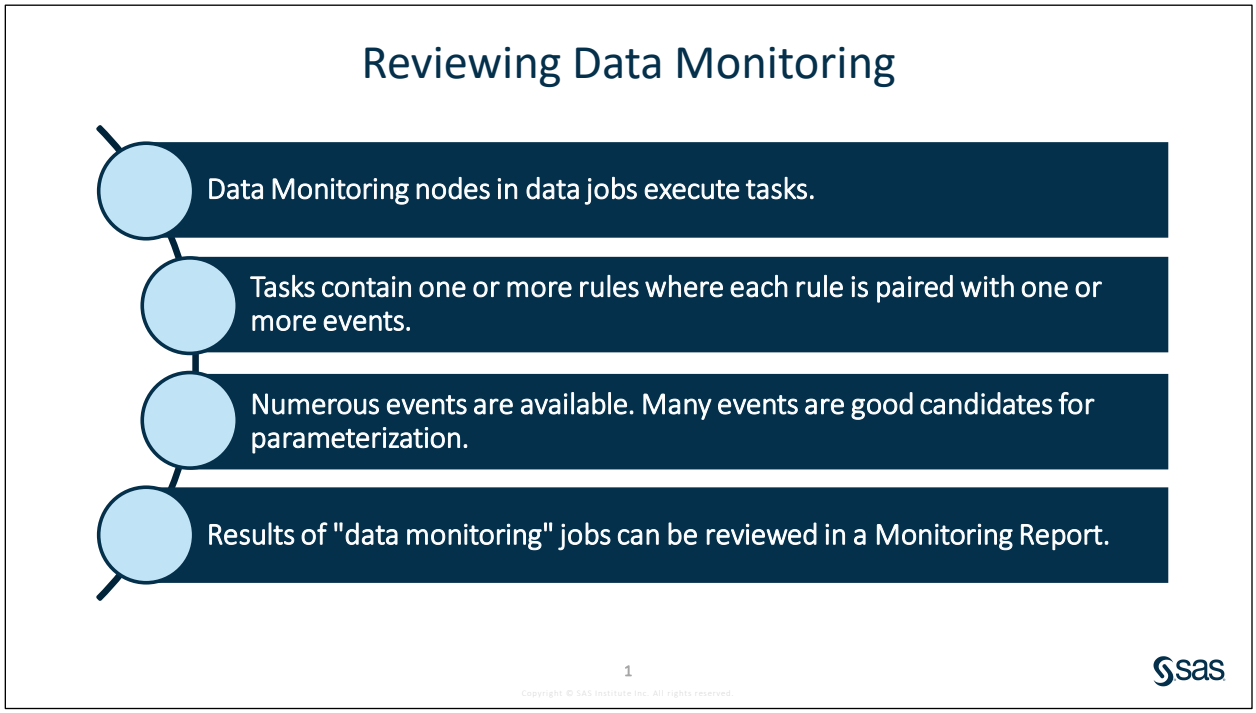
Using Macros in Data Monitoring

Using Macros in Data Monitoring

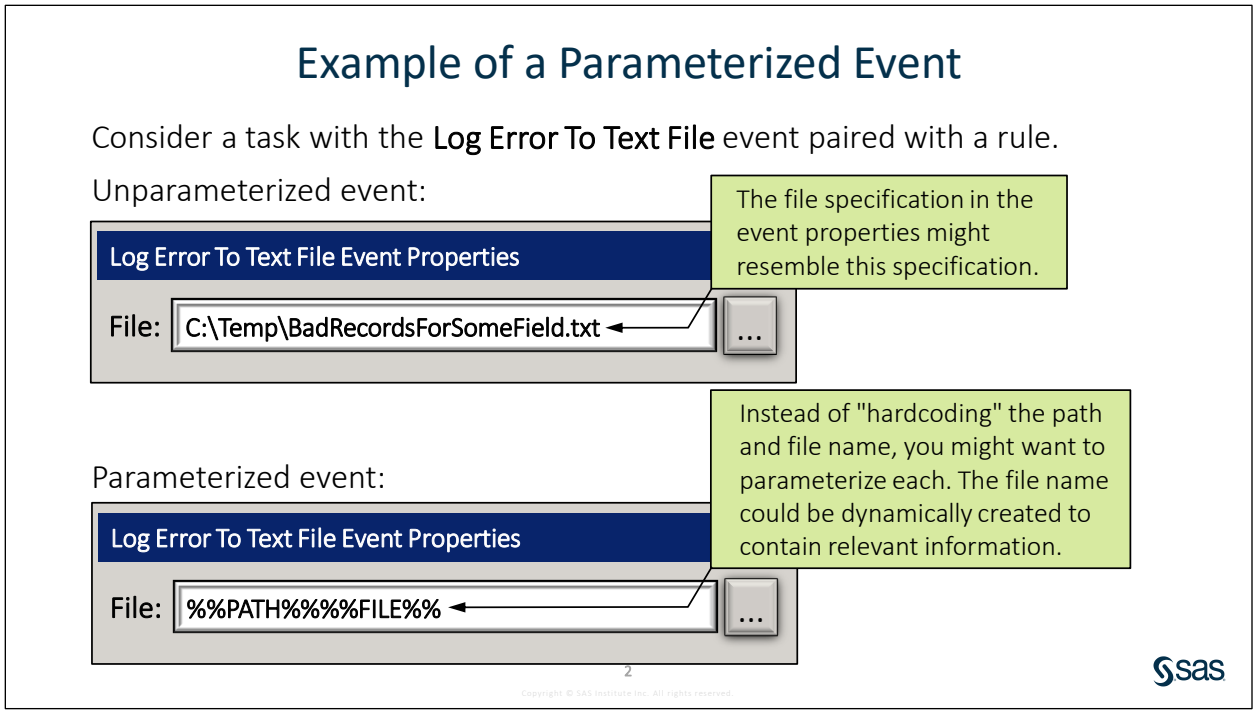


Recall that Data Monitoring nodes in a data job can execute a specified task.

Also recall that tasks are rules combined with events. If the rules have triggers, then the events occur.

There are numerous events available for a task, and many of these events are excellent candidates for parameterization.

A simple one to work with is the **Log Error To Text File** event. We will examine some possible parameterizations using this event.

