Project Users

A customer needs to associate a number of Users to an Opportunity so that when something changes on that Opportunity all the associated Users are notified via email.

We could build new functionality into the system to allow a CRM User to link a User to an Opportunity, but maybe there is an easier solution that might satisfy the customer's needs without the need for lots of 'coding'...

This article will show you how to:

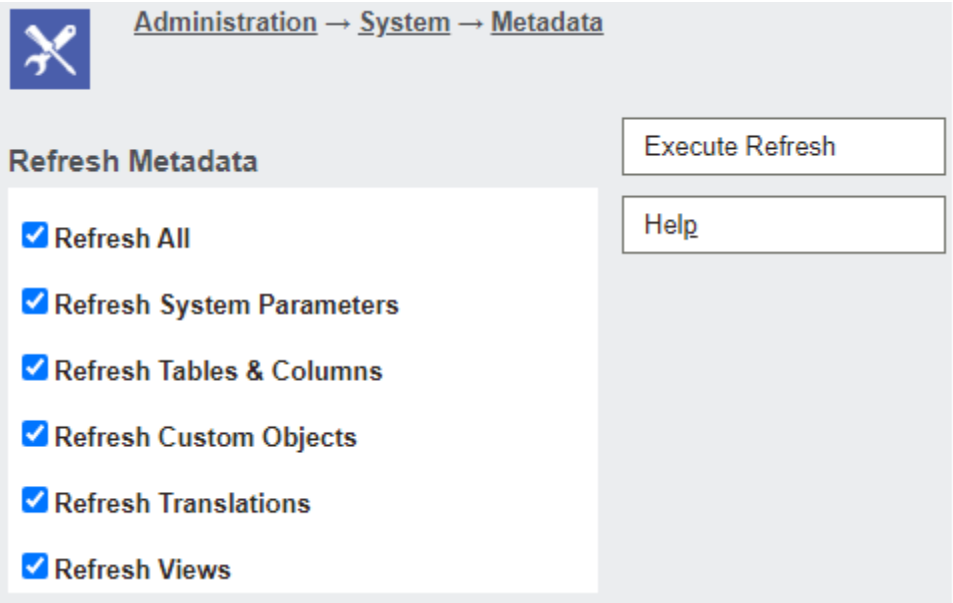
- Create a 'User Multi-select' field.
- Create a 'virtual' field in CRM - a field that exists in a view but not on a table.
- Through a SQL 'trick' create a list of email addresses.
- Send an email out to that list of email addresses.

Step 1 - Set up the User Multi-select meta data.

The first step is to create the User Multi-select field.

Create a new 'dummy' translation in the CRM system with the following settings:

Caption Code	1
Caption Family	Users
Caption Family Type	Choices
Caption Order	
Caption Context	
US Translation	Users
UK Translation	Users
French Translation	Utilisateurs
German Translation	Benutzer
Spanish Translation	Usuarios
Dutch Translation	Gebruikers



Step 2 - Create the new Opportunity field

Next, we need to create our new 'Project Users' field on the Opportunity entity.

Entry Type	Multi-select
Column Name	`oppo_c_ProjectUsers`
Caption	Project Users
Selection Height	5
Lookup Family	Users
Lookup Width px	
Default	

Add the new field to the Opportunity Summary screen.

