empath-e

MAKING CRM MAKE **SENSE**

SalesLogix Enhancements

Flattening SalesLogix Data

Date: 27/01/10
Version: 1.0
Author: Adam Travers
Contact: 0845 1368449 / 07540 163920

Acknowledgements: Ryan Farley Director of Development for Customer FX Corporation

CONTENTS

1.0 OVERVIEW

Provision

- Flattens out SalesLogix data to combine multiple rows into a single string using SQL.

Contents

The following SQL script templates are supplied with this update.

Type	Item
Database Object 1	Create Function SQL Script
Database Object 2	SQL View

Impact Analysis

The above additions will create two database objects within SQL Server, both objects can be easily removed without issue.

Installation

Two SQL server database objects will be installed when running both SQL queries:

- Scalar-valued Function
- SQL View

2.0 EXAMPLE 1: ACTIVITY DATA

In this example, we will be using the SalesLogix activity table which has a 1:M (One to many) relationship with the Contact table.

We will be combining all rows of activity data (the date and description only) into a single string for each contact. So, instead of having multiple rows of activities for each contact, we'll end up with a string of activities like this:

"3/2/2009 Lunch Meeting, 4/02/2009 Appointment, 8/10/2009 Follow-up Call".

SQL Create Function

Login to SQL Server as the **sysdba** user and run the following SQL statement, this function will take in a parameter of a ContactID and will return a string of Activities for each Contact:

```
Create Function sysdba.GetActivityListForContact
(
    @contactid varchar(12)
)
Returns varchar(8000)
as
Begin

    Declare @list varchar(8000)
    Set @list = ''

    Select
      @list = @list + convert(varchar, T1.startdate, 101) + ' ' +
      T1.description + ', '
    From
      sysdba.activity T1
    Where
      T1.contactid = @contactid
    Order by
      T1. startdate

    Set @list = rtrim(@list)
    If @list <> '' Set @list = left(@list, len(@list) - 1)

    Return @list

End
```

Now that we have the function, we can use it in a query, or wrap it in a SQL view as follows:

Create SQL View

Create a **New Query** against your database using the SYSDBA user, copy and paste the below statement in to the query window to create the vContactExtended SQL view.

```
Create view sysdba.vContactExtended
as

    Select
          *
        , sysdba.GetEventListForContact(contactid) as ActivityList

    From
        sysdba.contact
```

Expand Databases | Database | Views

Select and open the **vContactExtended** SQL view

You will now see the new ActivityList column added where you will see the activities listed all in a single field for each Contact 'Flattened'

	FIRSTNAME	LASTNAME	ACCOUNT	ActivityList
	John	Alfred	Ellington Tool Co...	10/26/2006 Follow-up, 02/16/2008 Research
	Peter	Bates	Alpert Fan Inc	10/26/2006 Follow-up, 10/26/2006 Demo
	Wyman	Bro	CTL Capital Group	10/26/2006 Follow-up
	Jeff	Carston	Indiana Tool and...	10/26/2006 Send quote
	Robert	Class	Farm Express Pr...	10/28/2006 New Meeting
	Jonathon	Hardy	Abbott WorldWide	10/26/2006 Referred from Abbott Ltd., 07/02/2007 send stuff, 07/03/2007 Ticket 001-00-000037
	Mike	Majewski	Chicago Auto Pr...	10/26/2006 Follow up
	Lindsay	Malvin	Advising Group	10/27/2006 RE: Status of work at Abbott Ltd?
	John	Milligan	Clayco Manufact...	10/26/2006 Demonstration
	Patrick	Norman	Davidson Drills	10/24/2006 Presentation

3.0 EXAMPLE 2: HISTORY DATA

In our second example, we will be using the SalesLogix History table which again has a 1:M (One to many) relationship with the Contact table.

We will be combining all rows of History data (the category and description only) into a single string for each contact.

"Cold Call Discuss opportunities, Schedule initial qualification meeting".

SQL Create Function

Login to SQL Server as the **sysdba** user and run the following SQL statement, this function will take in a parameter of a ContactID and will return a string of History records for each Contact:

```
Create Function sysdba.GetHistoryListForContact
(
    @contactid varchar(12)
)
Returns varchar(8000)
as
Begin

    Declare @list varchar(8000)
    Set @list = ''

    Select
      @list = @list + convert(varchar, T1.category, 101) + ' ' +
      T1.description + ', '
    From
      sysdba.History T1
    Where
      T1.contactid = @contactid
    Order by
      T1. createdate

    Set @list = rtrim(@list)
    If @list <> '' Set @list = left(@list, len(@list) - 1)

    Return @list

End
```

Create SQL View

Create a **New Query** against your database using the SYSDBA user, copy and paste the below statement in to the query window to create the vContactHistoryExtended SQL view.

```
Create view sysdba. vContactHistoryExtended
as

    Select
          *
        , sysdba.GetHistoryListForContact(contactid) as HistoryList

    From
        sysdba.contact
```

Expand Databases | Database | Views

Select and open the **vContactHistoryExtended** SQL view

You will now see the new HistoryList column added where you will see the History items listed all in a single field for each Contact.

ACCOUNT	FIRSTNAME	LASTNAME	HistoryList
Maxet Garden T...	Steve	Turnbull	Sales Visit Purchase Question, Introductory Meeting
Equity Residenti...	Sharyn	Tattam	Sales Trade Show Qualification, Sales Research the prospect, Sales Gain more i...
Koll Trucks	Joe	Garcia	Sales Trade Show Qualification, Sales Discovery phone call
West Travels	Matthew	Zulauf	Sales Trade Show Qualification, PROC Lead Qualification, Sales Follow-up Next ...
Bank of the Sun	John	Sherman	Sales Trade Show Qualification, PROC Lead Qualification, Sales Follow-up Next ...
Ciba Multimedia	Harlin	Brown	Sales Trade Show Qualification, PROC Lead Qualification, Sales Follow-up Next ...
High Risers	Bill	Worthington	Sales Trade Show Qualification, PROC Lead Qualification, Sales Follow-up Next ...
S. Pacific Corpor...	Charles	Armigo	Sales Trade Show Qualification, PROC Lead Qualification, Sales Follow-up Next ...
Equity Residenti...	Sharon	Merrell	Sales Trade Show Qualification, PROC Lead Qualification, PROC Contact Status...
Trailer Hitch	Kadri	Vikta	Sales Trade Show Qualification, PROC Lead Qualification, PROC Contact Status...

4.0 ENABLE VIEW WITHIN SALESLOGIX

If you are running SalesLogix v7 and above you will have the ability to expose SQL tables and views generated outside of SalesLogix within the SalesLogix Administrator, this will enable users to view the data through the SalesLogix query builder.

To enable a view in the database

Log in to the SalesLogix Administrator as admin | Manage | Database

In the Database Manager, Locate the view you want to enable.

Right-click on the view, and then click Properties.

In the view Properties dialog box, verify that the information is correct, and then click Enable.
When a view has been enabled, it cannot be disabled. The Enable button becomes unavailable.

If you now login to the SalesLogix client | Lookup | Contacts | Query Builder and create a join from Contact.Contactid to VContactExtended.Contactid this will expose the SQL view for querying.