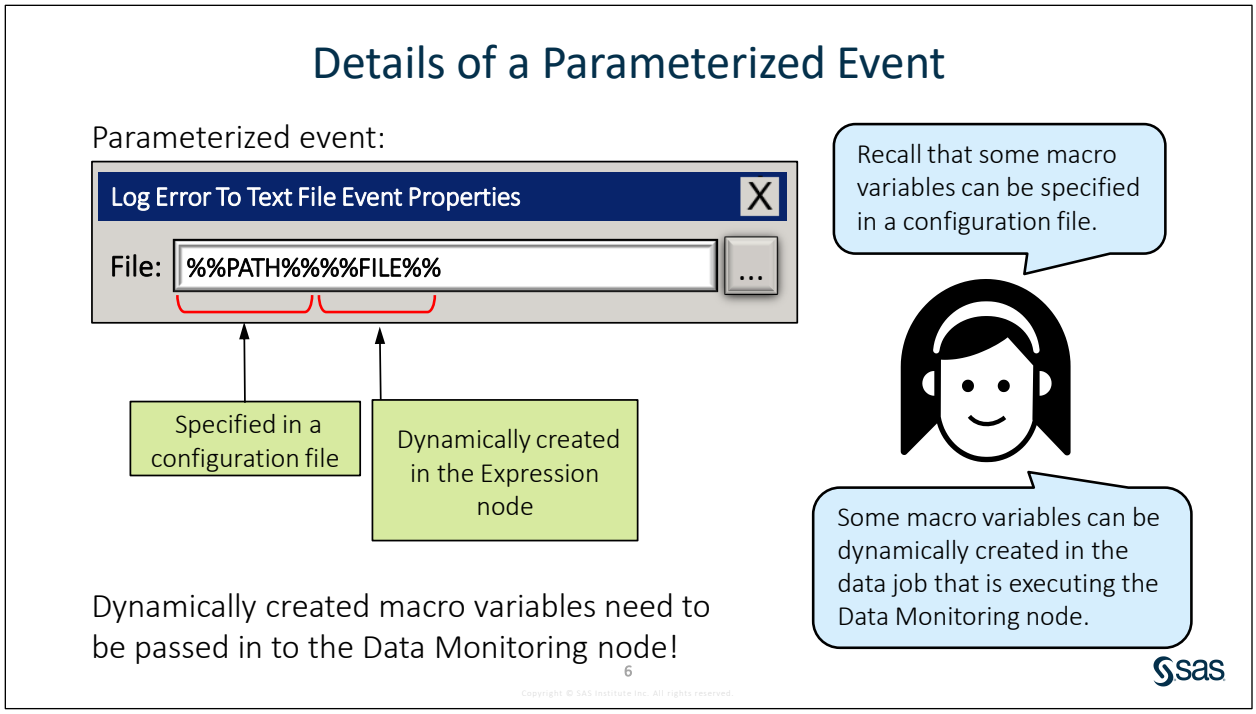


The Properties window for the **Log Error To Text File** event has a **File** field that allows for the specification of the file.

We have seen that the Expression node can be used to create a "dynamic" file name (that is, a file name that uses macro variables in its construction).

Note: Recall that the constructed file name from the previous section resembled the following: **Ch12D3_Macro_Example_dfConglomerateGifts_Customers.txt**

The **INPUT_DSN** and **INPUT_TABLE** macro variables were used to help construct this file name.

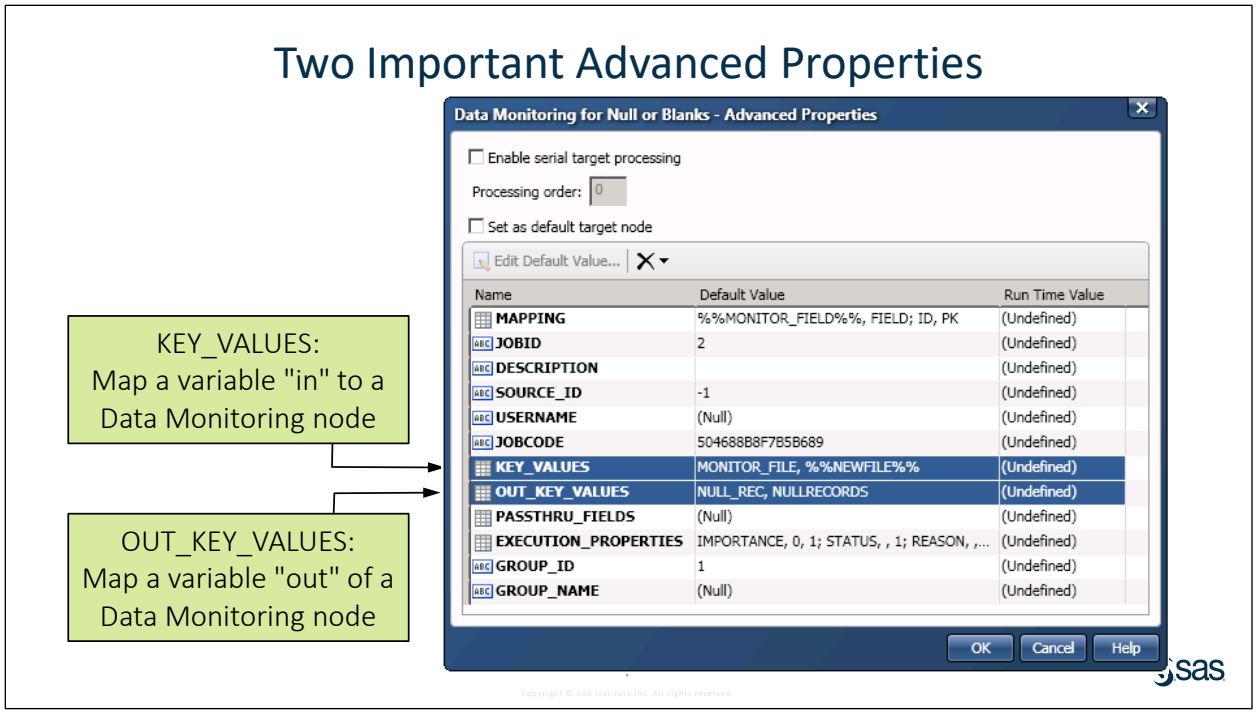


Returning to the event **Log Error To Text File**, we could parameterize the file information to be **%%PATH%%%%FILE%%**.

The **PATH** macro variable could be a macro variable defined in a configuration file.

The **FILE** macro variable could be one dynamically created.

If the Data Monitoring node is to use a dynamically created macro variable (one created in the data job, not one read in from a configuration file), the dynamic macro variable must be passed in to the node using the KEY_VALUES advanced property.



Each Data Monitoring node allows for the specification of basics properties in a Properties window.

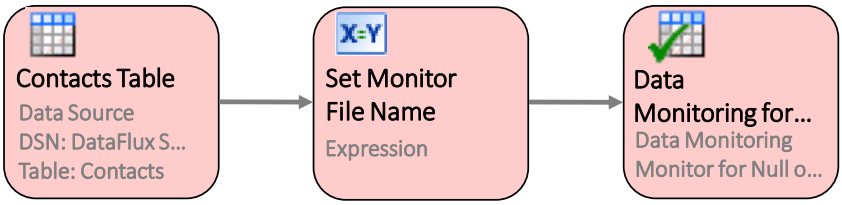
The Advanced Properties window for a Data Monitoring node shows additional items that cannot be set in the corresponding Properties window.

Two of these advanced properties are KEY_VALUES and OUT_KEY_VALUES.

KEY_VALUES	Allows for the mapping of one or more macro variables (perhaps created in an Expression node) to one or more variables referenced in one or more events for the selected task.
OUT_KEY_VALUES	Allows for the mapping of one or more variables (perhaps created with an event) to one or more variables that can then be subsequently referenced in the data job.

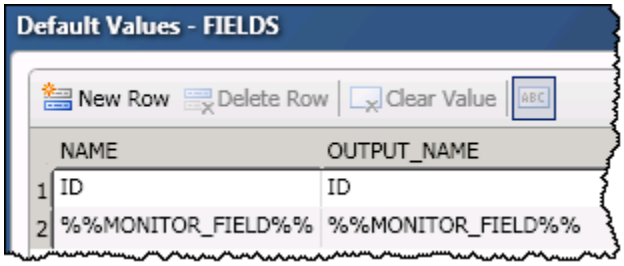
Example Data Monitoring Job

Consider a simple three-node data job:



8

- The Data Source node is reading two fields from the Contacts table: an ID field and a field that needs to be checked or monitored for null or blank values.
- The field to be monitored is parameterized with a macro variable (for example, **MONITOR_FIELD = STATE**).
- The parameterization of the field to be monitored is accomplished by accessing the Advanced Properties of the Data Source node. The FIELDS attribute is edited. (Note that it is a table attribute.) The **New Row** tool can be selected, and you can enter **%%MONITOR_FIELD%%** for both the **NAME** and **OUTPUT_NAME** locations.

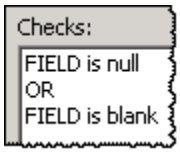


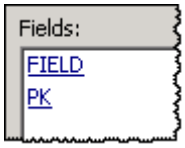
- The Expression node uses the macro variable with the field name to be monitored to create a new macro variable for a file name, and the file name will have the field name to be monitored embedded in it.

```
hidden string filename
filename='Ch12D4_Null_or_Blank_Records_for_' & getvar("MONITOR_FIELD") & '_Field.txt'
setvar("NEWFILE", filename)
```

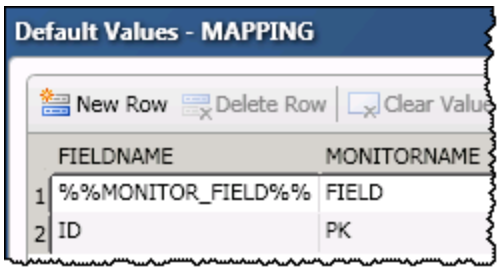
Thus, the **NEWFILE** dynamic macro variable is what needs to be passed in to the Data Monitoring node.