Step 3 - Create a view for the Workflow

We need to create a view (vSummaryOpportunityCust), on the Opportunity entity, that we can use in the Workflow to get the email addresses. The example below is just a copy of the select statement from the vSummaryOpportunity view, modified to include a new field, `oppo_c_ProjectUsersEmail`, for the email addresses.

```
SELECT
  epd_pers.epd_EmailAddress AS Pers_EmailAddress,
  epd_pers.epd_PhoneCountryCode AS Pers_PhoneCountryCode,
  epd_pers.epd_PhoneAreaCode AS Pers_PhoneAreaCode,
  epd_pers.epd_PhoneNumber AS Pers_PhoneNumber,
  epd_pers.epd_FaxCountryCode AS Pers_FaxCountryCode,
  epd_pers.epd_FaxAreaCode AS Pers_FaxAreaCode,
  epd_pers.epd_FaxNumber AS Pers_FaxNumber,
  epd_comp.epd_EmailAddress AS Comp_EmailAddress,
  epd_comp.epd_PhoneCountryCode AS Comp_PhoneCountryCode,
  epd_comp.epd_PhoneAreaCode AS Comp_PhoneAreaCode,
```

```
  epd_comp.epd_PhoneNumber AS Comp_PhoneNumber,
  epd_comp.epd_FaxCountryCode AS Comp_FaxCountryCode,
  epd_comp.epd_FaxAreaCode AS Comp_FaxAreaCode,
  epd_comp.epd_FaxNumber AS Comp_FaxNumber,
  (
    SELECT
      CAST(
        CAST(Parm_Value AS NCHAR) AS INTEGER
      ) AS Expr1
    FROM
      dbo.Custom_SysParams
    WHERE
      (Parm_Name = 'BaseCurrency')
  ) AS Oppo_WeightedForecast_CID,
  (
    (
      Oppo_Forecast / Oppo_Forecast_Currency.Curr_Rate
    ) * Oppo_Certainty / 100
  ) AS Oppo_WeightedForecast,
  Person.*,
  Company.*,
  Opportunity.*,
  (select ltrim(rtrim(User_EmailAddress)) + ';'
   from Users
   where charindex(',' + rtrim(User_UserId) + ',', oppo_ProjectUsers) > 0
     and rtrim(isnull(User_EmailAddress,'')) <> ''
   for xml path('')) AS oppo_c_ProjectUsersEmail,
  Account.* ,
  Fin_Year, Fin_Quarter, Fin_Month
FROM Opportunity
cross apply [dbo].[FinancialYQM](Oppo_Opened, 0, 31)
LEFT JOIN Person ON Oppo_PrimaryPersonId = Pers_PersonId
  AND Pers_Deleted IS NULL
LEFT JOIN CRMEmailPhoneData epd_pers ON epd_pers.epd_EntityID = 13
  AND epd_pers.epd_RecordID = Pers_PersonID
LEFT JOIN Company ON Oppo_PrimaryCompanyId = Comp_CompanyId
  AND Comp_Deleted IS NULL
LEFT JOIN CRMEmailPhoneData epd_comp ON epd_comp.epd_EntityID = 5
  AND epd_comp.epd_RecordID = Comp_CompanyID
LEFT JOIN Account ON Oppo_PrimaryAccountId = Acc_AccountId
LEFT OUTER JOIN dbo.Currency AS Oppo_Forecast_Currency ON
dbo.Opportunity.Oppo_Forecast_CID = Oppo_Forecast_Currency.Curr_CurrencyID
WHERE
  Oppo_Deleted IS NULL
```

The clever bit is:

```
  (select ltrim(rtrim(User_EmailAddress)) + ';'
   from Users
   where charindex(',' + rtrim(User_UserId) + ',', oppo_ProjectUsers) > 0
     and rtrim(isnull(User_EmailAddress,'')) <> ''
   for xml path('')) AS oppo_c_ProjectUsersEmail
```

We're making use of SQL's ability to form XML to produce the list of emails addresses.

Step 4 - Create the 'view' field

Before we can create the workflow rule, we need to create a CRM field for the `oppo_c_ProjectUsersEmail` field. We have to do this manually in the database.

To help create the script quickly I use the following SQL to generate the fields for the table I'm interested in - in this case the Custom_Edits table.

```
select
  case row_number() over(order by c.column_id)
    when 1 then '  '
    else ', '
  end + c.name as [Column]
from sys.tables t
inner join sys.columns c on c.object_id = t.object_id
where t.name = 'Custom_Edits'
order by c.column_id
```

N.B. you can always look at existing fields to help work out what fields and values you'll need to enter. In the SQL script below I'm creating a multi-line text field.

```
insert into Custom_Edits(
  ColP_CreatedBy
, ColP_CreatedDate
, ColP_UpdatedBy
, ColP_UpdatedDate
, ColP_TimeStamp
, ColP_Entity
, ColP_ColName
, ColP_EntryType
, ColP_EntrySize
, ColP_CustomTableIDFK)
values(
  1
, getdate()
, 1
, getdate()
, getdate()
, 'Opportunity'
, 'oppo_c_ProjectUsersEmail'
, 11
, 40
, 10)
```

As before manually refresh the meta data.

Step 5 - Create the workflow rule

For testing purposes we're just going to create a global rule to test sending out emails.

With the Opportunity Workflow create a 'Global Rule':

Rule Name	Send Email Update
Table	Opportunity
View	vSummaryOpportunityCust
Rule Name	Send Email Test

Rule Enabled	?
Type	Global Rule

Add a new 'Workflow Action' of type 'Send E-mail'

From:	Set this to a CRM User. N.B. the user must have a valid email address.
To:	`#oppo_c_projectusersemail#`
Subject:	Test

Save the changes to the Workflow and Activate it, then test.

For simple things like this we can use a free email server called hMailServer. It is relatively easy to set up and it allows us to configure multiple domains which can be handy for Testing and Demonstrations.