



Symptoms

Navigating around SalesLogix can take a very long time; opening views/tabs can take minutes and there are unexpected pauses that did not occur in prior versions.

Reason

Currently there is an issue with SalesLogix not setting the grid property **:BindID** in time prior to the data in that control being activated. This has the unfortunate side-effect of bringing back **all** rows. However, a split second later – the BindID is set and then the grid is refreshed again. As you can imagine, in the case of Notes-History tab, this could be many rows (it will return all history for everyone, not just the account/contact you are looking at).

From a users' perspective – you cannot actually see this occur, as the grid refreshes too quickly. You do, however, feel the initial delay whilst the grid content is built.

It should be noted that, currently, this issue impacts all grids – so you may see issues with the Notes-History tab, the Attachment tab and any customised grid.

NB: The Notes-History grids are fixed in version 7.2 Service Pack 2, but the same issue can arise on the standard attachment tabs.

Resolution

Fortunately, the resolution is fairly simple to implement and can be summarised as follows: -

- Add a condition to the grid to ensure that the primary **EntityID** does contain data
 - For example, if you are changing the account form – use AccountID, the contact form uses ContactID and the opportunity form uses OpportunityID
- Add code to the **AXFormChange** to test for null and exit if it is

NB: You must make both changes – implementing only the condition or the code change will not fix the issue.

The following pages describe in detail how to make this change.



Add the condition & code

Example: If you have an issue with the contact Notes-History grid, using Architect open the **Contact:Notes-History** form. Double-click the grid and then double-click the SQL property. Add a new condition of ContactID "does contain data".

The screenshot shows the Architect software interface. The main window is titled 'Form - (Contact) Notes-History'. A 'Query Builder - New Query' dialog box is open, showing the 'Conditions' tab. The 'Conditions' table is as follows:

Not	Field	Operator	Value	Case Sens	And/Or
	(
	History.Type	not equal to	262164	True	AND
	History.Type	not equal to	262162	True	OR
	History.UserID	equal to	UserID	True) AND
	History.Contactid	does	contain data	True	END

Red arrows in the image point to the 'Contactid' field in the tree view, the 'does' operator, and the 'contain data' value.

Next, go to the Script tab and locate the sub **AXFormChange** and add the following code:

```
Sub AXFormChange(Sender)
    splMain.Size = splMain.Size - 1
    splMain.Size = splMain.Size + 1
    memNotes.Clear

    ' This Procedure sets filtering and forces grid to load
    If (grdHistory.BindId = "") Then
        Exit Sub
    End If
    CommonCheckBoxChange grdHistory
    lblRecordsShown.Caption = Application.Translator.Localize("Records shown: ") & grdHistory.Recordset.RecordCount

    memNotes.Text = HistoryChangeNode(grdHistory.GetCurrentField("HISTORYID")) 'DNL
    cboUser.Text = Application.Users.Item(Application.BasicFunctions.CurrentUserID).Name
    cboDays.Text = cboDays.Items(0)
End Sub
```

Save and release the form to all users. It is recommended you make the same change for account/contact and opportunity forms.



How to identify slow running issues

If you experience slow-running queries or general unresponsiveness you can use the **SLXProfiler.exe** tool to examine what SalesLogix is doing. You can find this tool in `\program files\saleslogix`. Simply run this immediately prior to when you experience the issue:

Time Stamp	SQL Type	Parse(ms)	Prepare(ms)	Secure(ms)	Execute(ms)	GetRows(ms)	Logins	Rows	User ID	Proce.	Machi.	Cursor	Client SQL
2008/0...	SYS_VS...	0.0000	0.0000	0.0000	0.6803	0.0523	0.0000	52	Mike	5144	1. 0...	FORWARD ONLY	SELECT A1.HISTORYID FROM HISTORY A1 INNER JOIN PICKLIST A2
2008/0...	USER	0.2177	0.0525	0.0013	0.9697	0.0000	0.0000	52	Mike	5144	1. 0...	VSSC	SELECT A1.HISTORYID, A2.TEXT A2_TEXT, A1.COMPLETEDDATE, A
2008/0...	SYS_VS...	0.0000	0.0000	0.0000	1.3422	0.0743	0.0000	52	Mike	5144	1. 0...	FORWARD ONLY	SELECT A1.HISTORYID, A2.TEXT A2_TEXT, A1.COMPLETEDDATE, A
2008/0...	USER	0.0806	0.0310	0.0027	0.6584	0.0213	0.0000	1	Mike	5144	1. 0...	FORWARD ONLY	SELECT LONGNOTES FROM HISTORY WHERE HISTORYID = 'H6UJ9

This tool is extremely useful, it will show you the query being sent from the application (“Client SQL”) and how the SalesLogix OLE DB Provider interprets and secures that prior to sending it to SQL Server (“Executed SQL”) – the columns Execute(ms) and Rows will show you how long it took the query to execute and how many rows were returned. Any very large numbers in either column will indicate a slow running query.

For example, this is a trace from one customer – you can see that the query took over 70 seconds to run and returned 99,652 rows !

Time Stamp	SQL Type	Parse(ms)	Prepare(ms)	Secure(ms)	Execute(ms)	Rows
226.0653	USER	0.2235	0.0710	0.0031	71545.1137	99652

A few milliseconds later, a secondary query came along and reduced that to just a few rows – and this is the classic behaviour that surrounds this problem.

