error QTD'] DMEmployee.tblEmpW2['FICAErrorQTD']
Emp W2 FICA Error YTD	EmpW2['Emp W2 FICA Error YTD'] DMEmployee.tblEmpW2['FICAError']
Emp W2 FUTA Raw Wage QTD	EmpW2['Emp W2 FUTA Raw Wage QTD'] DMEmployee.tblEmpW2['FUTARawWageQTD']
Emp W2 FUTA Raw Wage YTD	EmpW2['Emp W2 FUTA Raw Wage YTD'] DMEmployee.tblEmpW2['FUTARawWageYTD']
Emp W2 FUTA Wage QTD	EmpW2['Emp W2 FUTA Wage QTD'] DMEmployee.tblEmpW2['FUTAWageQTD']
Emp W2 FUTA Wage YTD	EmpW2['Emp W2 FUTA Wage YTD'] DMEmployee.tblEmpW2['FUTAWageYTD']
Emp W2 Gross Federal Wage QTD	EmpW2['Emp W2 Gross Federal Wage QTD'] DMEmployee.tblEmpW2['GrossFederalWageQTD']
Emp W2 Gross Federal Wage YTD	EmpW2['Emp W2 Gross Federal Wage YTD'] DMEmployee.tblEmpW2['GrossFederalWageYTD']
Emp W2 Gross Local Wage QTD	EmpW2['Emp W2 Gross Local Wage QTD'] DMEmployee.tblEmpW2['GrossLocalWageQTD']
Emp W2 Gross Local Wage YTD	EmpW2['Emp W2 Gross Local Wage YTD'] DMEmployee.tblEmpW2['GrossLocalWageYTD']
Emp W2 Gross State Wage QTD	EmpW2['Emp W2 Gross State Wage QTD'] DMEmployee.tblEmpW2['GrossStateWageQTD']
Emp W2 Gross State Wage YTD	EmpW2['Emp W2 Gross State Wage YTD'] DMEmployee.tblEmpW2['GrossStateWageYTD']
Emp W2 Is Non Contract	EmpW2['Emp W2 Is Non Contract'] DMEmployee.tblEmpW2['IsNonContract']
Emp W2 Local 1 Name	EmpW2['Emp W2 Local 1 Name'] DMEmployee.tblEmpW2['Local1Name']
Emp W2 Local 1 Wage YTD	EmpW2['Emp W2 Local 1 Wage YTD'] DMEmployee.tblEmpW2['Local1WageYTD']
Emp W2 Local 1 Withholding YTD	EmpW2['Emp W2 Local 1 Withholding YTD'] DMEmployee.tblEmpW2['Local1WHYTD']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp W2 Local 2 Name	EmpW2['Emp W2 Local 2 Name'] DMEmployee.tblEmpW2['Local2Name']
Emp W2 Local 2 Wage YTD	EmpW2['Emp W2 Local 2 Wage YTD'] DMEmployee.tblEmpW2['Local2WageYTD']
Emp W2 Local 2 Withholding YTD	EmpW2['Emp W2 Local 2 Withholding YTD'] DMEmployee.tblEmpW2['Local2WHYTD']
Emp W2 Local 3 Name	EmpW2['Emp W2 Local 3 Name'] DMEmployee.tblEmpW2['Local3Name']
Emp W2 Local 3 Wage YTD	EmpW2['Emp W2 Local 3 Wage YTD'] DMEmployee.tblEmpW2['Local3WageYTD']
Emp W2 Local 3 Withholding YTD	EmpW2['Emp W2 Local 3 Withholding YTD'] DMEmployee.tblEmpW2['Local3WHYTD']
Emp W2 Local 4 Name	EmpW2['Emp W2 Local 4 Name'] DMEmployee.tblEmpW2['Local4Name']
Emp W2 Local 4 Wage YTD	EmpW2['Emp W2 Local 4 Wage YTD'] DMEmployee.tblEmpW2['Local4WageYTD']
Emp W2 Local 4 Withholding YTD	EmpW2['Emp W2 Local 4 Withholding YTD'] DMEmployee.tblEmpW2['Local4WHYTD']
Emp W2 Local Wage QTD	EmpW2['Emp W2 Local Wage QTD'] DMEmployee.tblEmpW2['LocalWageQTD']
Emp W2 Local Wage YTD	EmpW2['Emp W2 Local Wage YTD'] DMEmployee.tblEmpW2['LocalWageYTD']
Emp W2 Local Withholding QTD	EmpW2['Emp W2 Local Withholding QTD'] DMEmployee.tblEmpW2['LocalWithholdingQTD']
Emp W2 Local Withholding YTD	EmpW2['Emp W2 Local Withholding YTD'] DMEmployee.tblEmpW2['LocalWithholdingYTD']
Emp W2 MC Wage QTD	EmpW2['Emp W2 MC Wage QTD'] DMEmployee.tblEmpW2['MCWageQTD']
Emp W2 MC Wage YTD	EmpW2['Emp W2 MC Wage YTD'] DMEmployee.tblEmpW2['MCWageYTD']
Emp W2 MC Withholding QTD	EmpW2['Emp W2 MC Withholding QTD'] DMEmployee.tblEmpW2['MCWithholdingQTD']
Emp W2 MC Withholding YTD	EmpW2['Emp W2 MC Withholding YTD'] DMEmployee.tblEmpW2['MCWithholdingYTD']
Emp W2 Mfg Life QTD	EmpW2['Emp W2 Mfg Life QTD'] DMEmployee.tblEmpW2['Benefit1QTD']
Emp W2 Mfg Life YTD	EmpW2['Emp W2 Mfg Life YTD'] DMEmployee.tblEmpW2['Benefit1YTD']
Emp W2 Name	EmpW2['Emp W2 Name'] DMEmployee.tblEmpW2['Name']
Emp W2 Nationwid QTD	EmpW2['Emp W2 Nationwid QTD'] DMEmployee.tblEmpW2['Benefit2QTD']
Emp W2 Nationwid YTD	EmpW2['Emp W2 Nationwid YTD'] DMEmployee.tblEmpW2['Benefit2YTD']
Emp W2 Net Federal Wage QTD	EmpW2['Emp W2 Net Federal Wage QTD'] DMEmployee.tblEmpW2['NetFederalWageQTD']
Emp W2 Net Federal Wage YTD	EmpW2['Emp W2 Net Federal Wage YTD'] DMEmployee.tblEmpW2['NetFederalWageYTD']
Emp W2 Net Local Wage QTD	EmpW2['Emp W2 Net Local Wage QTD'] DMEmployee.tblEmpW2['NetLocalWageQTD']
Emp W2 Net Local Wage YTD	EmpW2['Emp W2 Net Local Wage YTD'] DMEmployee.tblEmpW2['NetLocalWageYTD']
Emp W2 Net State Wage QTD	EmpW2['Emp W2 Net State Wage QTD'] DMEmployee.tblEmpW2['NetStateWageQTD']
Emp W2 Net State Wage YTD	EmpW2['Emp W2 Net State Wage YTD'] DMEmployee.tblEmpW2['NetStateWageYTD']
Emp W2 Reimbursements QTD	EmpW2['Emp W2 Reimbursements QTD'] DMEmployee.tblEmpW2['ReimbursementsQTD']
Emp W2 Reimbursements YTD	EmpW2['Emp W2 Reimbursements YTD'] DMEmployee.tblEmpW2['ReimbursementsYTD']
Emp W2 Retirement Plan	EmpW2['Emp W2 Retirement Plan'] DMEmployee.tblEmpW2['RetirementPlan']
Emp W2 SDIF Raw Wage QTD	EmpW2['Emp W2 SDIF Raw Wage QTD'] DMEmployee.tblEmpW2['SDIFRawWageQTD']
Emp W2 SDIF Raw Wage YTD	EmpW2['Emp W2 SDIF Raw Wage YTD'] DMEmployee.tblEmpW2['SDIFRawWageYTD']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp W2 SDIF Wage QTD	EmpW2['Emp W2 SDIF Wage QTD'] DMEmployee.tblEmpW2['SDIFWageQTD']
Emp W2 SDIF Wage YTD	EmpW2['Emp W2 SDIF Wage YTD'] DMEmployee.tblEmpW2['SDIFWageYTD']
Emp W2 SDIF Withholding QTD	EmpW2['Emp W2 SDIF Withholding QTD'] DMEmployee.tblEmpW2['SDIFWithholdingQTD']
Emp W2 SDIF Withholding YTD	EmpW2['Emp W2 SDIF Withholding YTD'] DMEmployee.tblEmpW2['SDIFWithholdingYTD']
Emp W2 SSN	EmpW2['Emp W2 SSN'] DMEmployee.tblEmpW2['SSN']
Emp W2 SS Wage QTD	EmpW2['Emp W2 SS Wage QTD'] DMEmployee.tblEmpW2['SSWageQTD']
Emp W2 SS Wage YTD	EmpW2['Emp W2 SS Wage YTD'] DMEmployee.tblEmpW2['SSWageYTD']
Emp W2 SS Withholding QTD	EmpW2['Emp W2 SS Withholding QTD'] DMEmployee.tblEmpW2['SSWithholdingQTD']
Emp W2 SS Withholding YTD	EmpW2['Emp W2 SS Withholding YTD'] DMEmployee.tblEmpW2['SSWithholdingYTD']
Emp W2 State 1 Abbr	EmpW2['Emp W2 State 1 Abbr'] DMEmployee.tblEmpW2['State1Abbr']
Emp W2 State 1 ID	EmpW2['Emp W2 State 1 ID'] DMEmployee.tblEmpW2['State1ID']
Emp W2 State 1 Wage YTD	EmpW2['Emp W2 State 1 Wage YTD'] DMEmployee.tblEmpW2['State1WageYTD']
Emp W2 State 1 Withholding YTD	EmpW2['Emp W2 State 1 Withholding YTD'] DMEmployee.tblEmpW2['State1WHYTD']
Emp W2 State 2 Abbr	EmpW2['Emp W2 State 2 Abbr'] DMEmployee.tblEmpW2['State2Abbr']
Emp W2 State 2 ID	EmpW2['Emp W2 State 2 ID'] DMEmployee.tblEmpW2['State2ID']
Emp W2 State 2 Wage YTD	EmpW2['Emp W2 State 2 Wage YTD'] DMEmployee.tblEmpW2['State2WageYTD']
Emp W2 State 2 Withholding YTD	EmpW2['Emp W2 State 2 Withholding YTD'] DMEmployee.tblEmpW2['State2WHYTD']
Emp W2 State 3 Abbr	EmpW2['Emp W2 State 3 Abbr'] DMEmployee.tblEmpW2['State3Abbr']
Emp W2 State 3 ID	EmpW2['Emp W2 State 3 ID'] DMEmployee.tblEmpW2['State3ID']
Emp W2 State 3 Wage YTD	EmpW2['Emp W2 State 3 Wage YTD'] DMEmployee.tblEmpW2['State3WageYTD']
Emp W2 State 3 Withholding YTD	EmpW2['Emp W2 State 3 Withholding YTD'] DMEmployee.tblEmpW2['State3WHYTD']
Emp W2 State 4 Abbr	EmpW2['Emp W2 State 4 Abbr'] DMEmployee.tblEmpW2['State4Abbr']
Emp W2 State 4 ID	EmpW2['Emp W2 State 4 ID'] DMEmployee.tblEmpW2['State4ID']
Emp W2 State 4 Wage YTD	EmpW2['Emp W2 State 4 Wage YTD'] DMEmployee.tblEmpW2['State4WageYTD']
Emp W2 State 4 Withholding YTD	EmpW2['Emp W2 State 4 Withholding YTD'] DMEmployee.tblEmpW2['State4WHYTD']
Emp W2 State Wage QTD	EmpW2['Emp W2 State Wage QTD'] DMEmployee.tblEmpW2['StateWageQTD']
Emp W2 State Wage YTD	EmpW2['Emp W2 State Wage YTD'] DMEmployee.tblEmpW2['StateWageYTD']
Emp W2 State Withholding QTD	EmpW2['Emp W2 State Withholding QTD'] DMEmployee.tblEmpW2['StateWithholdingQTD']
Emp W2 State Withholding YTD	EmpW2['Emp W2 State Withholding YTD'] DMEmployee.tblEmpW2['StateWithholdingYTD']
Emp W2 Statutory Employee	EmpW2['Emp W2 Statutory Employee'] DMEmployee.tblEmpW2['StatutoryEmployee']
Emp W2 SUTA Wage QTD	EmpW2['Emp W2 SUTA Wage QTD'] DMEmployee.tblEmpW2['SUTAWageQTD']
Emp W2 SUTA Raw Wage QTD	EmpW2['Emp W2 SUTA Raw Wage QTD'] DMEmployee.tblEmpW2['SUTARawWageQTD']
Emp W2 SUTA Raw Wage YTD	EmpW2['Emp W2 SUTA Raw Wage YTD'] DMEmployee.tblEmpW2['SUTARawWageYTD']



## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Emp W2 SUTA Wage YTD	EmpW2['Emp W2 SUTA Wage YTD'] DMEmployee.tblEmpW2['SUTAWageYTD']
Emp W2 Third Party Sick Pay	EmpW2['Emp W2 Third Party Sick Pay'] DMEmployee.tblEmpW2['ThirdPartySickPay']
Emp W2 Valic QTD	EmpW2['Emp W2 Valic QTD'] DMEmployee.tblEmpW2['Benefit3QTD']
Emp W2 Valic YTD	EmpW2['Emp W2 Valic YTD'] DMEmployee.tblEmpW2['Benefit3YTD']
Grp Date Changed	Grp['Grp Date Changed'] DM.GrpTbl['DateChanged']
Grp Description	Grp['Grp Description'] DM.GrpTbl['Description']
Grp Name	Grp['Grp Name'] DM.GrpTbl['Name']
Grp COA Alternate Number	GrpCoa['Grp COA Alternate Number'] DM.GrpCoaTbl['AlternateNumber']
Grp COA Balance Sheet Column	GrpCoa['Grp COA Balance Sheet Column'] DM.GrpCoaTbl['BalanceSheetColumn']
Grp COA Blank Lines	GrpCoa['Grp COA Blank Lines'] DM.GrpCoaTbl['BlankLines']
Grp COA Blank Page	GrpCoa['Grp COA Blank Page'] DM.GrpCoaTbl['BlankPage']
Grp COA Comments	GrpCoa['Grp COA Comments'] DM.GrpCoaTbl['Comments']
Grp COA Date Changed	GrpCoa['Grp COA Date Changed'] DM.GrpCoaTbl['DateChanged']
Grp COA Date Created	GrpCoa['Grp COA Date Created'] DM.GrpCoaTbl['DateCreated']
Grp COA Master Account	GrpCoa['Grp COA Master Account'] DM.GrpCoaTbl['MasterAccount']
Grp COA Name	GrpCoa['Grp COA Name'] DM.GrpCoaTbl['Name']
Grp COA Number	GrpCoa['Grp COA Number'] DM.GrpCoaTbl['Number']
Grp COA Source	GrpCoa['Grp COA Source'] DM.GrpCoaTbl['Description']
Grp COA Sub Account	GrpCoa['Grp COA Sub Account'] DM.GrpCoaTbl['SubAccount']
Grp COA Total Level	GrpCoa['Grp COA Total Level'] DM.GrpCoaTbl['TotalLevel']
Inv Actual Amount	Inv['Inv Actual Amount'] DM.InvTbl['ActualAmount']
Inv Actual Discount Amount	Inv['Inv Actual Discount Amount'] DM.InvTbl['ActualDiscountAmount']
Inv Amount	Inv['Inv Amount'] DM.InvTbl['Amount']
Inv Amount Not Voided	Inv['Inv Amount Not Voided'] DM.InvTbl['AmountNotVoided']
Inv Authorized By	Inv['Inv Authorized By'] DM.InvTbl['AuthorizedBy']
Inv Batch Number	Inv['Inv Batch Number'] DM.InvTbl['Batch']
Inv Cash Account Number	Inv['Inv Cash Account Number'] DM.InvTbl['CashAcctNum']
Inv Date Changed	Inv['Inv Date Changed'] DM.InvTbl['DateChanged']
Inv Date Created	Inv['Inv Date Created'] DM.InvTbl['DateCreated']
Inv Discount Pct	Inv['Inv Discount Pct'] DM.InvTbl['DiscountPct']
Inv Discount Amount	Inv['Inv Discount Amount'] DM.InvTbl['DiscountAmt']
Inv Discount Date	Inv['Inv Discount Date'] DM.InvTbl['DiscountDate']
Inv Discount Terms	Inv['Inv Discount Terms'] DM.InvTbl['DiscountTerms']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Inv Discount Taken	Inv['Inv Discount Taken'] DM.InvTbl['DiscountTaken']
Inv Due Date With Discount	Inv['Inv Due Date With Discount'] DM.InvTbl['DueDateWithDiscount']
Inv Due Date	Inv['Inv Due Date'] DM.InvTbl['DueDate']
Inv Electronic Payment	Inv['Inv Electronic Payment'] DM.InvTbl['ElectronicPayment']
Inv Invoice Amount	Inv['Inv Invoice Amount'] DM.InvTbl['InvoiceAmount']
Inv Invoice Date	Inv['Inv Invoice Date'] DM.InvTbl['InvoiceDate']
Inv Invoice Number	Inv['Inv Invoice Number'] DM.InvTbl['Reference']
Inv Is Reconciled	Inv['Inv Is Reconciled'] DM.InvTbl['IsReconciled']
Inv Marked	Inv['Inv Marked'] DM.InvTbl['Marked']
Inv Notes	Inv['Inv Notes'] DM.InvTbl['Memo']
Inv O T V	Inv['Inv O T V'] DM.InvTbl['OTV']
Inv Partial Payment	Inv['Inv Partial Payment'] DM.InvTbl['PartialPayment']
Inv Payee Source Abbr	Inv['Inv Payee Source Abbr'] DM.InvTbl['PayeeSourceAbbr']
Inv Payee Source Name	Inv['Inv Payee Source Name'] DM.InvTbl['PayeeSourceName']
Inv Payment Authorized	Inv['Inv Payment Authorized'] DM.InvTbl['Authorized']
Inv Period	Inv['Inv Period'] DM.InvTbl['Period']
Inv Purchase Order	Inv['Inv Purchase Order'] DM.InvTbl['PurchaseOrder']
Inv Reconcile Date	Inv['Inv Reconcile Date'] DM.InvTbl['ReconcileLabel']
Inv Reissue of Old	Inv['Inv Reissue of Old'] DM.InvTbl['ReissueOfOld']
Inv Reverse of Old	Inv['Inv Reverse of Old'] DM.InvTbl['ReverseOfOld']
Inv Terms	Inv['Inv Terms'] DM.InvTbl['Terms']
Inv Transaction Date	Inv['Inv Transaction Date'] DM.InvTbl['TDate']
Inv Voided or Reversed	Inv['Inv Voided or Reversed'] DM.InvTbl['VoidedOrReversed']
Inv Voided	Inv['Inv Voided'] DM.InvTbl['Voided']
Prior Totals Actual Y1	PriorTotals['Prior Totals Actual Y1'] DM.CoaPYTbl['ActualY1']
Prior Totals Actual Y1 M1	PriorTotals['Prior Totals Actual Y1 M1'] DM.CoaPYTbl['ActualY1M1']
Prior Totals Actual Y1 M10	PriorTotals['Prior Totals Actual Y1 M10'] DM.CoaPYTbl['ActualY1M10']
Prior Totals Actual Y1 M11	PriorTotals['Prior Totals Actual Y1 M11'] DM.CoaPYTbl['ActualY1M11']
Prior Totals Actual Y1 M12	PriorTotals['Prior Totals Actual Y1 M12'] DM.CoaPYTbl['ActualY1M12']
Prior Totals Actual Y1 M2	PriorTotals['Prior Totals Actual Y1 M2'] DM.CoaPYTbl['ActualY1M2']
Prior Totals Actual Y1 M3	PriorTotals['Prior Totals Actual Y1 M3'] DM.CoaPYTbl['ActualY1M3']
Prior Totals Actual Y1 M4	PriorTotals['Prior Totals Actual Y1 M4'] DM.CoaPYTbl['ActualY1M4']
Prior Totals Actual Y1 M5	PriorTotals['Prior Totals Actual Y1 M5'] DM.CoaPYTbl['ActualY1M5']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Prior Totals Actual Y1 M6	PriorTotals['Prior Totals Actual Y1 M6'] DM.CoaPYTbl['ActualY1M6']
Prior Totals Actual Y1 M7	PriorTotals['Prior Totals Actual Y1 M7'] DM.CoaPYTbl['ActualY1M7']
Prior Totals Actual Y1 M8	PriorTotals['Prior Totals Actual Y1 M8'] DM.CoaPYTbl['ActualY1M8']
Prior Totals Actual Y1 M9	PriorTotals['Prior Totals Actual Y1 M9'] DM.CoaPYTbl['ActualY1M9']
Prior Totals Actual Y2	PriorTotals['Prior Totals Actual Y2'] DM.CoaPYTbl['ActualY2']
Prior Totals Actual Y2 M1	PriorTotals['Prior Totals Actual Y2 M1'] DM.CoaPYTbl['ActualY2M1']
Prior Totals Actual Y2 M10	PriorTotals['Prior Totals Actual Y2 M10'] DM.CoaPYTbl['ActualY2M10']
Prior Totals Actual Y2 M11	PriorTotals['Prior Totals Actual Y2 M11'] DM.CoaPYTbl['ActualY2M11']
Prior Totals Actual Y2 M12	PriorTotals['Prior Totals Actual Y2 M12'] DM.CoaPYTbl['ActualY2M12']
Prior Totals Actual Y2 M2	PriorTotals['Prior Totals Actual Y2 M2'] DM.CoaPYTbl['ActualY2M2']
Prior Totals Actual Y2 M3	PriorTotals['Prior Totals Actual Y2 M3'] DM.CoaPYTbl['ActualY2M3']
Prior Totals Actual Y2 M4	PriorTotals['Prior Totals Actual Y2 M4'] DM.CoaPYTbl['ActualY2M4']
Prior Totals Actual Y2 M5	PriorTotals['Prior Totals Actual Y2 M5'] DM.CoaPYTbl['ActualY2M5']
Prior Totals Actual Y2 M6	PriorTotals['Prior Totals Actual Y2 M6'] DM.CoaPYTbl['ActualY2M6']
Prior Totals Actual Y2 M7	PriorTotals['Prior Totals Actual Y2 M7'] DM.CoaPYTbl['ActualY2M7']
Prior Totals Actual Y2 M8	PriorTotals['Prior Totals Actual Y2 M8'] DM.CoaPYTbl['ActualY2M8']
Prior Totals Actual Y2 M9	PriorTotals['Prior Totals Actual Y2 M9'] DM.CoaPYTbl['ActualY2M9']
Prior Totals Actual Y3	PriorTotals['Prior Totals Actual Y3'] DM.CoaPYTbl['ActualY3']
Prior Totals Actual Y3 M1	PriorTotals['Prior Totals Actual Y3 M1'] DM.CoaPYTbl['ActualY3M1']
Prior Totals Actual Y3 M10	PriorTotals['Prior Totals Actual Y3 M10'] DM.CoaPYTbl['ActualY3M10']
Prior Totals Actual Y3 M11	PriorTotals['Prior Totals Actual Y3 M11'] DM.CoaPYTbl['ActualY3M11']
Prior Totals Actual Y3 M12	PriorTotals['Prior Totals Actual Y3 M12'] DM.CoaPYTbl['ActualY3M12']
Prior Totals Actual Y3 M2	PriorTotals['Prior Totals Actual Y3 M2'] DM.CoaPYTbl['ActualY3M2']
Prior Totals Actual Y3 M3	PriorTotals['Prior Totals Actual Y3 M3'] DM.CoaPYTbl['ActualY3M3']
Prior Totals Actual Y3 M4	PriorTotals['Prior Totals Actual Y3 M4'] DM.CoaPYTbl['ActualY3M4']
Prior Totals Actual Y3 M5	PriorTotals['Prior Totals Actual Y3 M5'] DM.CoaPYTbl['ActualY3M5']
Prior Totals Actual Y3 M6	PriorTotals['Prior Totals Actual Y3 M6'] DM.CoaPYTbl['ActualY3M6']
Prior Totals Actual Y3 M7	PriorTotals['Prior Totals Actual Y3 M7'] DM.CoaPYTbl['ActualY3M7']
Prior Totals Actual Y3 M8	PriorTotals['Prior Totals Actual Y3 M8'] DM.CoaPYTbl['ActualY3M8']
Prior Totals Actual Y3 M9	PriorTotals['Prior Totals Actual Y3 M9'] DM.CoaPYTbl['ActualY3M9']
Prior Totals Budget Y1	PriorTotals['Prior Totals Budget Y1'] DM.CoaPYTbl['BudgetY1']
Prior Totals Budget Y1 M1	PriorTotals['Prior Totals Budget Y1 M1'] DM.CoaPYTbl['BudgetY1M1']
Prior Totals Budget Y1 M10	PriorTotals['Prior Totals Budget Y1 M10'] DM.CoaPYTbl['BudgetY1M10']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Prior Totals Budget Y1 M11	PriorTotals['Prior Totals Budget Y1 M11'] DM.CoaPYTbl['BudgetY1M11']
Prior Totals Budget Y1 M12	PriorTotals['Prior Totals Budget Y1 M12'] DM.CoaPYTbl['BudgetY1M12']
Prior Totals Budget Y1 M2	PriorTotals['Prior Totals Budget Y1 M2'] DM.CoaPYTbl['BudgetY1M2']
Prior Totals Budget Y1 M3	PriorTotals['Prior Totals Budget Y1 M3'] DM.CoaPYTbl['BudgetY1M3']
Prior Totals Budget Y1 M4	PriorTotals['Prior Totals Budget Y1 M4'] DM.CoaPYTbl['BudgetY1M4']
Prior Totals Budget Y1 M5	PriorTotals['Prior Totals Budget Y1 M5'] DM.CoaPYTbl['BudgetY1M5']
Prior Totals Budget Y1 M6	PriorTotals['Prior Totals Budget Y1 M6'] DM.CoaPYTbl['BudgetY1M6']
Prior Totals Budget Y1 M7	PriorTotals['Prior Totals Budget Y1 M7'] DM.CoaPYTbl['BudgetY1M7']
Prior Totals Budget Y1 M8	PriorTotals['Prior Totals Budget Y1 M8'] DM.CoaPYTbl['BudgetY1M8']
Prior Totals Budget Y1 M9	PriorTotals['Prior Totals Budget Y1 M9'] DM.CoaPYTbl['BudgetY1M9']
Prior Totals Budget Y2	PriorTotals['Prior Totals Budget Y2'] DM.CoaPYTbl['BudgetY2']
Prior Totals Budget Y2 M1	PriorTotals['Prior Totals Budget Y2 M1'] DM.CoaPYTbl['BudgetY2M1']
Prior Totals Budget Y2 M10	PriorTotals['Prior Totals Budget Y2 M10'] DM.CoaPYTbl['BudgetY2M10']
Prior Totals Budget Y2 M11	PriorTotals['Prior Totals Budget Y2 M11'] DM.CoaPYTbl['BudgetY2M11']
Prior Totals Budget Y2 M12	PriorTotals['Prior Totals Budget Y2 M12'] DM.CoaPYTbl['BudgetY2M12']
Prior Totals Budget Y2 M2	PriorTotals['Prior Totals Budget Y2 M2'] DM.CoaPYTbl['BudgetY2M2']
Prior Totals Budget Y2 M3	PriorTotals['Prior Totals Budget Y2 M3'] DM.CoaPYTbl['BudgetY2M3']
Prior Totals Budget Y2 M4	PriorTotals['Prior Totals Budget Y2 M4'] DM.CoaPYTbl['BudgetY2M4']
Prior Totals Budget Y2 M5	PriorTotals['Prior Totals Budget Y2 M5'] DM.CoaPYTbl['BudgetY2M5']
Prior Totals Budget Y2 M6	PriorTotals['Prior Totals Budget Y2 M6'] DM.CoaPYTbl['BudgetY2M6']
Prior Totals Budget Y2 M7	PriorTotals['Prior Totals Budget Y2 M7'] DM.CoaPYTbl['BudgetY2M7']
Prior Totals Budget Y2 M8	PriorTotals['Prior Totals Budget Y2 M8'] DM.CoaPYTbl['BudgetY2M8']
Prior Totals Budget Y2 M9	PriorTotals['Prior Totals Budget Y2 M9'] DM.CoaPYTbl['BudgetY2M9']
Prior Totals Budget Y3	PriorTotals['Prior Totals Budget Y3'] DM.CoaPYTbl['BudgetY3']
Prior Totals Budget Y3 M1	PriorTotals['Prior Totals Budget Y3 M1'] DM.CoaPYTbl['BudgetY3M1']
Prior Totals Budget Y3 M10	PriorTotals['Prior Totals Budget Y3 M10'] DM.CoaPYTbl['BudgetY3M10']
Prior Totals Budget Y3 M11	PriorTotals['Prior Totals Budget Y3 M11'] DM.CoaPYTbl['BudgetY3M11']
Prior Totals Budget Y3 M12	PriorTotals['Prior Totals Budget Y3 M12'] DM.CoaPYTbl['BudgetY3M12']
Prior Totals Budget Y3 M2	PriorTotals['Prior Totals Budget Y3 M2'] DM.CoaPYTbl['BudgetY3M2']
Prior Totals Budget Y3 M3	PriorTotals['Prior Totals Budget Y3 M3'] DM.CoaPYTbl['BudgetY3M3']
Prior Totals Budget Y3 M4	PriorTotals['Prior Totals Budget Y3 M4'] DM.CoaPYTbl['BudgetY3M4']
Prior Totals Budget Y3 M5	PriorTotals['Prior Totals Budget Y3 M5'] DM.CoaPYTbl['BudgetY3M5']
Prior Totals Budget Y3 M6	PriorTotals['Prior Totals Budget Y3 M6'] DM.CoaPYTbl['BudgetY3M6']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Prior Totals Budget Y3 M7	PriorTotals['Prior Totals Budget Y3 M7'] DM.CoaPYTbl['BudgetY3M7']
Prior Totals Budget Y3 M8	PriorTotals['Prior Totals Budget Y3 M8'] DM.CoaPYTbl['BudgetY3M8']
Prior Totals Budget Y3 M9	PriorTotals['Prior Totals Budget Y3 M9'] DM.CoaPYTbl['BudgetY3M9']
Rcv Amount	Rcv['Rcv Amount'] DM.RcvTbl['Amount']
Rcv Amount Not Voided	Rcv['Rcv Amount Not Voided'] DM.RcvTbl['AmountNotVoided']
Rcv Batch Number	Rcv['Rcv Batch Number'] DM.RcvTbl['Batch']
Rcv Cash Account Number	Rcv['Rcv Cash Account Number'] DM.RcvTbl['CashAcctNum']
Rcv Date Changed	Rcv['Rcv Date Changed'] DM.RcvTbl['DateChanged']
Rcv Date Created	Rcv['Rcv Date Created'] DM.RcvTbl['DateCreated']
Rcv Date	Rcv['Rcv Date'] DM.RcvTbl['TDate']
Rcv Income Source Abbr	Rcv['Rcv Income Source Abbr'] DM.RcvTbl['PayeeSourceAbbr']
Rcv Income Source	Rcv['Rcv Income Source'] DM.RcvTbl['PayeeSourceName']
Rcv Is Reconciled	Rcv['Rcv Is Reconciled'] DM.RcvTbl['IsReconciled']
Rcv Marked	Rcv['Rcv Marked'] DM.RcvTbl['Marked']
Rcv Notes	Rcv['Rcv Notes'] DM.RcvTbl['Memo']
Rcv O T V	Rcv['Rcv O T V'] DM.RcvTbl['OTV']
Rcv Partial Payment	Rcv['Rcv Partial Payment'] DM.RcvTbl['PartialPayment']
Rcv Period	Rcv['Rcv Period'] DM.RcvTbl['Period']
Rcv Reconcile Date	Rcv['Rcv Reconcile Date'] DM.RcvTbl['ReconcileLabel']
Rcv Reference	Rcv['Rcv Reference'] DM.RcvTbl['Reference']
Rcv Reissue of Old	Rcv['Rcv Reissue of Old'] DM.RcvTbl['ReissueOfOld']
Rcv Reverse of Old	Rcv['Rcv Reverse of Old'] DM.RcvTbl['ReverseOfOld']
Rcv Voided or Reversed	Rcv['Rcv Voided or Reversed'] DM.RcvTbl['VoidedOrReversed']
Rcv Voided	Rcv['Rcv Voided'] DM.RcvTbl['Voided']
R J E Date Changed	RJE['R J E Date Changed'] DM.RJETbl['DateChanged']
R J E Date Created	RJE['R J E Date Created'] DM.RJETbl['DateCreated']
R J E Inactive	RJE['R J E Inactive'] DM.RJETbl['Inactive']
R J E Last Amount Posted	RJE['R J E Last Amount Posted'] DM.RJETbl['LastAmount']
R J E Last Date Posted	RJE['R J E Last Date Posted'] DM.RJETbl['LastDate']
R J E Max # of Times	RJE['R J E Max # of Times'] DM.RJETbl['LimitCount']
R J E Max Total Amount	RJE['R J E Max Total Amount'] DM.RJETbl['LimitTotal']
R J E Name	RJE['R J E Name'] DM.RJETbl['Name']
R J E New Date	RJE['R J E New Date'] DM.RJETbl['NewDate']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
R J E Period	RJE['R J E Period'] DM.RJETbl['PeriodName']
R J E Recursion Type	RJE['R J E Recursion Type'] DM.RJETbl['RecursionTypeName']
R J E Remaining # of Times	RJE['R J E Remaining # of Times'] DM.RJETbl['RunningCount']
R J E Remaining Total Amount	RJE['R J E Remaining Total Amount'] DM.RJETbl['RunningTotal']
R J E Use Percent	RJE['R J E Use Percent'] DM.RJETbl['UsePercent']
R J E Voided	RJE['R J E Voided'] DM.RJETbl['Voided']
R J E Voided Or Reversed	RJE['R J E Voided Or Reversed'] DM.RJETbl['VoidedOrReversed']
R J E Trn Account Number	RJETrn['R J E Trn Account Number'] DM.SubRJETbl['Number']
R J E Trn Credit	RJETrn['R J E Trn Credit'] DM.SubRJETbl['Credit']
R J E Trn Date	RJETrn['R J E Trn Date'] DM.SubRJETbl['TDate']
R J E Trn Debit	RJETrn['R J E Trn Debit'] DM.SubRJETbl['Debit']
R J E Trn Description	RJETrn['R J E Trn Description'] DM.SubRJETbl['Description']
R J E Trn Voided	RJETrn['R J E Trn Voided'] DM.SubRJETbl['Voided']
R J E Trn Voided Or Reversed	RJETrn['R J E Trn Voided Or Reversed'] DM.SubRJETbl['VoidedOrReversed']
Src Abbr	Src['Src Abbr'] DM.VenTbl['Abbr']
Src Account Number	Src['Src Account Number'] DM.VenTbl['CustomerAcctNum']
Src Blank	Src['Src Blank'] DM.VenTbl['Blank']
Src Corrected Name	Src['Src Corrected Name'] DM.VenTbl['CorrectedName']
Src Date Changed	Src['Src Date Changed'] DM.VenTbl['DateChanged']
Src Date Created	Src['Src Date Created'] DM.VenTbl['DateCreated']
Src Date W9 Requested	Src['Src Date W9 Requested'] DM.VenTbl['W9Requested']
Src Discount Terms	Src['Src Discount Terms'] DM.VenTbl['calcDiscountTerms']
Src Doing Business as Name	Src['Src Doing Business as Name'] DM.VenTbl['DBAName']
Src Federal ID Number	Src['Src Federal ID Number'] DM.VenTbl['FedID']
Src ID Number	Src['Src ID Number'] DM.VenTbl['ID']
Src Inactive	Src['Src Inactive'] DM.VenTbl['Inactive']
Src Name	Src['Src Name'] DM.VenTbl['Name']
Src Organization Type	Src['Src Organization Type'] DM.VenTbl['calcVendorType']
Src Receives 1099	Src['Src Receives 1099'] DM.VenTbl['Receives1099']
Src Remarks	Src['Src Remarks'] DM.VenTbl['Remarks']
Src Since	Src['Src Since'] DM.VenTbl['VendorSince']
Src Terms	Src['Src Terms'] DM.VenTbl['calcTerms']
Src Unique ID	Src['Src Unique ID'] DM.VenTbl['VenRec']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Src W9 On File	Src['Src W9 On File'] DM.VenTbl['W9OnFile']
Src Web Page	Src['Src Web Page'] DM.VenTbl['WebPage']
Src Address Address 1	SrcAddress['Src Address Address 1'] DM.VContactTbl['Address']
Src Address Address 2	SrcAddress['Src Address Address 2'] DM.VContactTbl['Address2']
Src Address Address Block	SrcAddress['Src Address Address Block'] DM.VContactTbl['AddressBlock']
Src Address City Only	SrcAddress['Src Address City Only'] DM.VContactTbl['CityOnly']
Src Address City/State	SrcAddress['Src Address City/State'] DM.VContactTbl['CityState']
Src Address City/State/Zip	SrcAddress['Src Address City/State/Zip'] DM.VContactTbl['CityZip']
Src Address E Mail	SrcAddress['Src Address E Mail'] DM.VContactTbl['EMail']
Src Address Mailing Name	SrcAddress['Src Address Mailing Name'] DM.VContactTbl['Mailing']
Src Address Name	SrcAddress['Src Address Name'] DM.VContactTbl['Name']
Src Address Notes	SrcAddress['Src Address Notes'] DM.VContactTbl['Notes']
Src Address Phone List	SrcAddress['Src Address Phone List'] DM.VContactTbl['PhoneList']
Src Address Position	SrcAddress['Src Address Position'] DM.VContactTbl['Position']
Src Address Salutation	SrcAddress['Src Address Salutation'] DM.VContactTbl['Salutation']
Src Address State Only	SrcAddress['Src Address State Only'] DM.VContactTbl['StateOnly']
Src Address Zip	SrcAddress['Src Address Zip'] DM.VContactTbl['Zip']
Src Phone Number	SrcPhone['Src Phone Number'] DM.VCPhoneTbl['NNumber']
Src Phone Type	SrcPhone['Src Phone Type'] DM.VCPhoneTbl['PhoneType']
Src Phone Unlisted	SrcPhone['Src Phone Unlisted'] DM.VCPhoneTbl['Unlisted']
Src Phone X Number	SrcPhone['Src Phone X Number'] DM.VCPhoneTbl['XNumber']
Src Totals Balance	SrcTotals['Src Totals Balance'] DM.VTotalsTbl['Balance']
Src Totals Calendar YTD	SrcTotals['Src Totals Calendar YTD'] DM.VTotalsTbl['CalYTD']
Src Totals Fiscal YTD	SrcTotals['Src Totals Fiscal YTD'] DM.VTotalsTbl['FiscalYTD']
Src Totals Last Amount	SrcTotals['Src Totals Last Amount'] DM.VTotalsTbl['LastAmount']
Src Totals Last Date	SrcTotals['Src Totals Last Date'] DM.VTotalsTbl['LastDate']
Src Totals Last Reference	SrcTotals['Src Totals Last Reference'] DM.VTotalsTbl['LastReference']
Src Totals Next Amount	SrcTotals['Src Totals Next Amount'] DM.VTotalsTbl['NextAmount']
Src Totals Next Date	SrcTotals['Src Totals Next Date'] DM.VTotalsTbl['NextDate']
Src Totals Next Reference	SrcTotals['Src Totals Next Reference'] DM.VTotalsTbl['NextReference']
Sub Inv Amount	SubInv['Sub Inv Amount'] DM.SubInvTbl['Amount']
Sub Inv Amount Not Voided	SubInv['Sub Inv Amount Not Voided'] DM.SubInvTbl['AmountNotVoided']
Sub Inv Credit	SubInv['Sub Inv Credit'] DM.SubInvTbl['Credit']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Sub Inv Date	SubInv['Sub Inv Date'] DM.SubInvTbl['TDate']
Sub Inv Debit	SubInv['Sub Inv Debit'] DM.SubInvTbl['Debit']
Sub Inv Description	SubInv['Sub Inv Description'] DM.SubInvTbl['Description']
Sub Inv Name	SubInv['Sub Inv Name'] DM.SubInvTbl['AccName']
Sub Inv Number	SubInv['Sub Inv Number'] DM.SubInvTbl['Number']
Sub Inv Reconciled	SubInv['Sub Inv Reconciled'] DM.SubInvTbl['Reconciled']
Sub Inv Voided or Reversed	SubInv['Sub Inv Voided or Reversed'] DM.SubInvTbl['VoidedOrReversed']
Sub Inv Voided	SubInv['Sub Inv Voided'] DM.SubInvTbl['Voided']
Sub Rcv Amount	SubRcv['Sub Rcv Amount'] DM.SubRcvTbl['Amount']
Sub Rcv Amount Not Voided	SubRcv['Sub Rcv Amount Not Voided'] DM.SubRcvTbl['AmountNotVoided']
Sub Rcv Credit	SubRcv['Sub Rcv Credit'] DM.SubRcvTbl['Credit']
Sub Rcv Date	SubRcv['Sub Rcv Date'] DM.SubRcvTbl['TDate']
Sub Rcv Debit	SubRcv['Sub Rcv Debit'] DM.SubRcvTbl['Debit']
Sub Rcv Description	SubRcv['Sub Rcv Description'] DM.SubRcvTbl['Description']
Sub Rcv Name	SubRcv['Sub Rcv Name'] DM.SubRcvTbl['AccName']
Sub Rcv Number	SubRcv['Sub Rcv Number'] DM.SubRcvTbl['Number']
Sub Rcv Reconciled	SubRcv['Sub Rcv Reconciled'] DM.SubRcvTbl['Reconciled']
Sub Rcv Voided or Reversed	SubRcv['Sub Rcv Voided or Reversed'] DM.SubRcvTbl['VoidedOrReversed']
Sub Rcv Voided	SubRcv['Sub Rcv Voided'] DM.SubRcvTbl['Voided']
Sub Trn Amount	SubTrn['Sub Trn Amount'] DM.SubTrnTbl['Amount']
Sub Trn Amount Not Voided	SubTrn['Sub Trn Amount Not Voided'] DM.SubTrnTbl['AmountNotVoided']
Sub Trn Credit	SubTrn['Sub Trn Credit'] DM.SubTrnTbl['Credit']
Sub Trn Date	SubTrn['Sub Trn Date'] DM.SubTrnTbl['TDate']
Sub Trn Debit	SubTrn['Sub Trn Debit'] DM.SubTrnTbl['Debit']
Sub Trn Description	SubTrn['Sub Trn Description'] DM.SubTrnTbl['Description']
Sub Trn Name	SubTrn['Sub Trn Name'] DM.SubTrnTbl['Name']
Sub Trn Number	SubTrn['Sub Trn Number'] DM.SubTrnTbl['Number']
Sub Trn Reconciled	SubTrn['Sub Trn Reconciled'] DM.SubTrnTbl['Reconciled']
Sub Trn Voided or Reversed	SubTrn['Sub Trn Voided or Reversed'] DM.SubTrnTbl['VoidedOrReversed']
Sub Trn Voided	SubTrn['Sub Trn Voided'] DM.SubTrnTbl['Voided']
Totals Actual for Month 1	Totals['Totals Actual for Month 1'] DM.CoaTotalsTbl['ActualMonth1']
Totals Actual for Month 10	Totals['Totals Actual for Month 10'] DM.CoaTotalsTbl['ActualMonth10']
Totals Actual for Month 11	Totals['Totals Actual for Month 11'] DM.CoaTotalsTbl['ActualMonth11']



## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Totals Actual for Month 12	Totals['Totals Actual for Month 12'] DM.CoaTotalsTbl['ActualMonth12']
Totals Actual for Month 2	Totals['Totals Actual for Month 2'] DM.CoaTotalsTbl['ActualMonth2']
Totals Actual for Month 3	Totals['Totals Actual for Month 3'] DM.CoaTotalsTbl['ActualMonth3']
Totals Actual for Month 4	Totals['Totals Actual for Month 4'] DM.CoaTotalsTbl['ActualMonth4']
Totals Actual for Month 5	Totals['Totals Actual for Month 5'] DM.CoaTotalsTbl['ActualMonth5']
Totals Actual for Month 6	Totals['Totals Actual for Month 6'] DM.CoaTotalsTbl['ActualMonth6']
Totals Actual for Month 7	Totals['Totals Actual for Month 7'] DM.CoaTotalsTbl['ActualMonth7']
Totals Actual for Month 8	Totals['Totals Actual for Month 8'] DM.CoaTotalsTbl['ActualMonth8']
Totals Actual for Month 9	Totals['Totals Actual for Month 9'] DM.CoaTotalsTbl['ActualMonth9']
Totals Actual for Qtr 1	Totals['Totals Actual for Qtr 1'] DM.CoaTotalsTbl['ActualQtr1']
Totals Actual for Qtr 2	Totals['Totals Actual for Qtr 2'] DM.CoaTotalsTbl['ActualQtr2']
Totals Actual for Qtr 3	Totals['Totals Actual for Qtr 3'] DM.CoaTotalsTbl['ActualQtr3']
Totals Actual for Qtr 4	Totals['Totals Actual for Qtr 4'] DM.CoaTotalsTbl['ActualQtr4']
Totals Actual for Year	Totals['Totals Actual for Year'] DM.CoaTotalsTbl['ActualYear']
Totals Blank	Totals['Totals Blank'] DM.CoaTotalsTbl['Blank1']
Totals Budget for Month 1	Totals['Totals Budget for Month 1'] DM.CoaTotalsTbl['BudgetMonth1']
Totals Budget for Month 10	Totals['Totals Budget for Month 10'] DM.CoaTotalsTbl['BudgetMonth10']
Totals Budget for Month 11	Totals['Totals Budget for Month 11'] DM.CoaTotalsTbl['BudgetMonth11']
Totals Budget for Month 12	Totals['Totals Budget for Month 12'] DM.CoaTotalsTbl['BudgetMonth12']
Totals Budget for Month 13	Totals['Totals Budget for Month 13'] DM.CoaTotalsTbl['BudgetMonth13']
Totals Budget for Month 14	Totals['Totals Budget for Month 14'] DM.CoaTotalsTbl['BudgetMonth14']
Totals Budget for Month 15	Totals['Totals Budget for Month 15'] DM.CoaTotalsTbl['BudgetMonth15']
Totals Budget for Month 16	Totals['Totals Budget for Month 16'] DM.CoaTotalsTbl['BudgetMonth16']
Totals Budget for Month 17	Totals['Totals Budget for Month 17'] DM.CoaTotalsTbl['BudgetMonth17']
Totals Budget for Month 18	Totals['Totals Budget for Month 18'] DM.CoaTotalsTbl['BudgetMonth18']
Totals Budget for Month 19	Totals['Totals Budget for Month 19'] DM.CoaTotalsTbl['BudgetMonth19']
Totals Budget for Month 2	Totals['Totals Budget for Month 2'] DM.CoaTotalsTbl['BudgetMonth2']
Totals Budget for Month 20	Totals['Totals Budget for Month 20'] DM.CoaTotalsTbl['BudgetMonth20']
Totals Budget for Month 21	Totals['Totals Budget for Month 21'] DM.CoaTotalsTbl['BudgetMonth21']
Totals Budget for Month 22	Totals['Totals Budget for Month 22'] DM.CoaTotalsTbl['BudgetMonth22']
Totals Budget for Month 23	Totals['Totals Budget for Month 23'] DM.CoaTotalsTbl['BudgetMonth23']
Totals Budget for Month 24	Totals['Totals Budget for Month 24'] DM.CoaTotalsTbl['BudgetMonth24']
Totals Budget for Month 25	Totals['Totals Budget for Month 25'] DM.CoaTotalsTbl['BudgetMonth25']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Totals Budget for Month 26	Totals['Totals Budget for Month 26'] DM.CoaTotalsTbl['BudgetMonth26']
Totals Budget for Month 27	Totals['Totals Budget for Month 27'] DM.CoaTotalsTbl['BudgetMonth27']
Totals Budget for Month 28	Totals['Totals Budget for Month 28'] DM.CoaTotalsTbl['BudgetMonth28']
Totals Budget for Month 29	Totals['Totals Budget for Month 29'] DM.CoaTotalsTbl['BudgetMonth29']
Totals Budget for Month 3	Totals['Totals Budget for Month 3'] DM.CoaTotalsTbl['BudgetMonth3']
Totals Budget for Month 30	Totals['Totals Budget for Month 30'] DM.CoaTotalsTbl['BudgetMonth30']
Totals Budget for Month 31	Totals['Totals Budget for Month 31'] DM.CoaTotalsTbl['BudgetMonth31']
Totals Budget for Month 32	Totals['Totals Budget for Month 32'] DM.CoaTotalsTbl['BudgetMonth32']
Totals Budget for Month 33	Totals['Totals Budget for Month 33'] DM.CoaTotalsTbl['BudgetMonth33']
Totals Budget for Month 34	Totals['Totals Budget for Month 34'] DM.CoaTotalsTbl['BudgetMonth34']
Totals Budget for Month 35	Totals['Totals Budget for Month 35'] DM.CoaTotalsTbl['BudgetMonth35']
Totals Budget for Month 36	Totals['Totals Budget for Month 36'] DM.CoaTotalsTbl['BudgetMonth36']
Totals Budget for Month 4	Totals['Totals Budget for Month 4'] DM.CoaTotalsTbl['BudgetMonth4']
Totals Budget for Month 5	Totals['Totals Budget for Month 5'] DM.CoaTotalsTbl['BudgetMonth5']
Totals Budget for Month 6	Totals['Totals Budget for Month 6'] DM.CoaTotalsTbl['BudgetMonth6']
Totals Budget for Month 7	Totals['Totals Budget for Month 7'] DM.CoaTotalsTbl['BudgetMonth7']
Totals Budget for Month 8	Totals['Totals Budget for Month 8'] DM.CoaTotalsTbl['BudgetMonth8']
Totals Budget for Month 9	Totals['Totals Budget for Month 9'] DM.CoaTotalsTbl['BudgetMonth9']
Totals Budget for Qtr 1	Totals['Totals Budget for Qtr 1'] DM.CoaTotalsTbl['BudgetQtr1']
Totals Budget for Qtr 10	Totals['Totals Budget for Qtr 10'] DM.CoaTotalsTbl['BudgetQtr10']
Totals Budget for Qtr 11	Totals['Totals Budget for Qtr 11'] DM.CoaTotalsTbl['BudgetQtr11']
Totals Budget for Qtr 12	Totals['Totals Budget for Qtr 12'] DM.CoaTotalsTbl['BudgetQtr12']
Totals Budget for Qtr 2	Totals['Totals Budget for Qtr 2'] DM.CoaTotalsTbl['BudgetQtr2']
Totals Budget for Qtr 3	Totals['Totals Budget for Qtr 3'] DM.CoaTotalsTbl['BudgetQtr3']
Totals Budget for Qtr 4	Totals['Totals Budget for Qtr 4'] DM.CoaTotalsTbl['BudgetQtr4']
Totals Budget for Qtr 5	Totals['Totals Budget for Qtr 5'] DM.CoaTotalsTbl['BudgetQtr5']
Totals Budget for Qtr 6	Totals['Totals Budget for Qtr 6'] DM.CoaTotalsTbl['BudgetQtr6']
Totals Budget for Qtr 7	Totals['Totals Budget for Qtr 7'] DM.CoaTotalsTbl['BudgetQtr7']
Totals Budget for Qtr 8	Totals['Totals Budget for Qtr 8'] DM.CoaTotalsTbl['BudgetQtr8']
Totals Budget for Qtr 9	Totals['Totals Budget for Qtr 9'] DM.CoaTotalsTbl['BudgetQtr9']
Totals Budget for Year 2	Totals['Totals Budget for Year 2'] DM.CoaTotalsTbl['BudgetYear2']
Totals Budget for Year 3	Totals['Totals Budget for Year 3'] DM.CoaTotalsTbl['BudgetYear3']
Totals Budget for Year	Totals['Totals Budget for Year'] DM.CoaTotalsTbl['BudgetYear']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Totals Budget Remaining	Totals['Totals Budget Remaining'] DM.CoaTotalsTbl['BudgetRemaining']
Totals Column 1	Totals['Totals Column 1'] DM.CoaTotalsTbl['Column1']
Totals Column 2	Totals['Totals Column 2'] DM.CoaTotalsTbl['Column2']
Totals Column 3	Totals['Totals Column 3'] DM.CoaTotalsTbl['Column3']
Totals Current Amount	Totals['Totals Current Amount'] DM.CoaTotalsTbl['CurrentAmount']
Totals Current Budget	Totals['Totals Current Budget'] DM.CoaTotalsTbl['CurrentBudget']
Totals Current Name	Totals['Totals Current Name'] DM.CoaTotalsTbl['CurrentName']
Totals Last Year Amount	Totals['Totals Last Year Amount'] DM.CoaTotalsTbl['LastYearAmount']
Totals Last Year Budget	Totals['Totals Last Year Budget'] DM.CoaTotalsTbl['LastYearBudget']
Totals Last Year Column 1	Totals['Totals Last Year Column 1'] DM.CoaTotalsTbl['LastYearColumn1']
Totals Last Year Column 2	Totals['Totals Last Year Column 2'] DM.CoaTotalsTbl['LastYearColumn2']
Totals Last Year Column 3	Totals['Totals Last Year Column 3'] DM.CoaTotalsTbl['LastYearColumn3']
Totals Last Year Name	Totals['Totals Last Year Name'] DM.CoaTotalsTbl['LastYearName']
Totals Last Year YTD Amount	Totals['Totals Last Year YTD Amount'] DM.CoaTotalsTbl['LastYearYTDAmount']
Totals Last Year YTD Budget	Totals['Totals Last Year YTD Budget'] DM.CoaTotalsTbl['LastYearYTDBudget']
Totals Pct Current Amount	Totals['Totals Pct Current Amount'] DM.CoaTotalsTbl['PctCurrentAmount']
Totals Pct Current Budget	Totals['Totals Pct Current Budget'] DM.CoaTotalsTbl['PctCurrentBudget']
Totals Pct Last Year Amount	Totals['Totals Pct Last Year Amount'] DM.CoaTotalsTbl['PctLastYearAmount']
Totals Pct Last Year Budget	Totals['Totals Pct Last Year Budget'] DM.CoaTotalsTbl['PctLastYearBudget']
Totals Pct Last Year YTD Amount	Totals['Totals Pct Last Year YTD Amount'] DM.CoaTotalsTbl['PctLastYearYTDamt']
Totals Pct Last Year YTD Budget	Totals['Totals Pct Last Year YTD Budget'] DM.CoaTotalsTbl['PctLastYearYTDBud']
Totals Pct Total Budget	Totals['Totals Pct Total Budget'] DM.CoaTotalsTbl['PctTotalBudget']
Totals Pct Var Current to Budget	Totals['Totals Pct Var Current to Budget'] DM.CoaTotalsTbl['PctVarCurrentBud']
Totals Pct Var Current to Last Year	Totals['Totals Pct Var Current to Last Year'] DM.CoaTotalsTbl['PctVarCurrentLY']
Totals Pct Var YTD to Budget YTD	Totals['Totals Pct Var YTD to Budget YTD'] DM.CoaTotalsTbl['PctVarYTDBudgetYTD']
Totals Pct Var YTD to Last Year YTD	Totals['Totals Pct Var YTD to Last Year YTD'] DM.CoaTotalsTbl['PctVarYTDLYYTD']
Totals Pct Var YTD to Total Budget	Totals['Totals Pct Var YTD to Total Budget'] DM.CoaTotalsTbl['PctVarYTDTotalBud']
Totals Pct YTD Amount	Totals['Totals Pct YTD Amount'] DM.CoaTotalsTbl['PctYTDAmount']
Totals Pct YTD Budget	Totals['Totals Pct YTD Budget'] DM.CoaTotalsTbl['PctYTDBudget']
Totals Pct Of Budget	Totals['Totals Pct Of Budget'] DM.CoaTotalsTbl['PctOfBudget']
Totals Pct Raw Accrual	Totals['Totals Pct Raw Accrual'] DM.CoaTotalsTbl['PctRawAccrual']
Totals Pct Raw Cash	Totals['Totals Pct Raw Cash'] DM.CoaTotalsTbl['PctRawCash']
Totals Pct Raw Encumbrance	Totals['Totals Pct Raw Encumbrance'] DM.CoaTotalsTbl['PctRawEncumbrance']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Totals Pct Second Year Amount	Totals['Totals Pct Second Year Amount'] DM.CoaTotalsTbl['PctSecYearAmt']
Totals Pct Second Year YTD Amount	Totals['Totals Pct Second Year YTD Amount'] DM.CoaTotalsTbl['PctSecYearYTDamt']
Totals Pct YTD Raw Accrual	Totals['Totals Pct YTD Raw Accrual'] DM.CoaTotalsTbl['PctYTDRawAccrual']
Totals Pct YTD Raw Cash	Totals['Totals Pct YTD Raw Cash'] DM.CoaTotalsTbl['PctYTDRawCash']
Totals Pct YTD Raw Encumbrance	Totals['Totals Pct YTD Raw Encumbrance'] DM.CoaTotalsTbl['PctYTDRawEncum']
Totals Prior YTD Amount	Totals['Totals Prior YTD Amount'] DM.CoaTotalsTbl['PriorYTDAmount']
Totals Raw Accrual	Totals['Totals Raw Accrual'] DM.CoaTotalsTbl['RawAccrual']
Totals Raw Cash	Totals['Totals Raw Cash'] DM.CoaTotalsTbl['RawCash']
Totals Raw Encumbrance	Totals['Totals Raw Encumbrance'] DM.CoaTotalsTbl['RawEncumbrance']
Totals Second Year Amount	Totals['Totals Second Year Amount'] DM.CoaTotalsTbl['SecYearAmt']
Totals Second Year YTD Amount	Totals['Totals Second Year YTD Amount'] DM.CoaTotalsTbl['SecYearYTDamt']
Totals Total Budget	Totals['Totals Total Budget'] DM.CoaTotalsTbl['TotalBudget']
Totals Variance Current to Budget	Totals['Totals Variance Current to Budget'] DM.CoaTotalsTbl['VarCurrentBudget']
Totals Variance Current to Last Year	Totals['Totals Variance Current to Last Year'] DM.CoaTotalsTbl['VarCurrentLastYear']
Totals Variance for Month 1	Totals['Totals Variance for Month 1'] DM.CoaTotalsTbl['VarianceMonth1']
Totals Variance for Month 10	Totals['Totals Variance for Month 10'] DM.CoaTotalsTbl['VarianceMonth10']
Totals Variance for Month 11	Totals['Totals Variance for Month 11'] DM.CoaTotalsTbl['VarianceMonth11']
Totals Variance for Month 12	Totals['Totals Variance for Month 12'] DM.CoaTotalsTbl['VarianceMonth12']
Totals Variance for Month 2	Totals['Totals Variance for Month 2'] DM.CoaTotalsTbl['VarianceMonth2']
Totals Variance for Month 3	Totals['Totals Variance for Month 3'] DM.CoaTotalsTbl['VarianceMonth3']
Totals Variance for Month 4	Totals['Totals Variance for Month 4'] DM.CoaTotalsTbl['VarianceMonth4']
Totals Variance for Month 5	Totals['Totals Variance for Month 5'] DM.CoaTotalsTbl['VarianceMonth5']
Totals Variance for Month 6	Totals['Totals Variance for Month 6'] DM.CoaTotalsTbl['VarianceMonth6']
Totals Variance for Month 7	Totals['Totals Variance for Month 7'] DM.CoaTotalsTbl['VarianceMonth7']
Totals Variance for Month 8	Totals['Totals Variance for Month 8'] DM.CoaTotalsTbl['VarianceMonth8']
Totals Variance for Month 9	Totals['Totals Variance for Month 9'] DM.CoaTotalsTbl['VarianceMonth9']
Totals Variance for Qtr 1	Totals['Totals Variance for Qtr 1'] DM.CoaTotalsTbl['VarianceQtr1']
Totals Variance for Qtr 2	Totals['Totals Variance for Qtr 2'] DM.CoaTotalsTbl['VarianceQtr2']
Totals Variance for Qtr 3	Totals['Totals Variance for Qtr 3'] DM.CoaTotalsTbl['VarianceQtr3']
Totals Variance for Qtr 4	Totals['Totals Variance for Qtr 4'] DM.CoaTotalsTbl['VarianceQtr4']
Totals Variance for Year	Totals['Totals Variance for Year'] DM.CoaTotalsTbl['VarianceYear']
Totals Variance YTD to Budget YTD	Totals['Totals Variance YTD to Budget YTD'] DM.CoaTotalsTbl['VarYTDBudgetYTD']
Totals Variance YTD to Last Year YTD	Totals['Totals Variance YTD to Last Year YTD'] DM.CoaTotalsTbl['VarYTDLastYearYTD']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Totals Variance YTD to Total Budget	Totals['Totals Variance YTD to Total Budget'] DM.CoaTotalsTbl['VarYTDTotalBudget']
Totals YTD Amount	Totals['Totals YTD Amount'] DM.CoaTotalsTbl['YTDAmount']
Totals YTD Budget	Totals['Totals YTD Budget'] DM.CoaTotalsTbl['YTDBudget']
Totals YTD Raw Accrual	Totals['Totals YTD Raw Accrual'] DM.CoaTotalsTbl['YTDRawAccrual']
Totals YTD Raw Cash	Totals['Totals YTD Raw Cash'] DM.CoaTotalsTbl['YTDRawCash']
Totals YTD Raw Encumbrance	Totals['Totals YTD Raw Encumbrance'] DM.CoaTotalsTbl['YTDRawEncumbrance']
Trn Amount Not Voided	Trn['Trn Amount Not Voided'] DM.TrnTbl['AmountNotVoided']
Trn Authorized By	Trn['Trn Authorized By'] DM.TrnTbl['AuthorizedBy']
Trn Authorized	Trn['Trn Authorized'] DM.TrnTbl['Authorized']
Trn Batch Number	Trn['Trn Batch Number'] DM.TrnTbl['Batch']
Trn Blank1	Trn['Trn Blank1'] DM.TrnTbl['Blank1']
Trn Blank2	Trn['Trn Blank2'] DM.TrnTbl['Blank2']
Trn Blank3	Trn['Trn Blank3'] DM.TrnTbl['Blank3']
Trn Blank4	Trn['Trn Blank4'] DM.TrnTbl['Blank4']
Trn Cash Account Name	Trn['Trn Cash Account Name'] DM.TrnTbl['CashAcctName']
Trn Cash Account Number	Trn['Trn Cash Account Number'] DM.TrnTbl['CashAcctNum']
Trn Check Amount	Trn['Trn Check Amount'] DM.TrnTbl['Amount']
Trn Check Number	Trn['Trn Check Number'] DM.TrnTbl['CheckNumber']
Trn Corrected Payee Name	Trn['Trn Corrected Payee Name'] DM.TrnTbl['CorrectedPayeeName']
Trn Date Changed	Trn['Trn Date Changed'] DM.TrnTbl['DateChanged']
Trn Date Created	Trn['Trn Date Created'] DM.TrnTbl['DateCreated']
Trn Date	Trn['Trn Date'] DM.TrnTbl['TDate']
Trn Discount Amt	Trn['Trn Discount Amt'] DM.TrnTbl['DiscountAmt']
Trn Discount Date	Trn['Trn Discount Date'] DM.TrnTbl['DiscountDate']
Trn Discount Pct	Trn['Trn Discount Pct'] DM.TrnTbl['DiscountPct']
Trn Discount Terms	Trn['Trn Discount Terms'] DM.TrnTbl['DiscountTerms']
Trn Discount Taken	Trn['Trn Discount Taken'] DM.TrnTbl['DiscountTaken']
Trn Due Date	Trn['Trn Due Date'] DM.TrnTbl['DueDate']
Trn Electronic Payment	Trn['Trn Electronic Payment'] DM.TrnTbl['ElectronicPayment']
Trn Emp Department	Trn['Trn Emp Department'] DM.TrnTbl['Department']
Trn Employer	Trn['Trn Employer'] DM.TrnTbl['Employer']
Trn Emp Position	Trn['Trn Emp Position'] DM.TrnTbl['Position']
Trn Hours	Trn['Trn Hours'] DM.TrnTbl['Hours']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Trn Invoice Amount	Trn['Trn Invoice Amount'] DM.TrnTbl['InvoiceAmount']
Trn Invoice Date	Trn['Trn Invoice Date'] DM.TrnTbl['InvoiceDate']
Trn Is Reconciled	Trn['Trn Is Reconciled'] DM.TrnTbl['IsReconciled']
Trn Notes	Trn['Trn Notes'] DM.TrnTbl['Memo']
Trn Overtime 1 Hours	Trn['Trn Overtime 1 Hours'] DM.TrnTbl['Overtime1']
Trn Overtime 2 Hours	Trn['Trn Overtime 2 Hours'] DM.TrnTbl['Overtime2']
Trn Partial Payment	Trn['Trn Partial Payment'] DM.TrnTbl['PartialPayment']
Trn Period	Trn['Trn Period'] DM.TrnTbl['Period']
Trn Purchase Order	Trn['Trn Purchase Order'] DM.TrnTbl['PurchaseOrder']
Trn Reconcile Date	Trn['Trn Reconcile Date'] DM.TrnTbl['ReconcileLabel']
Trn Reference	Trn['Trn Reference'] DM.TrnTbl['Reference']
Trn Regular Hours	Trn['Trn Regular Hours'] DM.TrnTbl['RegularHours']
Trn Reissue of Old	Trn['Trn Reissue of Old'] DM.TrnTbl['ReissueOfOld']
Trn Reverse of Old	Trn['Trn Reverse of Old'] DM.TrnTbl['ReverseOfOld']
Trn Special Hours	Trn['Trn Special Hours'] DM.TrnTbl['SpecialHours']
Trn Terms	Trn['Trn Terms'] DM.TrnTbl['Terms']
Trn Type	Trn['Trn Type'] DM.TrnTbl['Type']
Trn Type Short	Trn['Trn Type Short'] DM.TrnTbl['TypeShort']
Trn Vendor Abbr	Trn['Trn Vendor Abbr'] DM.TrnTbl['PayeeSourceAbbr']
Trn Vendor Name	Trn['Trn Vendor Name'] DM.TrnTbl['PayeeSourceName']
Trn Voided or Reversed	Trn['Trn Voided or Reversed'] DM.TrnTbl['VoidedOrReversed']
Trn Voided	Trn['Trn Voided'] DM.TrnTbl['Voided']
Ven Abbr	Ven['Ven Abbr'] DM.VenTbl['Abbr']
Ven Account Number	Ven['Ven Account Number'] DM.VenTbl['CustomerAcctNum']
Ven Blank	Ven['Ven Blank'] DM.VenTbl['Blank']
Ven Corrected Name	Ven['Ven Corrected Name'] DM.VenTbl['CorrectedName']
Ven Date Changed	Ven['Ven Date Changed'] DM.VenTbl['DateChanged']
Ven Date Created	Ven['Ven Date Created'] DM.VenTbl['DateCreated']
Ven Date W9 Requested	Ven['Ven Date W9 Requested'] DM.VenTbl['W9Requested']
Ven Discount Terms	Ven['Ven Discount Terms'] DM.VenTbl['calcDiscountTerms']
Ven Doing Business as Name	Ven['Ven Doing Business as Name'] DM.VenTbl['DBAName']
Ven Federal ID Number	Ven['Ven Federal ID Number'] DM.VenTbl['FedID']
Ven ID Number	Ven['Ven ID Number'] DM.VenTbl['ID']