- The Data Monitoring node is executing a task that contains one rule and three events.
- The rule is defined for a field named **FIELD** and does a simple check of **FIELD is null OR FIELD is blank**. Two fields (**PK**, **FIELD**) are identified for aiding in discovering records that violate the rule.





- When the task is applied via the Data Monitoring node in the example data job, the **ID** field from the **Contacts** table is paired with the **PK** field, and the field to be monitored (which we are referencing as **%%MONITOR_FIELD%%**) is paired with the **FIELD** field. These pairings can be specified in the Advanced Properties of the Data Monitoring node with the MAPPING attribute.

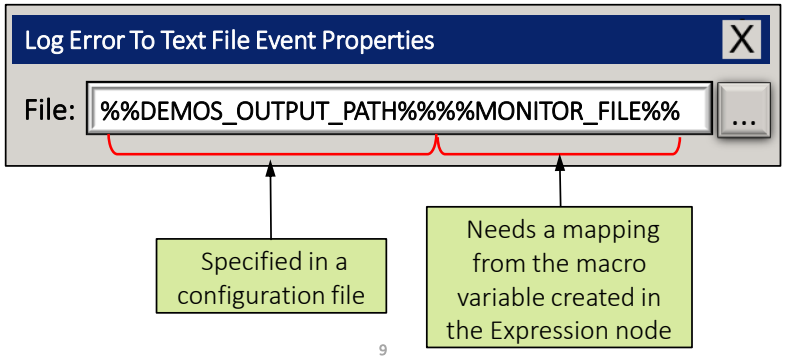


- Here are the three events defined in the task:
  - Log error to repository
  - Log error to text file
  - Set a data flow key/value
- The **Log error to repository** event writes the PK values (recall that PK = ID values) of the offending records to the repository (viewable on the Trigger Values tab in the Monitoring Report).

Parameterizing the Log Error To Text File Event

The Log Error To Text File event has macro variables specified for the path and filename.

- The path macro variable is set in a configuration file.
- The file name macro variable is dynamically set in the Expression node.



9

Using the KEY_VALUES Attribute

Reviewing:
- The Expression node created a dynamic macro variable named **NEWFILE**.
- The **Log Error To Text File** event was defined with a reference for **MONITOR_FILE**.
- **NEWFILE** needs to pass its value to **MONITOR_FILE**.

The **KEY_VALUES** attribute for the Data Monitoring node (found in the Advanced Properties) will resolve a reference to **NEWFILE** and assign this to **MONITOR_FILE**.

Default Values – KEY_VALUES

KEY	VALUE
MONITOR_FILE	%%NEWFILE%%

11

The Advanced Properties for the Data Monitoring node has a KEY_VALUES attribute that can be used to pass values in to key information needed for one or more of the events defined for the selected task.

For the current example, the MONITOR_FILE is a key piece of information needed for the **Log Error To Text File** event. The value for MONITOR_FILE will be the resolved value of the **NEWFILE** macro variable reference.

Investigating the Set A Data Flow Key/Value Event

Set Data Flow Key/Value Event Properties

Name	Value
NULL_REC	TRUE

1 This event creates a "key" named **NULL_REC** with a value of TRUE.

Default Values – OUT_KEY_VALUES

KEY	OUT_KEY
NULL_REC	NULLRECORDS

2 The advanced property **OUT_KEY_VALUES** is assigning the value of **NULL_REC** to the data job output variable **NULLRECORDS**.

3 The data job output variable can be examined to see the value of TRUE.

Data Flow | Settings | Variables | Log

Name	External Use	Value
NULLRECORDS	Output	TRUE

16

If a trigger occurs for a rule paired with the **Set a Data Flow Key/Value** event, the defined variable/value can be mapped to an existing data job output variable.

Some things to keep in mind:

- The **Set a data flow key/value** event allows for the creation of variable and value pairs that can be passed out of a Data Monitoring node.
- The **OUT_KEY_VALUES** attribute is an advanced property for the Data Monitoring node and maps the variable/value pairs created with the **Set a data flow key/value** event to the data job output variables.
- Data jobs have a Variables tab. This tab allows for the creation of input and output variables. Typically, these variables are used within a process job.

In the example event Properties window, a "key" is created with the name of NULL_REC and a value of TRUE.

Recall that the rule in this example is searching for null or blank values for a selected field.

If a trigger occurs (that is, if a null or blank value is found for one or more records for the specified field), then NULL_REC will be created with the value of TRUE.

If this key and corresponding value is needed elsewhere, then it can be passed out of the Data Monitoring node to an output variable for the data job. The OUT_KEY_VALUES advanced property can map the triggered key to a data job's output variable.

Note: The output variable must be defined on the data job's Variables tab.

When the data job executes, if the monitored rule has triggers, the mapping of the NULL_REC value to NULLRECORDS can be seen on the data job's Variables tab.

Macro Variables in Data Monitoring

To summarize, the data job uses three macro variables:

MONITOR_FIELD	• Specified in Data Source node as a selected field • Used in Expression node to create output file name macro variable • Used in Data Monitoring node as the field to monitor
DEMOS_OUTPUT_PATH	• Used in an event in selected task for Data Monitoring node
MONITOR_FILE	• Created in Expression node • Used in event from selected task in Data Monitoring node

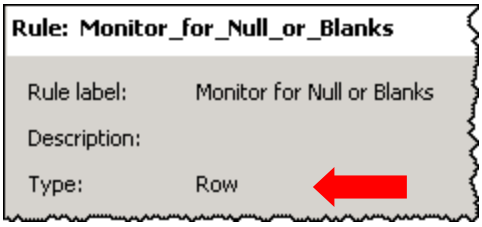
17

sas

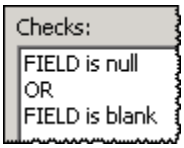
Using Macro Variables in Data Monitoring

This demonstration illustrates the use of macro variables in a data job that uses the Data Monitoring node. The needed macro variables are first created and examined. The task to be used is examined. An existing data job is accessed, and the use of the macro variables is explained.

1. If necessary, invoke Data Management Studio.
   a. Select **Start** ⇨ **All Programs** ⇨ **DataFlux** ⇨ **Data Management Studio 2.7**.
   b. Click **Cancel** in the Log On window.
2. Verify two macro variables (**DEMOS_OUTPUT_PATH** and **MONITOR_FIELD**).
   a. Click the **Administration** riser bar.
   b. Expand the **Macro Files** item.
   c. Click **Advanced**.
   d. In the navigation pane, under Macro Files, click **Advanced**.
   e. Verify that the macro variable **DEMOS_OUTPUT_PATH** is set to the value *D:\Workshop\dqdmp2\FTDemos\files\output_files*.
   f. Verify that the macro variable **MONITOR_FIELD** is set to the value *STATE*.
3. Access the Business Rules Manager to view the rule and task named **Monitor_for_Null_or_Blanks**.
   a. Select **Tools** ⇨ **Business Rules Manager** ⇨ **Advanced Demos**.
   b. Expand the **Rules** folder.
   c. Click the rule named **Monitor_for_Null_or_Blanks**.
   d. Verify that this rule is a row-based rule.



   e. Verify that the check is the following:



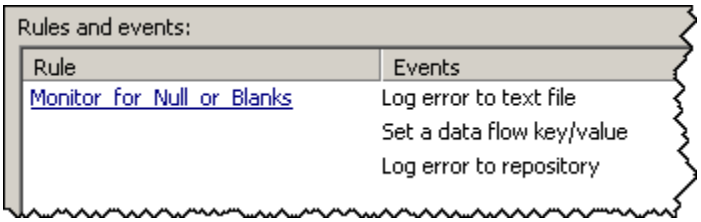
   f. Verify that the fields involved in the rule are **FIELD** and **PK**.

g. Expand the **Tasks** folder.

h. Click the task named **Monitor_for_Null_or_Blanks**.

i. Verify that the fields involved in the task are **FIELD** and **PK**.

j. Verify that the rule and events involved in the task are the following:



In the above task, three events are defined.

- The event **Log error to text file** has two references to variables (**DEMOS_OUTPUT_PATH** and **MONITOR_FILE**) specified in its properties.
  - The **DEMOS_OUTPUT_PATH** variable can (and will be) read from the Advanced configuration file.
  - The **MONITOR_FILE** variable is going to be constructed in an Expression node that precedes the Data Monitoring node. (The Data Monitoring node will execute this task.) Because **MONITOR_FILE** is to be a dynamically calculated, we need to pass the value in to the Data Monitoring node. This is accomplished with the KEY_VALUES advanced property of the Data Monitoring node.
- The event **Log error to repository** will write the primary key values of the bad data to the Trigger Values tab in the Monitoring Report for this repository. There is no parameterization in this event.
- The event **Set a data flow key/value** is an event that enables you to define a variable and a value. This variable (and its value) can be passed out of the Data Monitoring node (possibly for use as an output variable from the data job) using the OUT_KEY_VALUES advanced property of the Data Monitoring node.

k. Review the settings for the three events.

1) Click **Edit Task**.

2) Click **Rule Details**.

a) Click the **Log error to text file** event and then click **Edit**.

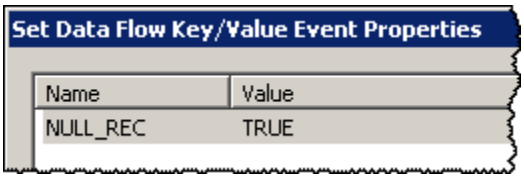
(1) Verify that the file value is **%%DEMOS_OUTPUT_PATH%%%%MONITOR_FILE%%**.

Note: **DEMOS_OUTPUT_PATH** will be read from a configuration file. **MONITOR_FILE** will be dynamically calculated.

(2) Click **Cancel** to close the Log Error To Text File Event Properties window.

b) Click the **Set a data flow key/value** event and then click **Edit**.

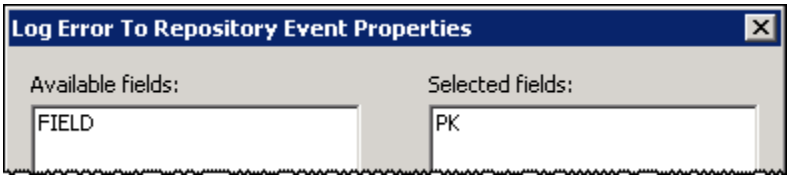
(1) Verify that one variable NULL_REC is defined and assigned a value of TRUE.



(2) Click **Cancel** to close the Set Data Flow Key/Value Event Properties window.

c) Click the **Log error to repository** event and then click **Edit**.

(1) Verify that the **PK** field is selected.



(2) Click **Cancel** to close the Log Error To Repository Event Properties window.

d) Click **Cancel** to close the Rule Details window.

3) Click **Cancel** to close the Task Properties window.

l. Select **File** ⇨ **Close** to close the Business Rules Manager tab.

4. Access an existing data job.

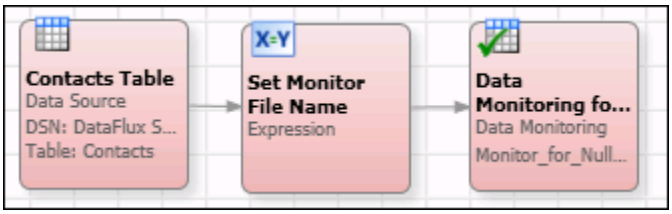
a. Click the **Folders** riser bar.

b. Expand the **Advanced Demos** repository.

c. Click the **batch_jobs** folder.

d. Double-click the **Ch12D4_Macros_in_Monitoring** data job.

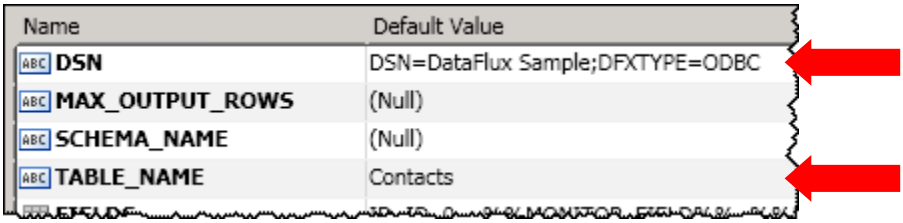
The data job opens and displays the following data flow:



5. Review the properties of the Data Source node.

a. Right-click the node labeled **Contacts Table** and select **Advanced Properties**.

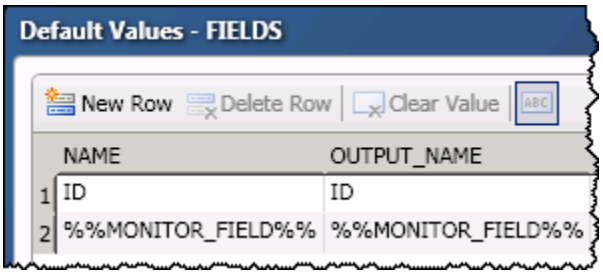
b. Verify that the selected table is the **Contacts** table from the **DataFlux Sample** data connection.



c. Click the **FIELDS** attribute.

d. Click **Edit Default Value**.

e. Verify that two fields are specified: the **ID** field and a field passed via the **MONITOR_FIELD** macro variable.



f. Click **Cancel** to close the Default Values window.

g. Click **Cancel** to close the Advanced Properties window.

6. Review the properties of the Expression node.

a. Right-click the node labeled **Set Monitor File Name** and select **Properties**.

b. If necessary, click the **Expression** tab.

c. Verify that the pre-processing expression code assigns a value to a variable **fn**, and then assigns this value to the **MONITOR_FILE** macro variable with the following code:

```
hidden string filename

filename = 'Ch12D4_Null_or_Blank_Records_for_'
                                    and getvar("MONITOR_FIELD") &
'_Field.txt'

setvar("NEWFILE", filename)
```

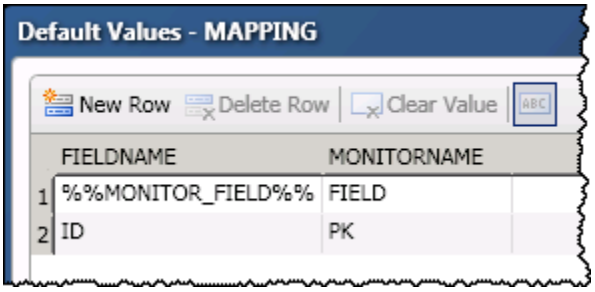
d. Click the main item on the thread (labeled **Ch12D4_Macros_in_Monitoring**) to return to the job flow.

7. Review the properties of the Data Monitoring node.

a. Right-click the node labeled **Data Monitoring for Null or Blanks** and select **Advanced Properties**.

b. Click the **MAPPING** property.

c. Click **Edit Default Value**.



Recall the Monitor_for_Null_or_Blanks rule (used in the same-named task) was defined with two fields, one named **FIELD** and one named **PK**. These two fields are identified in the MONITORNAME column.

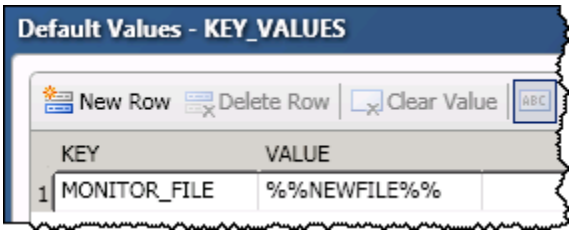
The FIELDNAME column is identifying the fields from the data job's flow that should be mapped to the fields defined in the business rule.

The field being monitored is parameterized (referenced as **%%MONITOR_FIELD%%**) and paired with the **FIELD** field. Also, the **ID** field is paired to the **PK** field.

d. Click **Cancel** to close the Default Values window.

e. Click the **KEY_VALUES** property.

f. Click **Edit Default Value**.

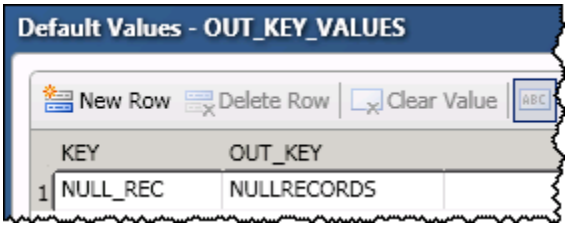


The event **Log error to text file** references a variable **MONITOR_FILE**, and this variable will need a value. The KEY_VALUES attribute for the Data Monitoring node can be used to pass information in to the node. In this example, we are pairing the calculated value for the **NEWFILE** macro variable (recall that **NEWFILE** was dynamically created in the Expression node prior to this Data Monitoring node) with the **MONITOR_FILE** variable. Thus, **MONITOR_FILE** is being assigned the value of the **NEWFILE** macro variable.

g. Click **Cancel** to close the Default Values window.

h. Click the **OUT_KEY_VALUES** property.

i. Click **Edit Default Value**.



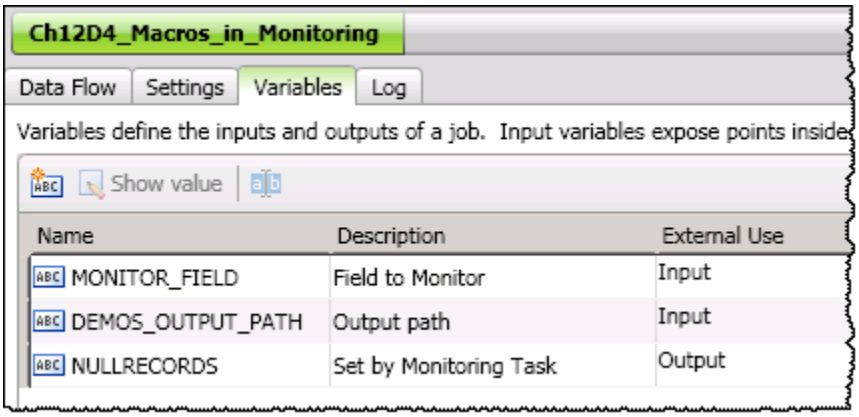
The event **Set a data flow key/value** defines a variable **NULL_REC** with a value of *TRUE* (***if*** there are triggers). This variable's value can be used subsequently in a data job if you want. However, it must be passed out of the Data Monitoring node using the OUT_KEY_VALUES attribute for the Data Monitoring node. In this example, the value of the **NULL_RECS** variable will be assigned to a variable named **NULLRECORDS**.

In this data job, the **NULLRECORDS** variable will be defined as an output variable for the data job. The ability to pass variables with values out of a data job becomes important when using the data job within a process job. For this example, we will simply verify that the output variable is defined with the correct value.

j. Click **Cancel** to close the Default Values window.

k. Click **Cancel** to close the Advanced Properties window.

8. Review the Variables tab of the data job.
   a. Click the **Variables** tab.



   b. Verify that there are two ***input*** variables defined (**DEMOS_OUTPUT_PATH** and **MONITOR_FIELD**).

      In our example, both input variables have a corresponding variable defined in an active macro configuration file. However, if this data job is imported to a Data Management Server instance, the job can be run on the server with run-time specifications for these two variables.

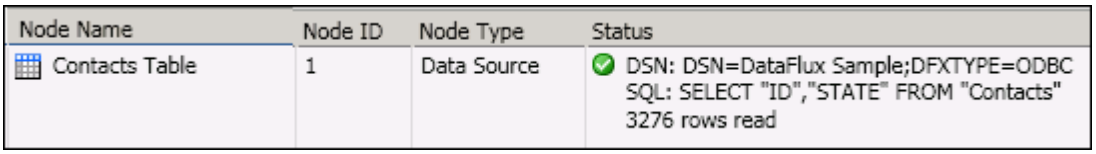
   c. Verify that there is an ***output*** variable defined named **NULLRECORDS**.

      In our example, the **Set a data flow key/value** event created a variable (**NULL_RECS**) that was mapped to **NULLRECORDS** for use “later” in the data job. This defined output variable (of the same name) is a use of this variable.

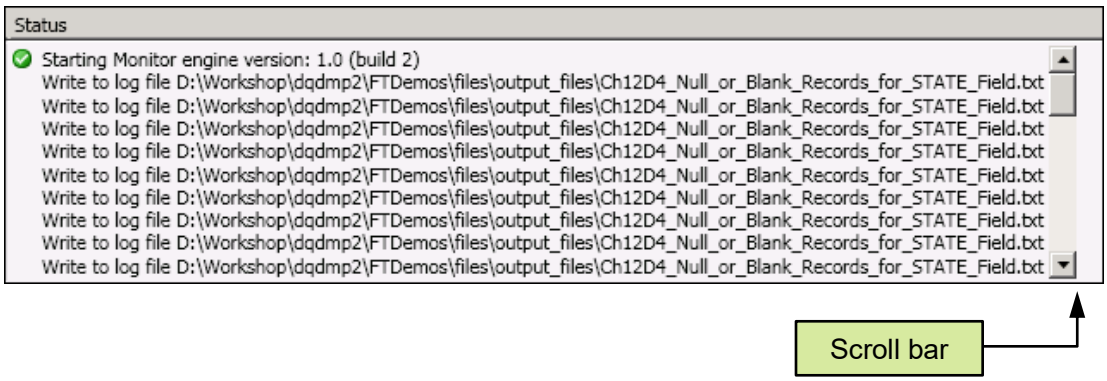
   d. Click the **Data Flow** tab.

9. Select **Actions** ⇨ **Run Data Job**.

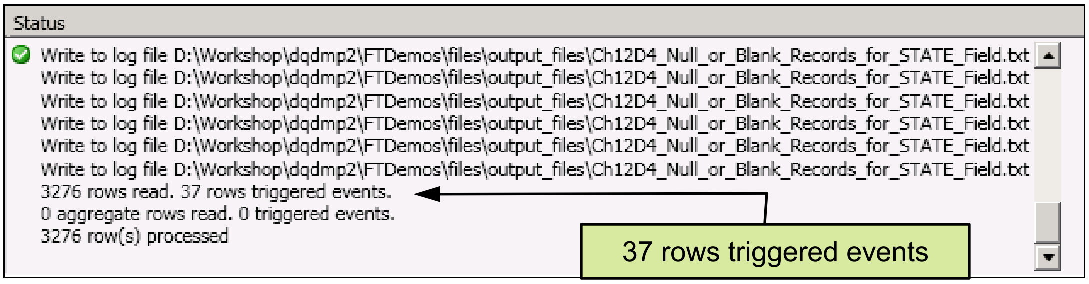
10. Click the **Log** tab.
    a. Verify that 3276 rows were read from the source table.



    b. Verify that there were multiple triggers (multiple messages regarding writing to text file).
       1) Expand the **Status** field in the log to the right when the scroll bar appears for the Data Monitoring row.

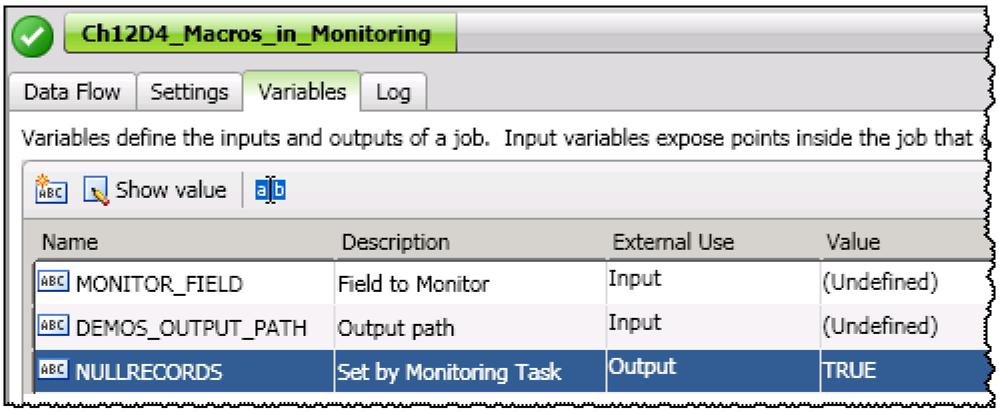


2) Scroll to the bottom and verify that there were 37 triggers.



11. Click the **Variables** tab.

12. Verify that the output variable **NULLRECORDS** has a run-time value of *TRUE* defined in the monitoring task.



13. Select **File** ⇨ **Close** to close the data job.

14. View the file that errors were logged to.

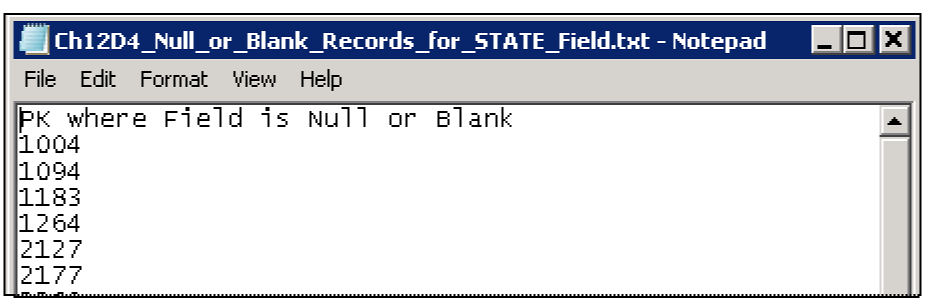
   a. If necessary, click the **Home** tab.

   b. If necessary, click the **Folders** riser bar.

   c. If necessary, expand **Advanced Demos**.

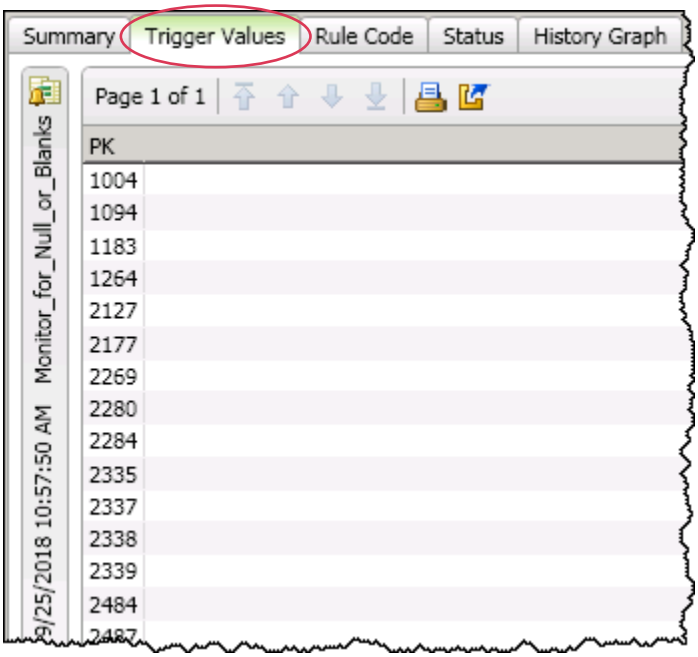
   d. Click the **output_files** folder.

   e. Double-click the file created, **Ch12D4_Null_or_Blank_Records_for_STATE_Field.txt**.



   f. Select **File** ⇨ **Exit** to close the Notepad window.

15. View the logged information in the repository.
    a. Select **Tools** ⇨ **Monitoring Report** ⇨ **Advanced Demos**.
    b. Verify that there were 37 triggers for the rule **Monitor_for_Null_or_Blanks**.
    c. With the rule **Monitor_for_Null_or_Blanks** selected, click the **Trigger Values** tab.



    d. Select **File** ⇨ **Close** to close the Monitoring Report.
16. If necessary, select **File** ⇨ **Close** to close the data job **Ch12D4_Macros_in_Monitoring**.

End of Demonstration

Exercise (Optional)

1. Using Macros in Data Monitoring

This exercise investigates an existing business rule. A new task is created using the business rule with two events and with one of the events parameterized. An existing data job is then updated to appropriately work with the parameterized event.

- Investigate the rule **Field_Length_Check** in the **Advanced Exercises** repository.
  - Verify that the rule is checking for the length of a field not equaling 3.
  - Verify that the rule has two fields for output.
- Create a task using the rule **Field_Length_Check**.
  - Name the task **Field_Length_Check**.
  - For **Rule Details**, select two events: **Log error to repository** and **Log error to text file**.
    - For the **Log error to repository** event, select the **PK** field.
    - For the **Log error to text file** event:
      - Parameterize the path and file to be created using **%%EXERCISES_OUTPUT_PATH%%%%NEWFILE%%**

        Note: **EXERCISES_OUTPUT_PATH** is a macro variable defined in the Advanced Exercises macro file.

        NEWFILE is a macro variable that needs to be defined dynamically because it will contain the name of the field whose length is being checked.
      - Specify **PK, FIELD** as the header.
      - Specify **%%PK%%, %%FIELD%%** as the message.
      - Specify **– End of File –** as the footer.
- Update the data job named **Ch12E6_Macros_Monitoring_Start** in the **Advanced Exercises** repository.
  - Open the job and then select **File** ⇨ **Save As**. Enter a name of **Ch12E6_Macros_Monitoring**.
  - Preview the Text File Input node and verify that the **COUNTRY** field has some values where the length is not 3. (Hence, the rule **Field_Length_Check** will have triggers.)
  - In the Expression node, enter pre-processing code:
    - Declare a string field named **FN** with a scope of hidden.
    - Assign **FN** to a value similar to **Ch12E6_*COUNTRY*_ValuesWhereLengthNot3.txt** where the **EXERCISES_MONITOR_FIELD** macro variable supplies the value ***COUNTRY***.
    - Use the setvar() function to assign **FN** to a new macro variable **FILENAME**.
    - Test the construction of the **FN** field by removing the hidden scope and previewing the Expression node. When you are satisfied with the value of **FN**, add the hidden scope back to **FN**.

- In the Data Monitoring Properties window, select the task named **Field_Length_Check**, and pair the **ID** field to the **PK** field.
- In the Data Monitoring Advanced Properties window, configure the following attributes:
  - For the MAPPING attribute, pair **%%EXERCISES_MONITOR_FIELD%%** with **FIELD**, and verify that **ID** is paired with **PK**.

    Note: You need to enter **%%EXERCISES_MONITOR_FIELD%%**.
  - For the KEY_VALUES attribute, define a new row where the KEY is **NEWFILE** and the VALUE is **%%FILENAME%%**.

    Note: You need to enter both the KEY and VALUE information.

- Save and run the data job.

  Question: Did the data job run successfully?

  Answer: _____

  Question: Were there any triggers to the rule executed via the Data Monitoring node?

  Answer: _____

  - Verify that the Monitoring Report for the Advanced Exercises repository has a list of "bad" data on the Trigger Values tab.
  - Verify that the text file created from the **Log error to text file** event looks appropriate.

- **(BONUS - Challenge)** Update the task **Field_Length_Check** to add a third event, **Set a data flow key/value**. Also add a data job output variable to show the use of this variable.
  - For the new event **Set a data flow key/value**, define a key with name of **FOUND_LENGTH_NOT_THREE** and with a value of *TRUE*.
  - Add an output variable to the data job named **LEN_NOT_3**.
  - Update the advanced properties for the Data Monitoring node. For the OUT_KEY_VALUES attribute, define a new row where the KEY is **FOUND_LENGTH_NOT_THREE** and the OUT_KEY is **LEN_NOT_3**.

    Note: In order to use this key variable's value, it must be passed out of the Data Monitoring node using the OUT_KEY_VALUES advanced property.
  - Save and rerun the data job.

    Question: Does the output variable created have a run-time value of *TRUE*?

    Answer: _____

    Question: What would need to be done to replace the hardcoded length of 3 with a macro variable reference?

    Answer: _____

End of Exercises

Solutions

Solutions to Exercises

1. **Using Macros in Data Monitoring**
   - **a.** If necessary, invoke Data Management Studio.
     1) Select **Start** ⇨ **All Programs** ⇨ **DataFlux** ⇨ **Data Management Studio 2.7**.
     2) Click **Cancel** in the Log On window.
   - **b.** Investigate the rule **Field_Length_Check** in the **Advanced Exercises** repository.
     1) Select **Tools** ⇨ **Business Rules Manager** ⇨ **Advanced Exercises**.
     2) Expand the **Rules** folder.
     3) Click the rule **Field_Length_Check**.
     4) Verify that the check is **Length of FIELD is not equal to 3**.
     5) Verify that there are two defined fields, **PK** and **FIELD**.
   - **c.** Create a task using the rule **Field_Length_Check**.
     1) Right-click the **Tasks** folder and select **New Task**.
     2) Name the task **Field_Length_Check**.
     3) Double-click the rule **Field_Length_Check** to move it to the Selected list.
     4) Verify that the rule **Field_Length_Check** is selected in the Selected list.
     5) Click **Rule Details**.
     6) Click **Add** in the Rule Details window.
        a) Select **Log error to repository**.
        b) Click **Continue**.
        c) Double-click the **PK** field to move it to the Selected fields list.
        d) Click **OK**.
     7) Click **Add** in the Rule Details window.
        a) Select **Log error to text file**.
        b) Click **Continue**.
        c) Enter **%%EXERCISES_OUTPUT_PATH%%%%NEWFILE%%** for **File**.
        d) Enter **PK, FIELD** for **Header**.
        e) Enter **%%PK%%, %%FIELD%%** for **Message**.
        f) Enter **– End of File –** for **Footer**.
        g) Click **OK**.
   - **d.** Rename the data job **Ch12E6_Macros_Monitoring_Start**.
     1) Click the **Home** tab.
     2) Click the **Folders** riser bar.

3) Click the **batch_jobs** folder under **Advanced Exercises**.
4) Double-click the data job **Ch12E6_Macros_Monitoring_Start**.

   The data job opens on a primary tab.
5) Select **File** ⇨ **Save As**.
   a) Enter **Ch12E6_Macros_Monitoring** for Name.
   b) Click **Save**.

e. Review the Text File Input node in the data job.

1) If necessary, toggle on the Details pane by selecting **View** ⇨ **Show Details Pane**.
2) Right-click the **Text File Input** node and select **Preview**.
3) Verify that **COUNTRY** has values that are not three characters. (Hence, there will be triggers for the rule **FIELD_LENGTH_CHECK**).

	ID	NAME	ORGANIZATION	ADDRESS	CITY	STATE_PROVINCE	POSTALCODE	COUNTRY
1	1	Ewan Woodard	Sterling Financial...	259 Nigel Road	Toronto	ON	A20417	CANADA
2	2	Clay Xavier M...	Denali Ventures	477 Billerica	Montreal	QC	15062	Canada
3	3	Mary-Louise R...	Cura Group	46 Liam Drive	Mechanicsburg	PA	15245	USA
4	4	Famke Fiennes	SurModics	51st Street NW	Washington	DC	21627	USA
5	5	King Herrmann	Spotfire Holdings	64 Denny Drive	Chinnor		3X5 7K1	United Kingdom
6	6	Bridgette S. L...	KSJ & Associates	55 Peniston Road	Sugar Land	TX	18666	USA
7	7	Carol Bosco	L.E.M. Products	277 Ticotin Drive	Regina	SK	5I7 1O7	Canada
8	8	Powers Mars	Acsis	392 Camp Street	Carson City	NV	76384	USA
9	9	Gerald Pepper	Random Walk Co...	50 Cube Road	Slough		7T4 2Q8	United Kingdom
10	10	Andrea Butler	MHF Logistical So...	84 Fred Road Ste...	West Windsor	NJ	16968	USA
11	11	Nastassja Danes	Pan-Pacific Retail...	468 Latin Street	Cannock		4X9 8T9	United Kingdom
12	12	Raymond Rispoli	PowerLight	3 McCain Road #...	Bloomington	IN	24995	USA

f. Specify Expression node code.

1) Double-click the **Expression** node.
2) Enter the following code in the Pre-processing section:

```
hidden string FN
FN='Ch12E6_'&getvar("EXERCISES_MONITOR_FIELD")&
                                              '_ValuesWhereLengthNot3.txt'
setvar("FILENAME", FN)
```

3) Click the main item on the thread to return to the data job flow.

g. Update the properties of the Data Monitoring node.

1) Double-click the **Data Monitoring** node to open the Properties window.
   a) Select the task **Field_Length_Check**.
   b) Select **ID** as the variable to pair to **PK**.
   c) Click **OK** to close the Properties window.
2) Right-click the Data Monitoring node and select **Advanced Properties**.
   a) Right-click the **MAPPING** attribute and select **Edit Default Value**.
      (1) Enter **%%EXERCISES_MONITOR_FIELD%%** as the **FIELDNAME** value (the pairing for **FIELD**).

(2) Click **OK** to close the Default Values window.

b) Right-click the **KEY_VALUES** attribute and select **Edit Default Value**.

(1) Click **New Row**.

(2) Enter **NEWFILE** as the KEY value.

(3) Enter **%%FILENAME%%** as the VALUE value.

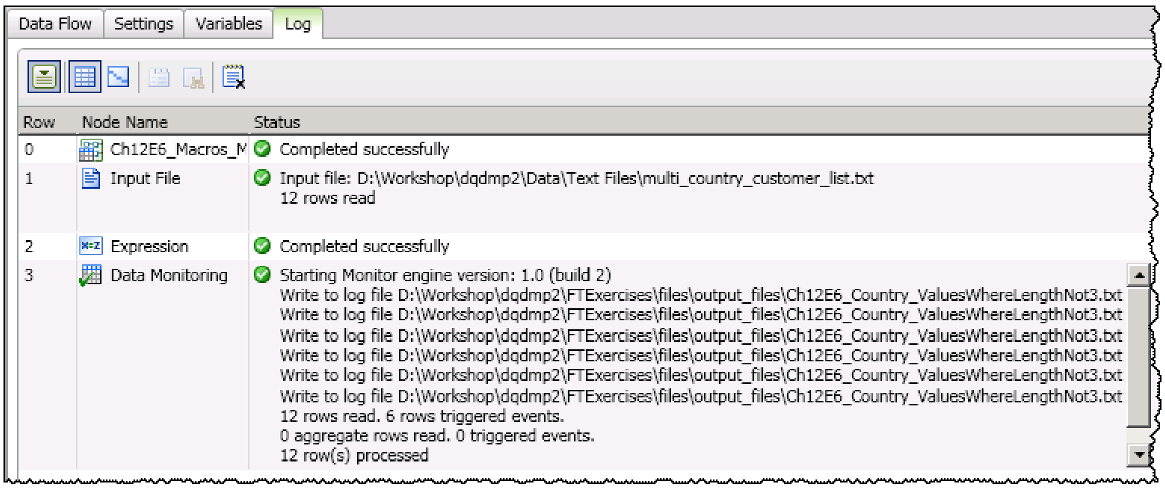
(4) Click **OK** to close the Default Values window.

c) Click **OK** to close the Advanced Properties window.

h. Select **File** ⇨ **Save** to save the data job.

i. Select **Actions** ⇨ **Run Data Job**.

j. Click the **Log** tab.



Question: Did the data job run successfully?

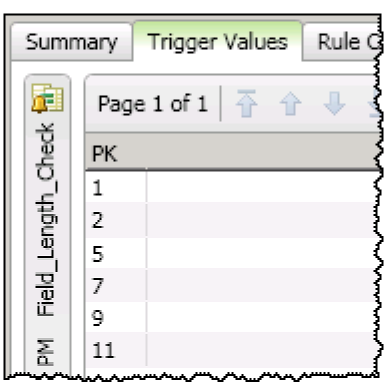
Answer: **Yes**

Question: Were there any triggers to the rule executed via the Data Monitoring node?

Answer: **Six rows triggered events**

k. Select **Tools** ⇨ **Monitoring Report** ⇨ **Advanced Exercises**.

1) Verify that the **Field_Length_Check** rule has six triggers listed on the Trigger Values tab.

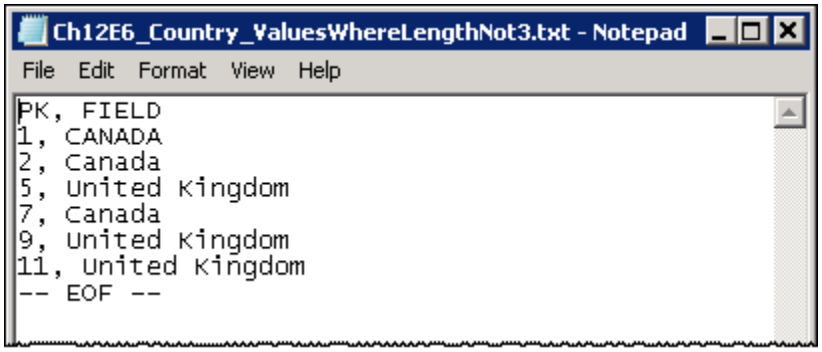


2) Select **File** ⇨ **Close** to close the Monitoring Report.

l. Verify that the text file was created.

1) Click the **Home** tab.
2) If necessary, click the **Folders** riser bar.
3) If necessary, expand **Advanced Exercises** ⇨ **output_files**.
4) Double-click **Ch12E6_Country_ValuesWhereLengthNot3.txt**.

   The file opens in Notepad window.

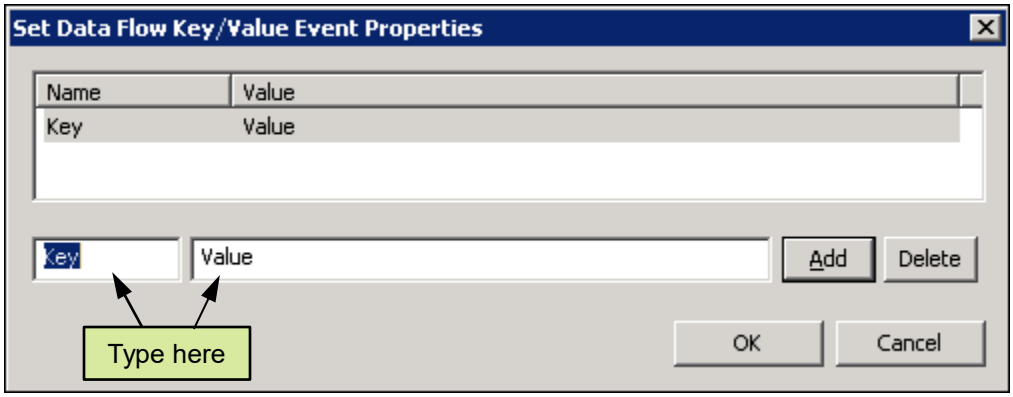


5) Select **File** ⇨ **Exit** to close the Notepad window.

Start of Bonus / Challenge Steps:

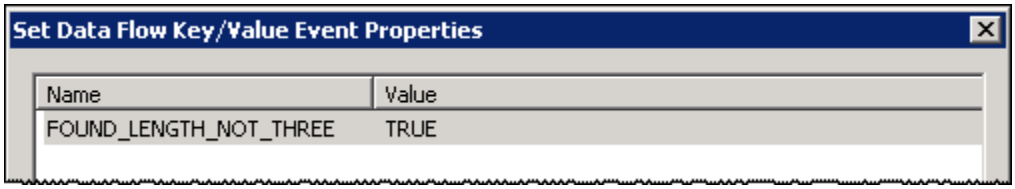
m. Update the task to add a third event.

1) If necessary, select **Tools** ⇨ **Business Rules Manager** ⇨ **Advanced Exercises**.
2) If necessary, expand the **Tasks** folder.
3) Right-click the **Field_Length_Check** task and select **Edit**.
4) Click **Rule Details** (under **Selected**, with the **Field_Length_Check** rule selected).
5) Click **Add** in the Rule Details window.
6) Select **Set a data flow key/value**.
7) Click **Continue**.
8) Click **Add** in the Set Data Flow Key/Value Event Properties window.



9) Enter **FOUND_LENGTH_NOT_THREE** in place of **Key**.
10) Enter **TRUE** in place of **Value**.

The new name/value pair appears at the top of the window:



11) Click **OK** to close the Set Data Flow Key/Value Event Properties window.

12) Click **OK** to close the Rule Details window.

13) Click **OK** to close the Task Properties window.

n. If necessary, open the job **Ch12E6_Macros_Monitoring**

1) Click the **Home** tab.

2) Click the **Folders** riser bar.

3) Click the **batch_jobs** folder under **Advanced Exercises**.

4) Double-click the data job **Ch12E6_Macros_Monitoring**.

The data job opens on a primary tab.

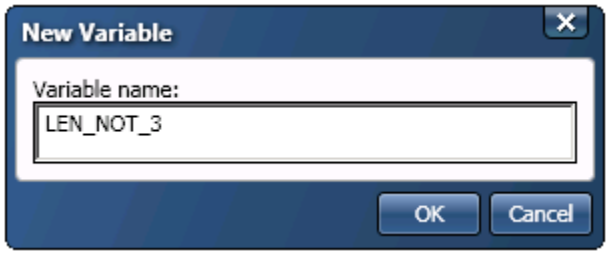
o. Add an output variable to the data job.

1) Click the **Variables** subtab.

2) Click (**Insert New Variables**).

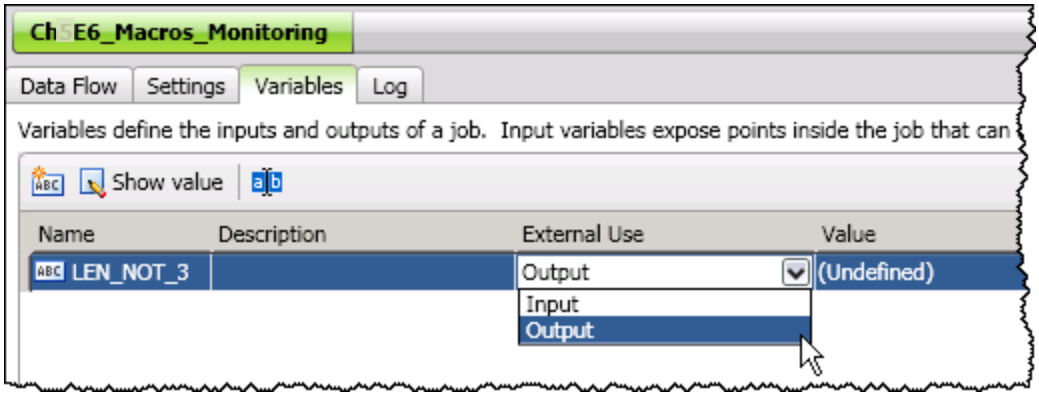
The New Variable window appears.

a) Enter **LEN_NOT_3**.



b) Click **OK** to close the New Variable window.

3) Change **External Use** to **Output**.



p. Select **File** ