

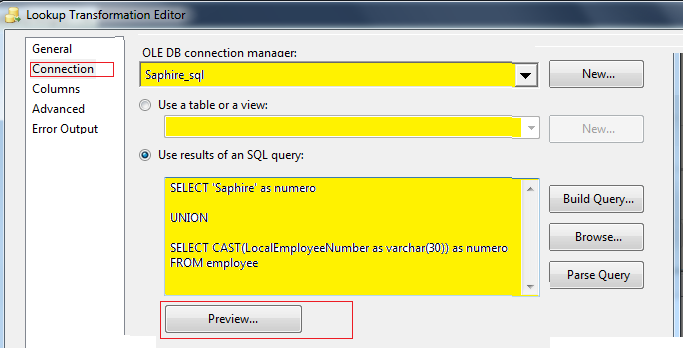
Because the fields are from different type, we can use Modified or:

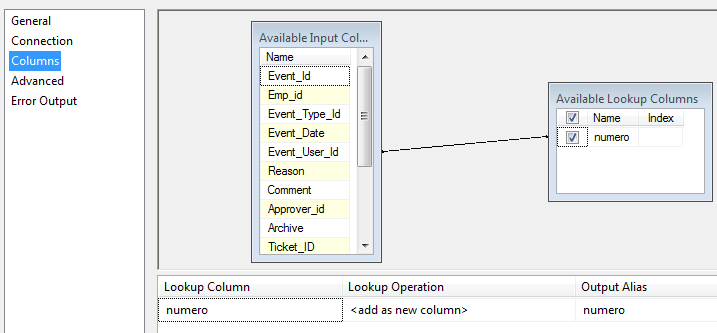
**SELECT CAST(LocalEmployeeNumber as Varchar(30)) as numero**

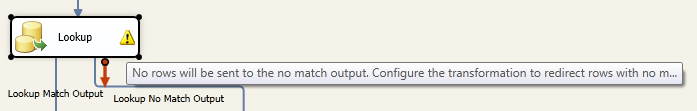
**FROM EMPLOYEE WITH(NOLOCK)**

**WHERE isnumeric(LocalEmployeeNumber) > 0**

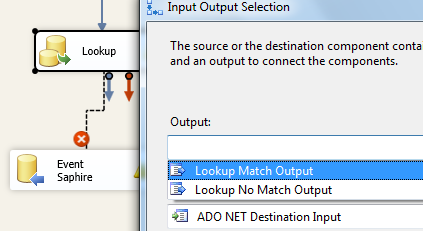
**AND LocalEmployeeNumber <> '330422740003'**



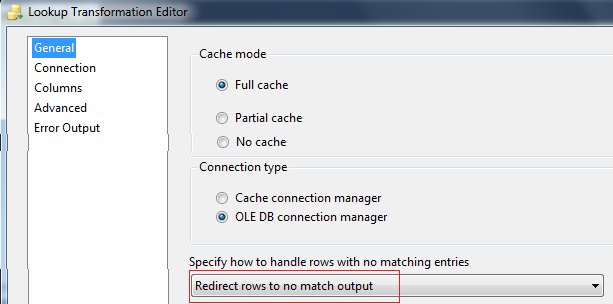


-

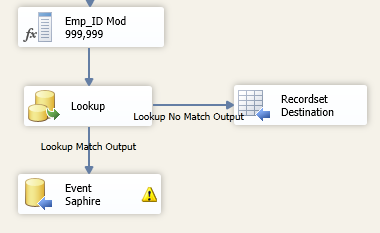
Assign Lookup



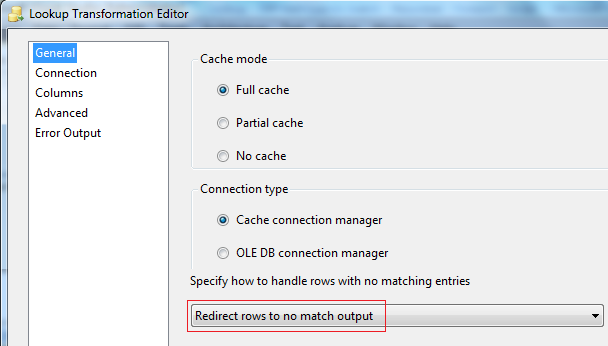
In the lookup, in



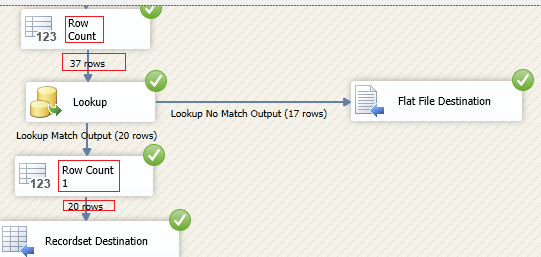
**Create a variable name MyData type Object to assign to the Recordset**



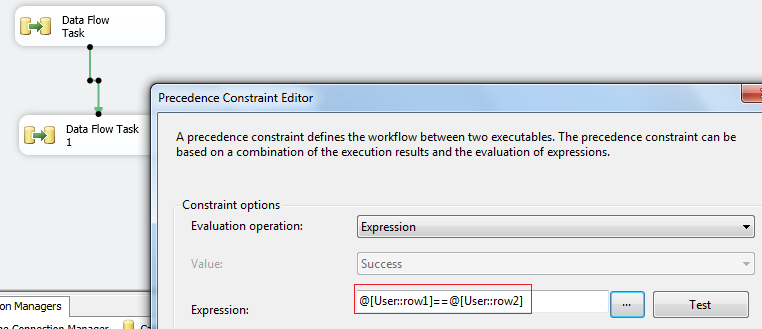
To make the Lookup work for both directions (matched and unmatched) you must set



You can also set the number of rows first and after

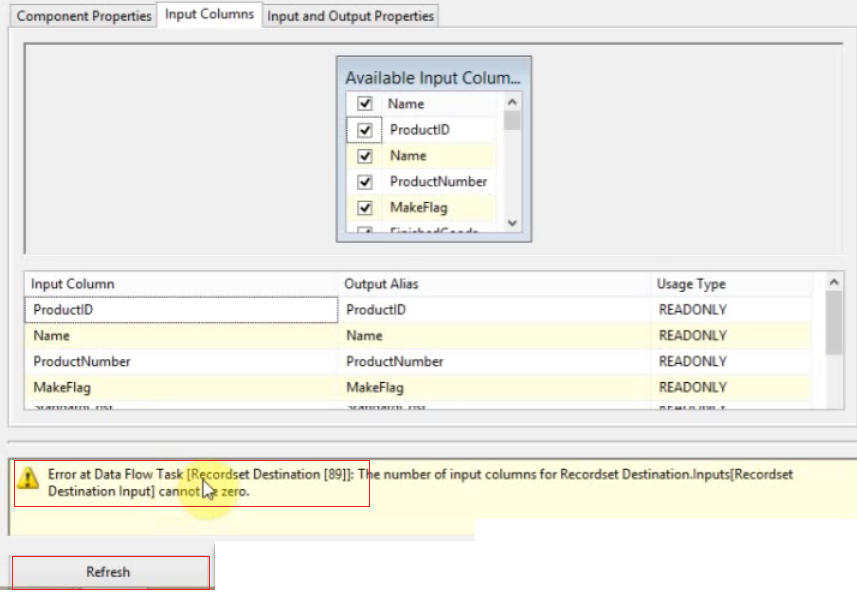


Then you can **set a condition to continue**



In case you will use a recordset, configure the Recordset by

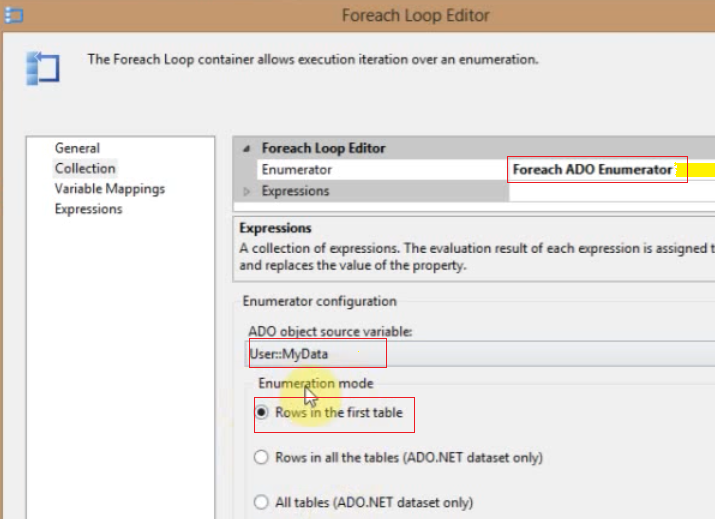
* Setting an Object Variable (MyData)
* Set the Input columns



To avoid the error **click Refresh**

To read the variable, you can use a script or a for each

In the foreach use

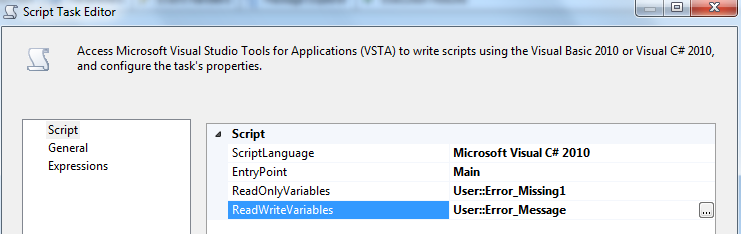


In the Variable Mapping

* Set local variables to each field in MyData
* Set the index to know the position inside MyData

Or get all the information in the script

Let use for Recordset store in variable **Error\_Missing1 instead of MyData.**



The script

Dts.TaskResult = (int)ScriptResults.Success;

OleDbDataAdapter oleDA = new OleDbDataAdapter();

DataTable dt = new DataTable();

DataColumn col = null;

DataRow row = null;

string strMsg = null;

oleDA.Fill(dt, Dts.Variables["Error\_Missing1"].Value);

foreach (DataRow row\_ in dt.Rows)

{

row = row\_;

foreach (DataColumn col\_ in dt.Columns)

{

col = col\_;

strMsg = strMsg + col.ColumnName + ": " + row[col.Ordinal].ToString() + Environment.NewLine;

}

}

MessageBox.Show(strMsg);

if (strMsg !="") Dts.Variables["Error\_Message"].Value = "Errores "+ strMsg;

Dts.TaskResult = (int)ScriptResults.Success;

= = = = = =

OleDbDataAdapter oleDA = new OleDbDataAdapter();

DataTable dt = new DataTable();

string userid2 = "";

HashSet<string> hsCodes = new HashSet<string>();

string strMsg = "The following values do not have reference with Employee table : ";

oleDA.Fill(dt, Dts.Variables["Error\_Missing1"].Value);

foreach (DataRow row1 in dt.Rows)

{

userid2 = row1[0].ToString();

hsCodes.Add(userid2);

}

foreach (string strVar2 in hsCodes)

{

strMsg = strMsg + strVar2 + ", ";

}

hsCodes.Clear();

//MessageBox.Show(strMsg);

if (strMsg != "") Dts.Variables["Error\_Message"].Value = strMsg;

Dts.TaskResult = (int)ScriptResults.Success;

DECLARE @Empl1 varchar(15)

DECLARE @Empl2 varchar(1000)

DECLARE @Empl3 varchar(1000)

DECLARE @Saphire varchar(10)

Select @Saphire = ?

DECLARE cur2 CURSOR FAST\_FORWARD FOR

SELECT Event\_user\_id

FROM Atrack\_Event\_Ticket\_DataStaging WITH(NOLOCK)

WHERE isnumeric(Event\_user\_id) > 0

AND Event\_user\_id not in

( SELECT LocalEmployeeNumber

FROM Employee WITH(NOLOCK)

WHERE isnumeric(LocalEmployeeNumber) > 0

AND LocalEmployeeNumber <> '330422740003' )

GROUP BY Event\_User\_Id

SELECT @Empl2 = ''

OPEN cur2

FETCH NEXT FROM cur2 INTO @Empl1

WHILE @@FETCH\_STATUS = 0

BEGIN

SELECT @Empl2 = @Empl2 + ', ' + @Empl1

FETCH NEXT FROM cur2 INTO @Empl1

END

CLOSE cur2

DEALLOCATE cur2

IF @Empl2 <> ''

BEGIN

SELECT @Empl2 = 'The following values do not have reference with Employee table : ' + @Empl2

END

SELECT @Empl2 AS LISTADO