## Ledger, Ledger/Payroll Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ven Inactive	Ven['Ven Inactive'] DM.VenTbl['Inactive']
Ven Name	Ven['Ven Name'] DM.VenTbl['Name']
Ven Organization Type	Ven['Ven Organization Type'] DM.VenTbl['calcVendorType']
Ven Receives 1099	Ven['Ven Receives 1099'] DM.VenTbl['Receives1099']
Ven Remarks	Ven['Ven Remarks'] DM.VenTbl['Remarks']
Ven Since	Ven['Ven Since'] DM.VenTbl['VendorSince']
Ven Terms	Ven['Ven Terms'] DM.VenTbl['calcTerms']
Ven Unique ID	Ven['Ven Unique ID'] DM.VenTbl['VenRec']
Ven W9 On File	Ven['Ven W9 On File'] DM.VenTbl['W9OnFile']
Ven Web Page	Ven['Ven Web Page'] DM.VenTbl['WebPage']
Ven Address Address 1	VenAddress['Ven Address Address 1'] DM.VContactTbl['Address']
Ven Address Address 2	VenAddress['Ven Address Address 2'] DM.VContactTbl['Address2']
Ven Address Address Block	VenAddress['Ven Address Address Block'] DM.VContactTbl['AddressBlock']
Ven Address City Only	VenAddress['Ven Address City Only'] DM.VContactTbl['CityOnly']
Ven Address City/State	VenAddress['Ven Address City/State'] DM.VContactTbl['CityState']
Ven Address City/State/Zip	VenAddress['Ven Address City/State/Zip'] DM.VContactTbl['CityZip']
Ven Address E Mail	VenAddress['Ven Address E Mail'] DM.VContactTbl['EMail']
Ven Address Mailing Name	VenAddress['Ven Address Mailing Name'] DM.VContactTbl['Mailing']
Ven Address Name	VenAddress['Ven Address Name'] DM.VContactTbl['Name']
Ven Address Notes	VenAddress['Ven Address Notes'] DM.VContactTbl['Notes']
Ven Address Phone List	VenAddress['Ven Address Phone List'] DM.VContactTbl['PhoneList']
Ven Address Position	VenAddress['Ven Address Position'] DM.VContactTbl['Position']
Ven Address Salutation	VenAddress['Ven Address Salutation'] DM.VContactTbl['Salutation']
Ven Address State Only	VenAddress['Ven Address State Only'] DM.VContactTbl['StateOnly']
Ven Address Zip	VenAddress['Ven Address Zip'] DM.VContactTbl['Zip']
Ven Phone Number	VenPhone['Ven Phone Number'] DM.VCPhoneTbl['NNumber']
Ven Phone Type	VenPhone['Ven Phone Type'] DM.VCPhoneTbl['PhoneType']
Ven Phone Unlisted	VenPhone['Ven Phone Unlisted'] DM.VCPhoneTbl['Unlisted']
Ven Phone X Number	VenPhone['Ven Phone X Number'] DM.VCPhoneTbl['XNumber']
Ven Totals Balance	VenTotals['Ven Totals Balance'] DM.VTotalsTbl['Balance']
Ven Totals Calendar YTD	VenTotals['Ven Totals Calendar YTD'] DM.VTotalsTbl['CalYTD']
Ven Totals Fiscal YTD	VenTotals['Ven Totals Fiscal YTD'] DM.VTotalsTbl['FiscalYTD']
Ven Totals Last Amount	VenTotals['Ven Totals Last Amount'] DM.VTotalsTbl['LastAmount']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Ven Totals Last Date	VenTotals['Ven Totals Last Date'] DM.VTotalsTbl['LastDate']
Ven Totals Last Reference	VenTotals['Ven Totals Last Reference'] DM.VTotalsTbl['LastReference']
Ven Totals Next Amount	VenTotals['Ven Totals Next Amount'] DM.VTotalsTbl['NextAmount']
Ven Totals Next Date	VenTotals['Ven Totals Next Date'] DM.VTotalsTbl['NextDate']
Ven Totals Next Reference	VenTotals['Ven Totals Next Reference'] DM.VTotalsTbl['NextReference']



## Facility Scheduler Field Names

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Evt All Req Met	Evt['Evt All Req Met'] DM.Evt['AllReqMet']
Evt Cleanup	Evt['Evt Cleanup'] DM.Evt['Cleanup']
Evt Date Altered	Evt['Evt Date Altered'] DM.Evt['DateAltered']
Evt Date Changed	Evt['Evt Date Changed'] DM.Evt['DateChanged']
Evt Date Created	Evt['Evt Date Created'] DM.Evt['DateCreated']
Evt Date	Evt['Evt Date'] DM.Evt['Date']
Evt End Time	Evt['Evt End Time'] DM.Evt['EndTime']
Evt Fac Name	Evt['Evt Fac Name'] DM.Evt['FacName']
Evt Inactive	Evt['Evt Inactive'] DM.Evt['Inactive']
Evt Name	Evt['Evt Name'] DM.Evt['Name']
Evt Org Name	Evt['Evt Org Name'] DM.Evt['OrgName']
Evt Parish	Evt['Evt Parish'] DM.Evt['Category1']
Evt Remarks	Evt['Evt Remarks'] DM.Evt['Remarks']
Evt Setup	Evt['Evt Setup'] DM.Evt['Setup']
Evt Start Time	Evt['Evt Start Time'] DM.Evt['StartTime']
Evt Total Balance	Evt['Evt Total Balance'] DM.Evt['TotalBalance']
Evt Total Credit	Evt['Evt Total Credit'] DM.Evt['TotalCredit']
Evt Total Due	Evt['Evt Total Due'] DM.Evt['TotalDue']
Evt Total Paid	Evt['Evt Total Paid'] DM.Evt['TotalPaid']
Evt Wedding	Evt['Evt Wedding'] DM.Evt['Category2']
Evt Chg Pmt Amount	EvtChgPmt['Evt Chg Pmt Amount'] DM.EvtChgPmt['Amount']
Evt Chg Pmt Check Number	EvtChgPmt['Evt Chg Pmt Check Number'] DM.EvtChgPmt['CheckNumber']
Evt Chg Pmt Date	EvtChgPmt['Evt Chg Pmt Date'] DM.EvtChgPmt['Date']
Evt Chg Pmt Description	EvtChgPmt['Evt Chg Pmt Description'] DM.EvtChgPmt['Description']
Fac Abbr	Fac['Fac Abbr'] DM.Fac['Abbr']
Fac Address	Fac['Fac Address'] DM.Fac['Address']
Fac Chairs	Fac['Fac Chairs'] DM.Fac['Feature3N']
Fac Charge Type	Fac['Fac Charge Type'] DM.Fac['ChargeType']
Fac City/State	Fac['Fac City/State'] DM.Fac['City']
Fac City/State/Zip	Fac['Fac City/State/Zip'] DM.Fac['CityStateZip']
Fac Date Changed	Fac['Fac Date Changed'] DM.Fac['DateChanged']
Fac Date Created	Fac['Fac Date Created'] DM.Fac['DateCreated']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Fac Inactive	Fac['Fac Inactive'] DM.Fac['Inactive']
Fac Name ( Group)	Fac['Fac Name'] DM.Fac['Name']
Fac Name	Fac['Fac Name'] DM.Fac['NameFixed']
Fac Phone	Fac['Fac Phone'] DM.Fac['NPhone']
Fac Remarks	Fac['Fac Remarks'] DM.Fac['Remarks']
Fac Seating Capacity	Fac['Fac Seating Capacity'] DM.Fac['Feature1N']
Fac Tables	Fac['Fac Tables'] DM.Fac['Feature2N']
Fac Unlisted	Fac['Fac Unlisted'] DM.Fac['PhoneUnlisted']
Fac Use Default Address	Fac['Fac Use Default Address'] DM.Fac['UseDefaultAddress']
Fac Vcr /Tv	Fac['Fac Vcr /Tv'] DM.Fac['Feature4B']
Fac X Unlisted	Fac['Fac X Unlisted'] DM.Fac['XPhone']
Fac Zip	Fac['Fac Zip'] DM.Fac['Zip']
Fac Charge Amount	FacCharge['Fac Charge Amount'] DM.FacCharge['Amount']
Fac Charge Description	FacCharge['Fac Charge Description'] DM.FacCharge['Description']
Fac Contact Address	FacContact['Fac Contact Address'] DM.FacContact['Address']
Fac Contact City/State	FacContact['Fac Contact City/State'] DM.FacContact['City']
Fac Contact City/State/Zip	FacContact['Fac Contact City/State/Zip'] DM.FacContact['CityStateZip']
Fac Contact E Mail	FacContact['Fac Contact E Mail'] DM.FacContact['EMail']
Fac Contact Mailing Name	FacContact['Fac Contact Mailing Name'] DM.FacContact['MailingName']
Fac Contact Name	FacContact['Fac Contact Name'] DM.FacContact['Name']
Fac Contact Phone List	FacContact['Fac Contact Phone List'] DM.FacContact['PhoneList']
Fac Contact Position	FacContact['Fac Contact Position'] DM.FacContact['Position']
Fac Contact Salutation	FacContact['Fac Contact Salutation'] DM.FacContact['Salutation']
Fac Contact Send No Mail	FacContact['Fac Contact Send No Mail'] DM.FacContact['SendNoMail']
Fac Contact Zip	FacContact['Fac Contact Zip'] DM.FacContact['Zip']
Fac Contact Phone Number	FacContactPhone['Fac Contact Phone Number'] DM.FacContactPhone['NNumber']
Fac Contact Phone Type	FacContactPhone['Fac Contact Phone Type'] DM.FacContactPhone['PhoneType']
Fac Contact Phone Unlisted	FacContactPhone['Fac Contact Phone Unlisted'] DM.FacContactPhone['Unlisted']
Fac Contact Phone X Unlisted	FacContactPhone['Fac Contact Phone X Unlisted'] DM.FacContactPhone['XNumber']
Group Facility Abbreviation	Group['Group Facility Abbreviation'] DM.Group['FacilityAbbr']
Group Facility Name	Group['Group Facility Name'] DM.Group['FacilityName']
Org Abbr	Org['Org Abbr'] DM.Org['Abbr']
Org Date Changed	Org['Org Date Changed'] DM.Org['DateChanged']

## Facility Scheduler Field Names

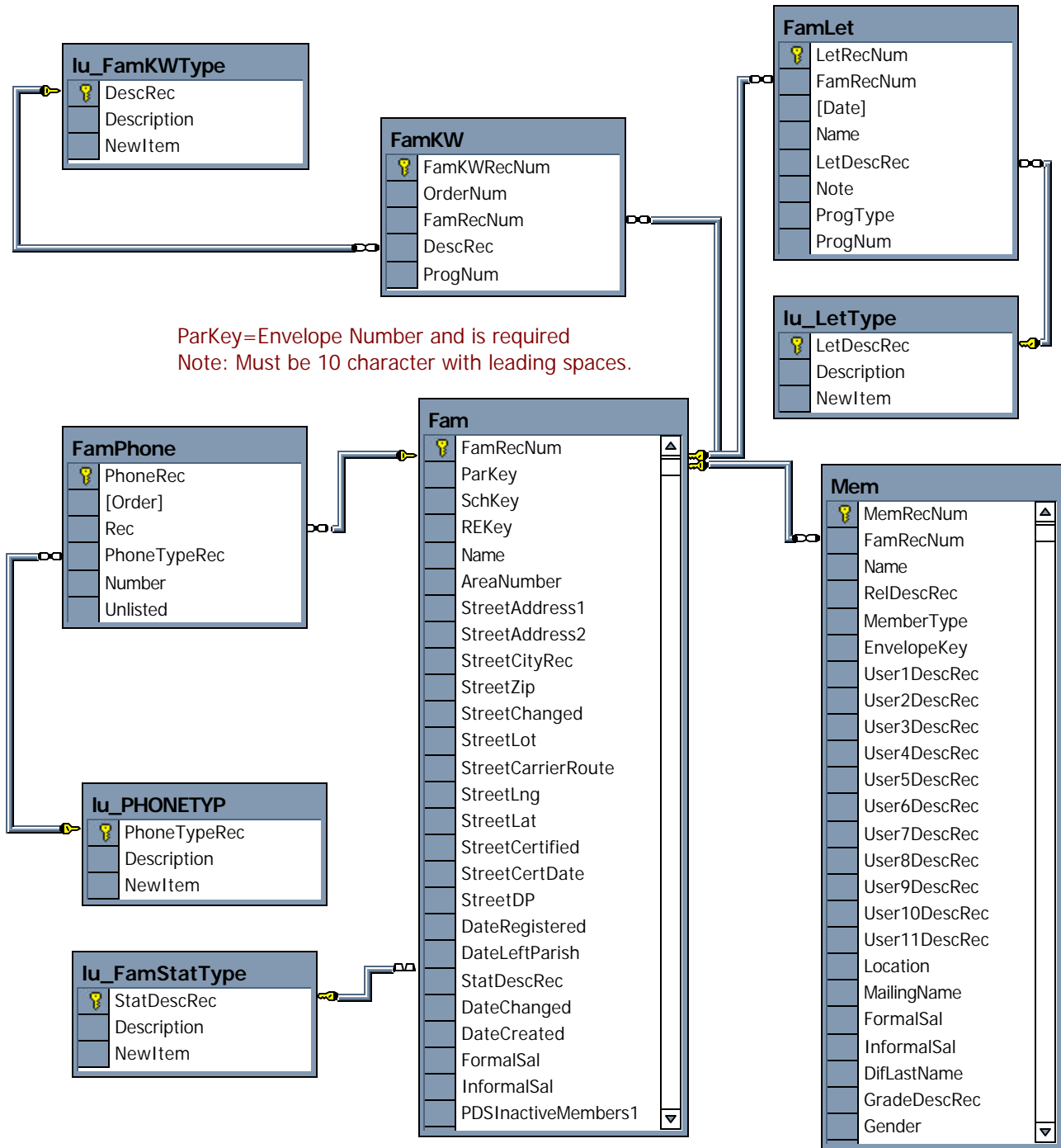
Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Org Date Created	Org['Org Date Created'] DM.Org['DateCreated']
Org Inactive	Org['Org Inactive'] DM.Org['Inactive']
Org Name	Org['Org Name'] DM.Org['Name']
Org Priority	Org['Org Priority'] DM.Org['Priority']
Org Remarks	Org['Org Remarks'] DM.Org['Remarks']
Org Total Balance	Org['Org Total Balance'] DM.Org['TotalBalance']
Org Total Credit	Org['Org Total Credit'] DM.Org['TotalCredit']
Org Total Due	Org['Org Total Due'] DM.Org['TotalDue']
Org Total Paid	Org['Org Total Paid'] DM.Org['TotalPaid']
Org Chg Pmt Amount	OrgChgPmt['Org Chg Pmt Amount'] DM.OrgChgPmt['Amount']
Org Chg Pmt Check Number	OrgChgPmt['Org Chg Pmt Check Number'] DM.OrgChgPmt['CheckNumber']
Org Chg Pmt Date	OrgChgPmt['Org Chg Pmt Date'] DM.OrgChgPmt['Date']
Org Chg Pmt Description	OrgChgPmt['Org Chg Pmt Description'] DM.OrgChgPmt['Description']
Org Contact Address	OrgContact['Org Contact Address'] DM.OrgContact['Address']
Org Contact City/State	OrgContact['Org Contact City/State'] DM.OrgContact['City']
Org Contact City/State/Zip	OrgContact['Org Contact City/State/Zip'] DM.OrgContact['CityStateZip']
Org Contact E Mail	OrgContact['Org Contact E Mail'] DM.OrgContact['EMail']
Org Contact Mailing Name	OrgContact['Org Contact Mailing Name'] DM.OrgContact['MailingName']
Org Contact Name	OrgContact['Org Contact Name'] DM.OrgContact['Name']
Org Contact Phone List	OrgContact['Org Contact Phone List'] DM.OrgContact['PhoneList']
Org Contact Position	OrgContact['Org Contact Position'] DM.OrgContact['Position']
Org Contact Salutation	OrgContact['Org Contact Salutation'] DM.OrgContact['Salutation']
Org Contact Send No Mail	OrgContact['Org Contact Send No Mail'] DM.OrgContact['SendNoMail']
Org Contact Zip	OrgContact['Org Contact Zip'] DM.OrgContact['Zip']
Org Contact Phone Number	OrgContactPhone['Org Contact Phone Number'] DM.OrgContactPhone['NNumber']
Org Contact Phone Type	OrgContactPhone['Org Contact Phone Type'] DM.OrgContactPhone['PhoneType']
Org Contact Phone Unlisted	OrgContactPhone['Org Contact Phone Unlisted'] DM.OrgContactPhone['Unlisted']
Org Contact Phone X Number	OrgContactPhone['Org Contact Phone X Number'] DM.OrgContactPhone['XNumber']
Pkg All Req Met	Pkg['Pkg All Req Met'] DM.Pkg['AllReqMet']
Pkg Date Changed	Pkg['Pkg Date Changed'] DM.Pkg['DateChanged']
Pkg Date Created	Pkg['Pkg Date Created'] DM.Pkg['DateCreated']
Pkg Inactive	Pkg['Pkg Inactive'] DM.Pkg['Inactive']
Pkg Name	Pkg['Pkg Name'] DM.Pkg['Name']

Field Names in Selections, Listings, Letters and Labels	Field Names in Embedded Code Field Names in Scripts
Pkg Remarks	Pkg['Pkg Remarks'] DM.Pkg['Remarks']
Pkg Total Balance	Pkg['Pkg Total Balance'] DM.Pkg['TotalBalance']
Pkg Total Credit	Pkg['Pkg Total Credit'] DM.Pkg['TotalCredit']
Pkg Total Due	Pkg['Pkg Total Due'] DM.Pkg['TotalDue']
Pkg Total Paid	Pkg['Pkg Total Paid'] DM.Pkg['TotalPaid']
Pkg Contact Address	PkgContact['Pkg Contact Address'] DM.PkgContact['Address']
Pkg Contact City/State	PkgContact['Pkg Contact City/State'] DM.PkgContact['City']
Pkg Contact City/State/Zip	PkgContact['Pkg Contact City/State/Zip'] DM.PkgContact['CityStateZip']
Pkg Contact E Mail	PkgContact['Pkg Contact E Mail'] DM.PkgContact['EMail']
Pkg Contact Mailing Name	PkgContact['Pkg Contact Mailing Name'] DM.PkgContact['MailingName']
Pkg Contact Name	PkgContact['Pkg Contact Name'] DM.PkgContact['Name']
Pkg Contact Phone List	PkgContact['Pkg Contact Phone List'] DM.PkgContact['PhoneList']
Pkg Contact Position	PkgContact['Pkg Contact Position'] DM.PkgContact['Position']
Pkg Contact Salutation	PkgContact['Pkg Contact Salutation'] DM.PkgContact['Salutation']
Pkg Contact Send No Mail	PkgContact['Pkg Contact Send No Mail'] DM.PkgContact['SendNoMail']
Pkg Contact Zip	PkgContact['Pkg Contact Zip'] DM.PkgContact['Zip']
Pkg Contact Phone Number	PkgContactPhone['Pkg Contact Phone Number'] DM.PkgContactPhone['NNumber']
Pkg Contact Phone Type	PkgContactPhone['Pkg Contact Phone Type'] DM.PkgContactPhone['PhoneType']
Pkg Contact Phone Unlisted	PkgContactPhone['Pkg Contact Phone Unlisted'] DM.PkgContactPhone['Unlisted']
Pkg Contact Phone X Unlisted	PkgContactPhone['Pkg Contact Phone X Unlisted'] DM.PkgContactPhone['XNumber']

(\* - Field can be used in listings, letters and labels but not selections)

## PDS Office Table Relationships

# Family Tables Links



ParKey=Envelope Number and is required  
Note: Must be 10 character with leading spaces.

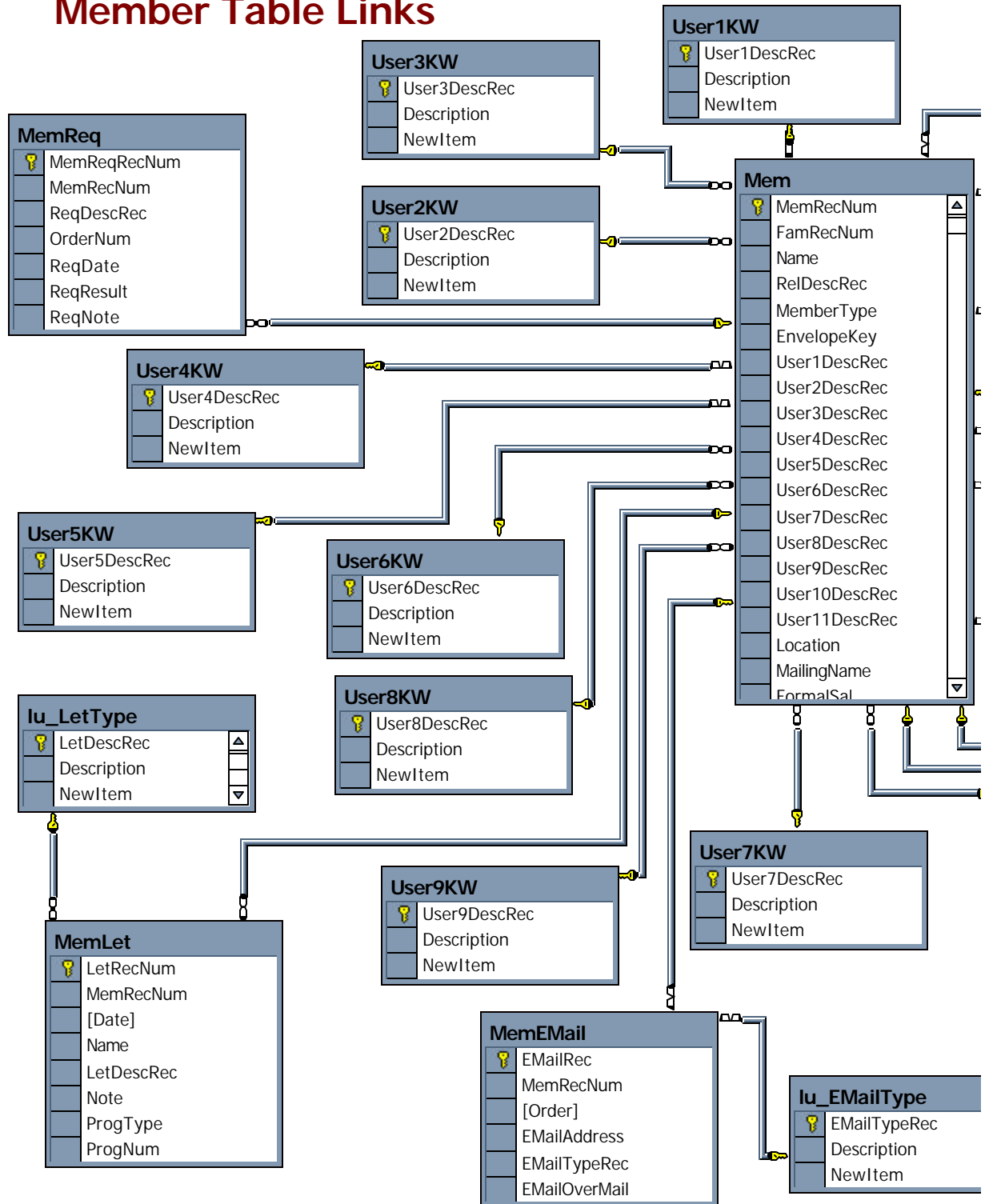
**NAME=Last,First\_(Spouse),Title,Suffix**

**Note: Include , if Suffix but no Title**

**Spouse=Last,First,Title**

**NOTE: SecondID must be 12 characters with leading spaces**

## Member Table Links

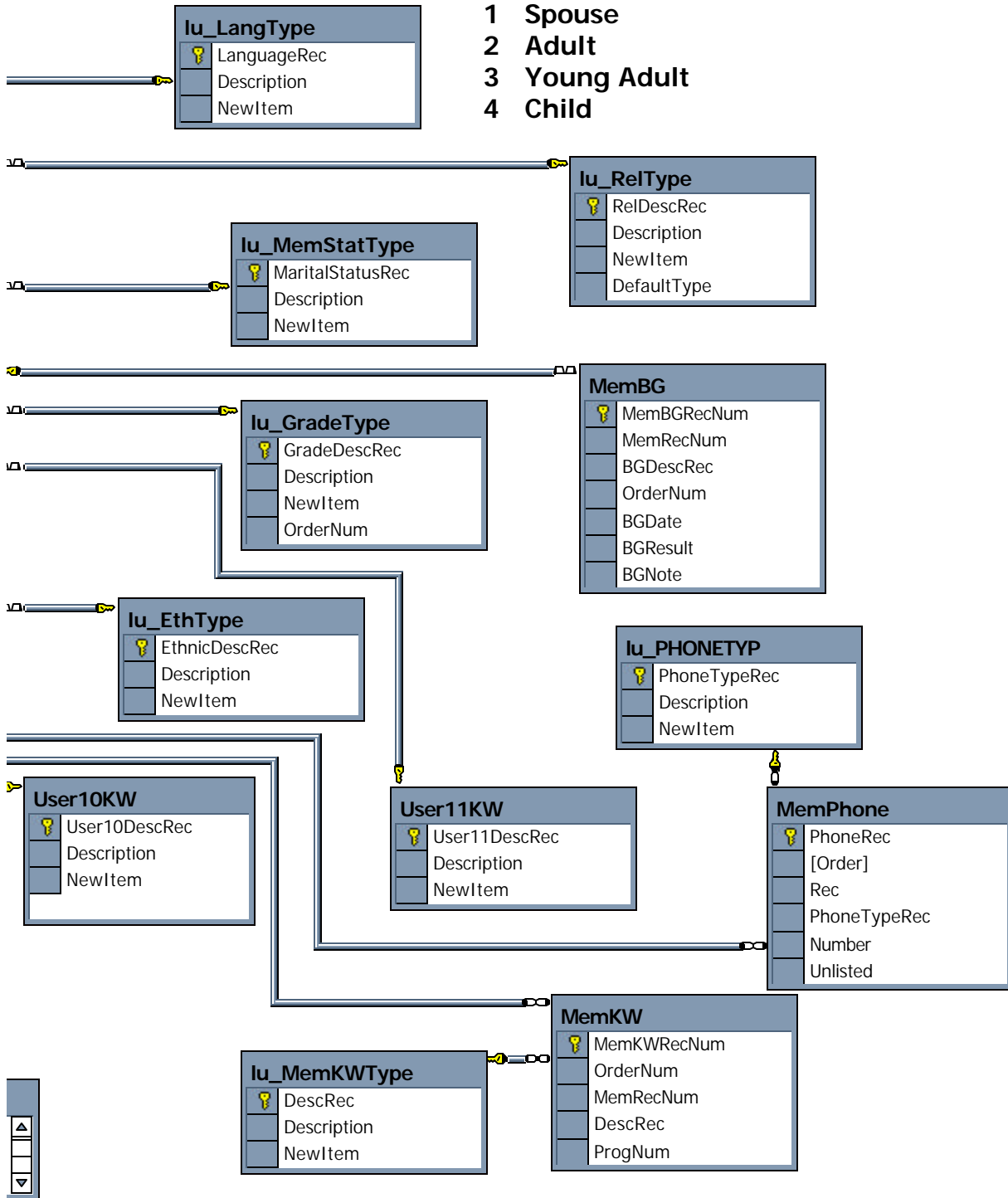


NAME=Last,First(Nickname)[Maiden Name],Title,Suffix

Not: Include , if they have a suffix but no title

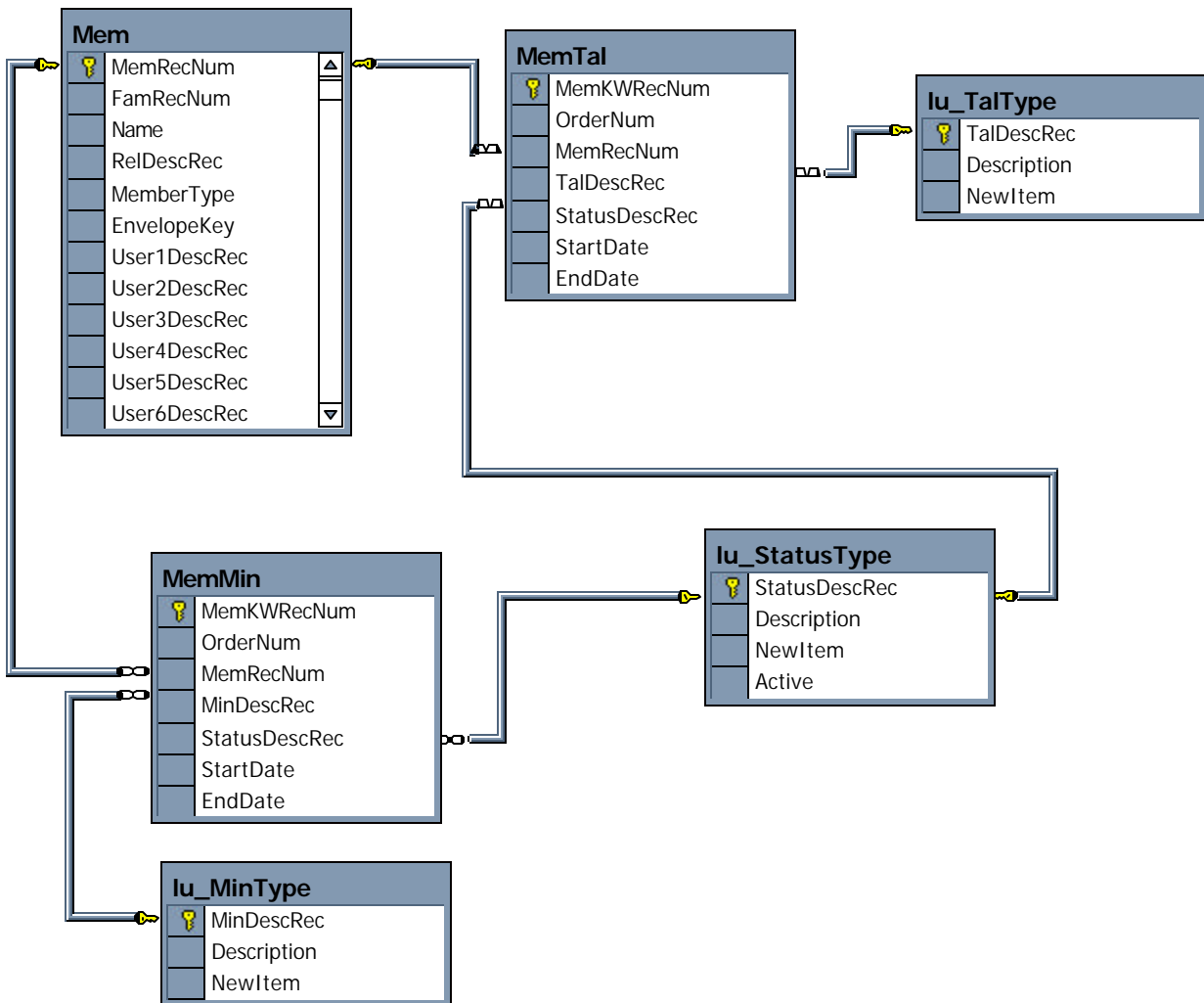
**Member Type  
Codes**

0	Head
1	Spouse
2	Adult
3	Young Adult
4	Child



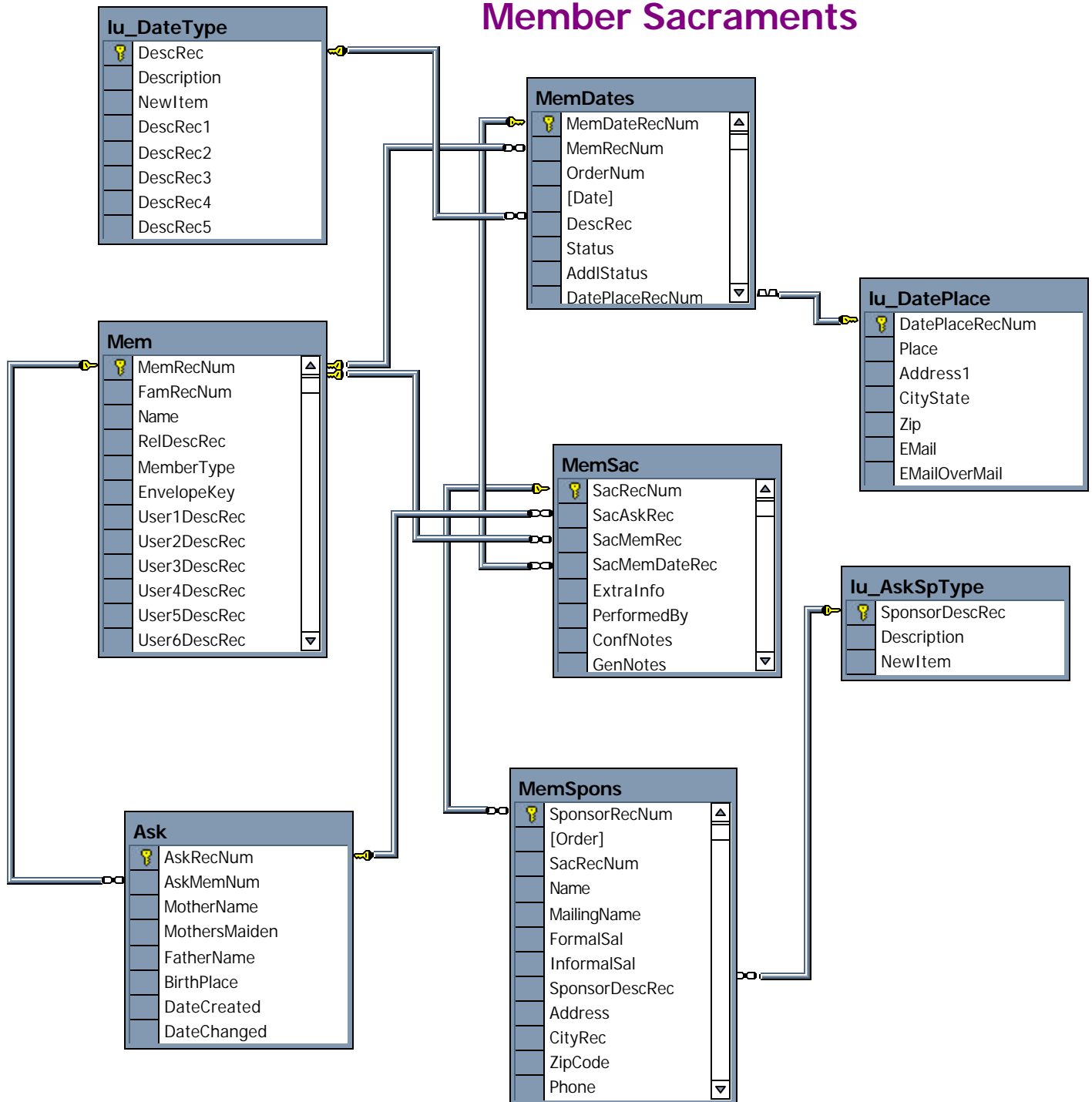
**Gender**  
**Male**  
**Female**

## Member Talents and Ministries

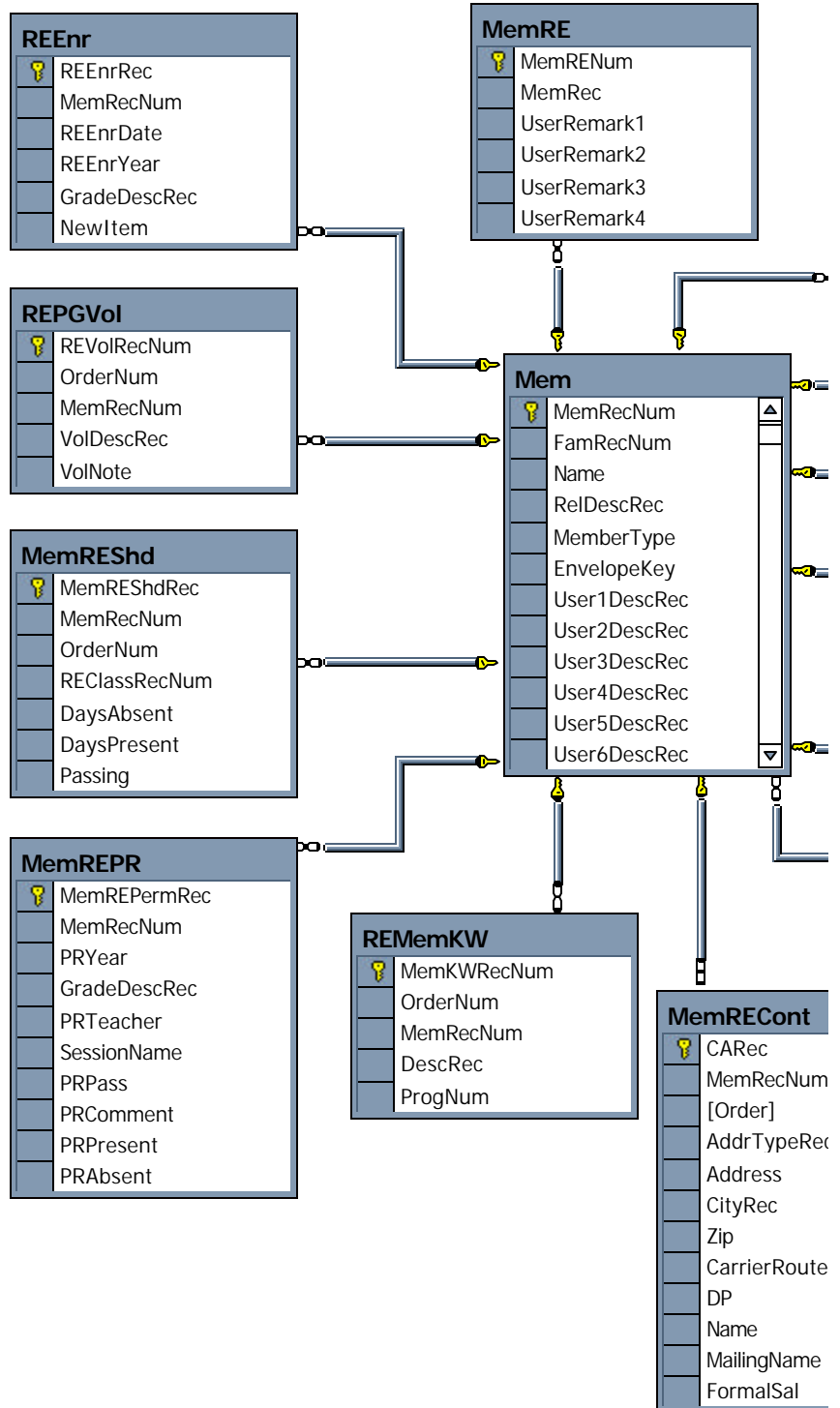


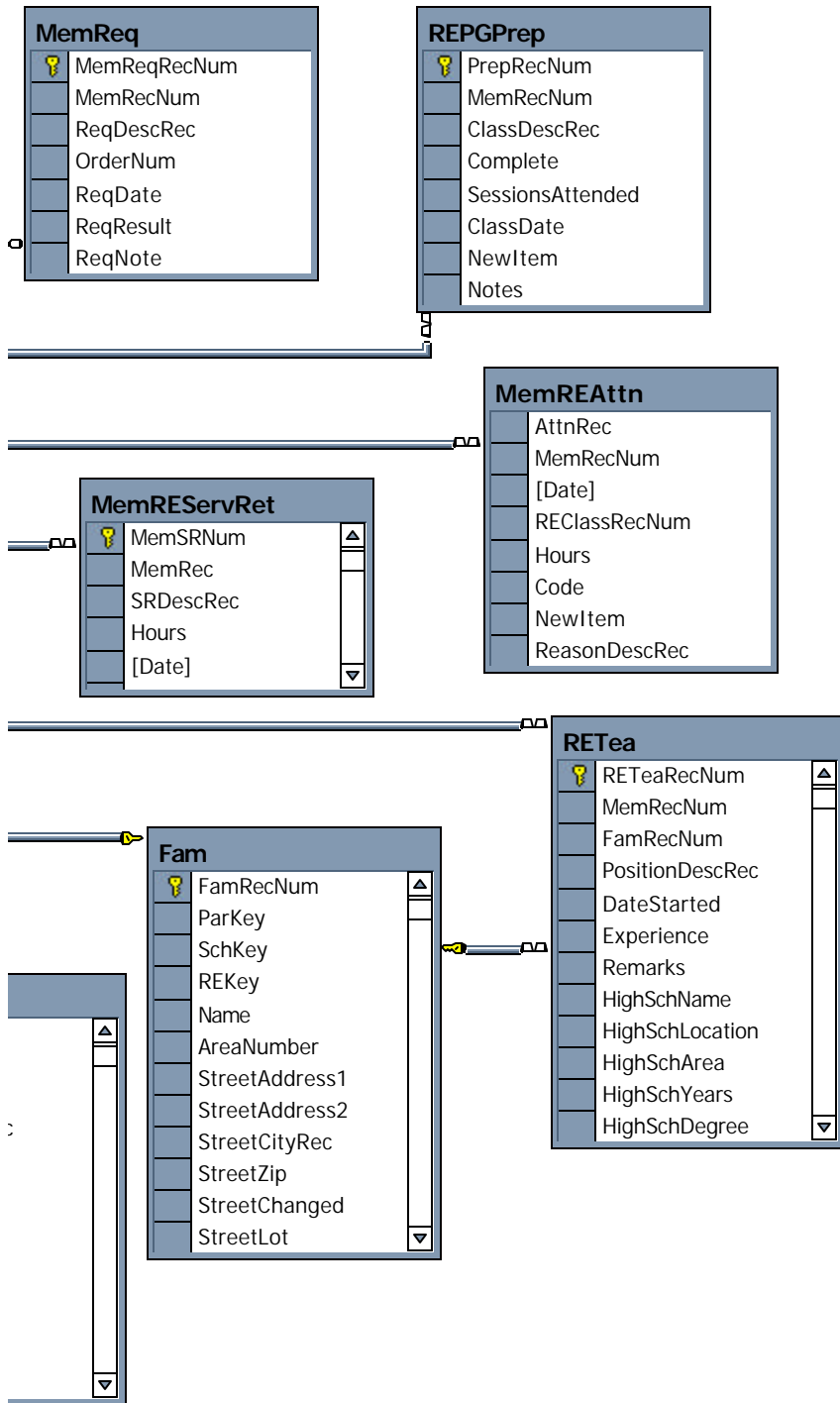


## Member Sacraments

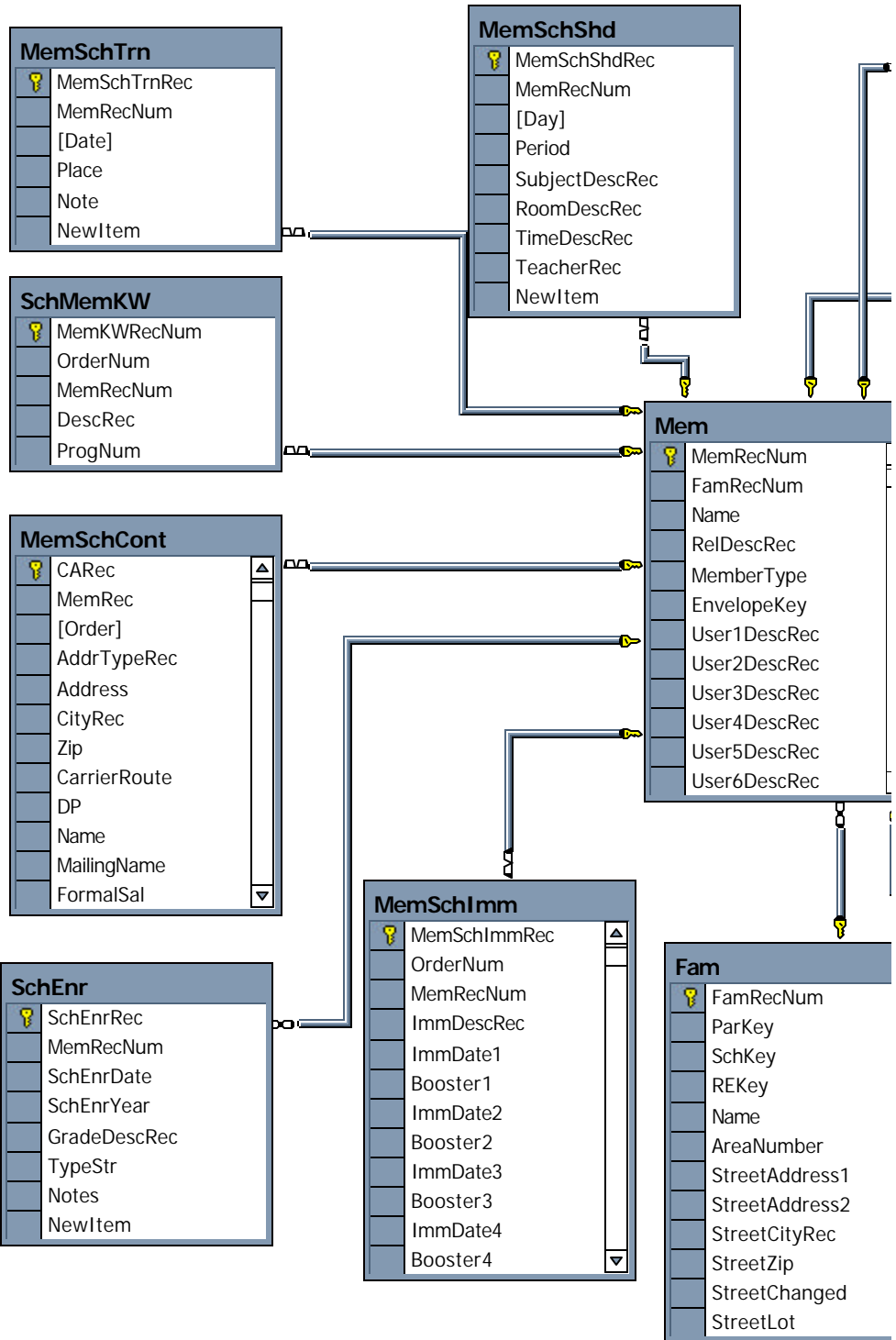


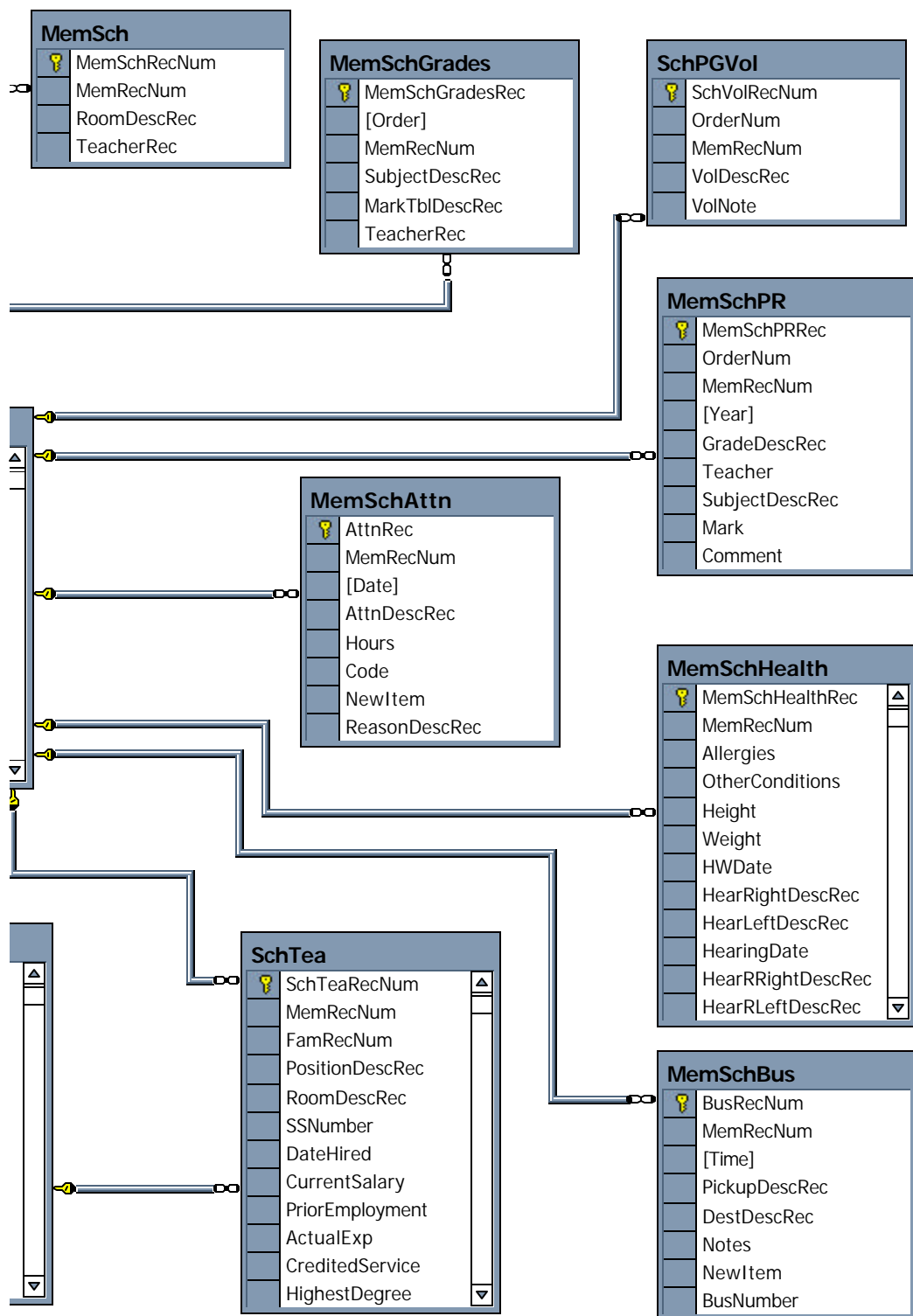
# Member Religious Ed Links



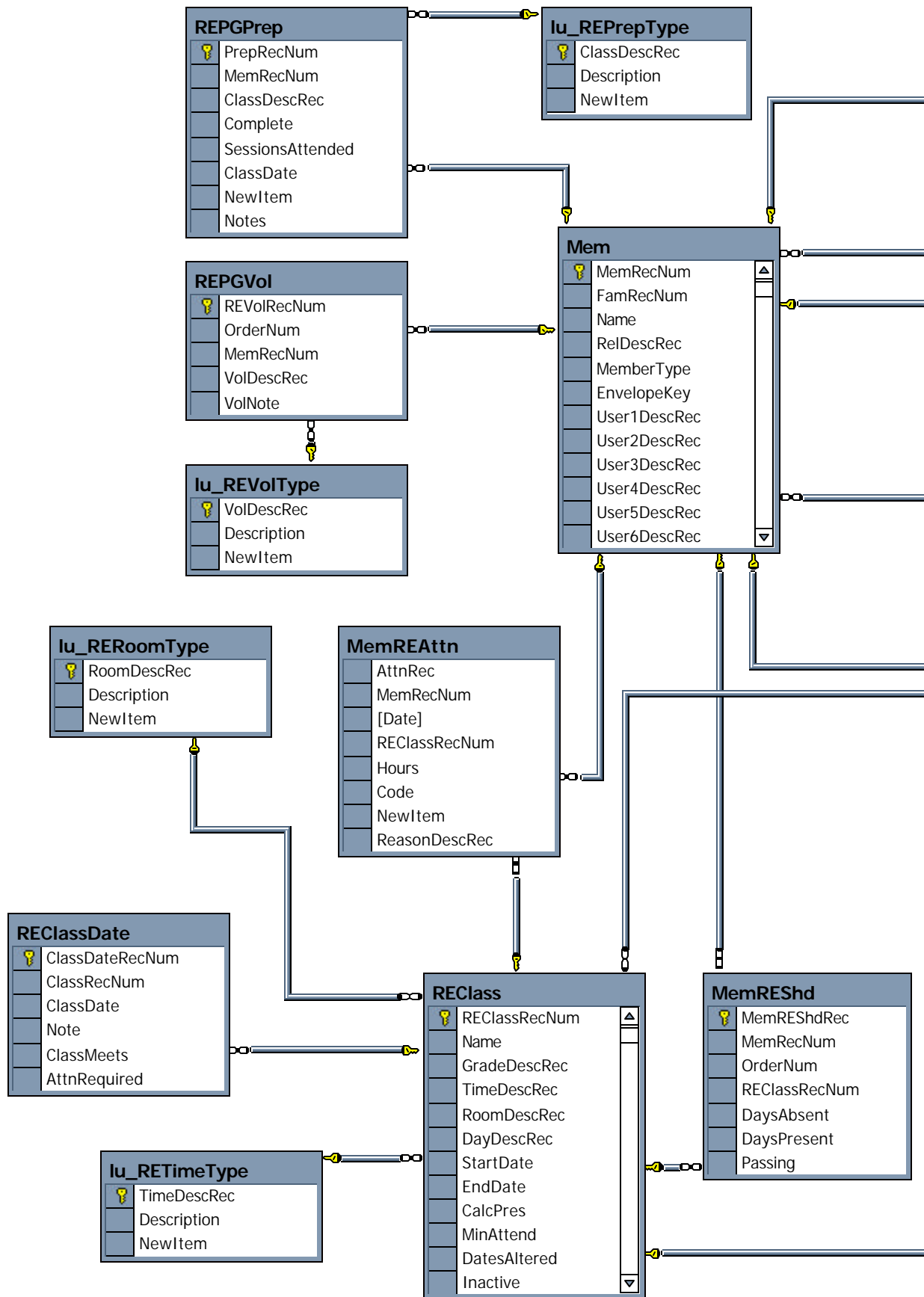


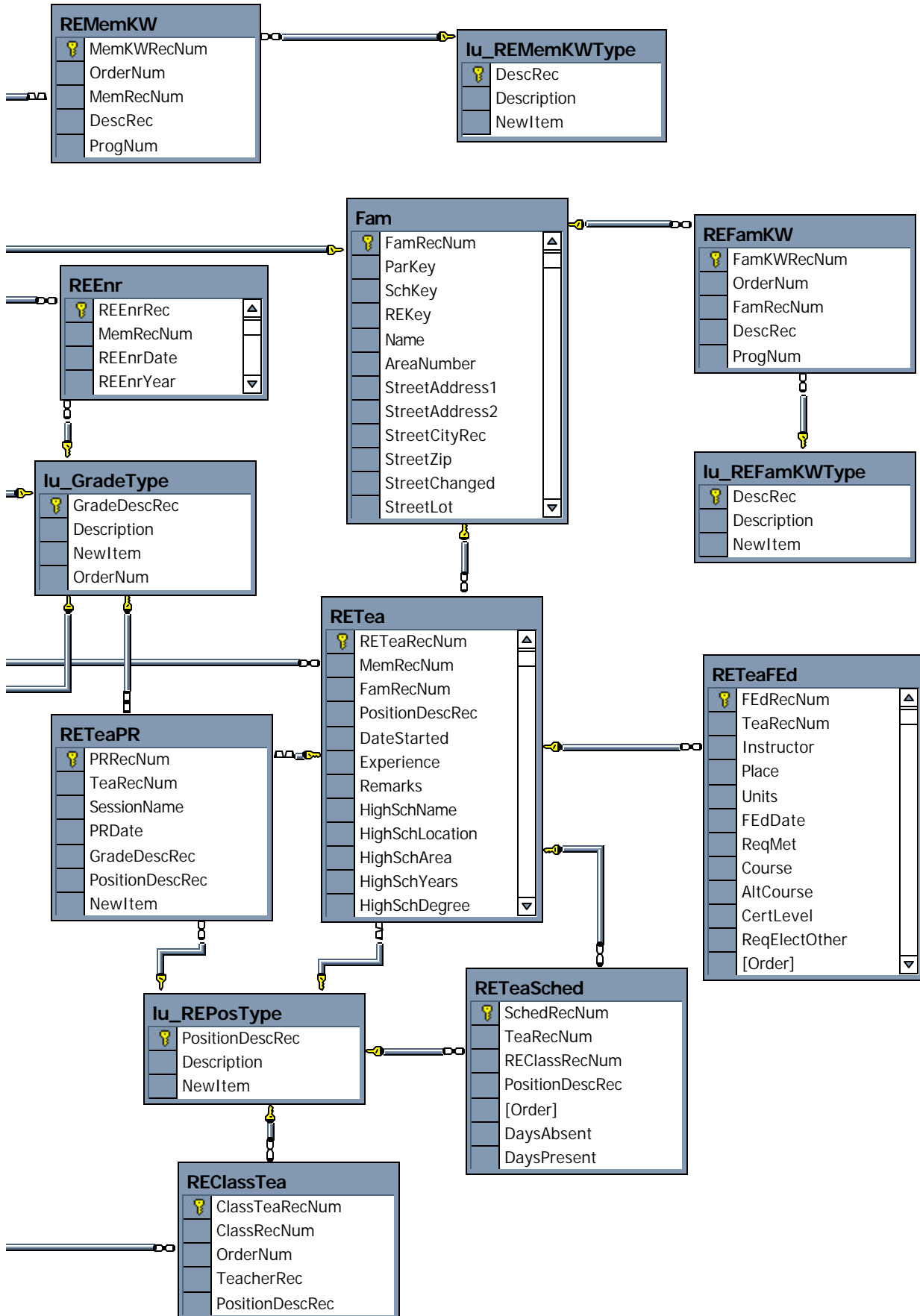
# Member School Links





# Religious Ed Links





# School Links

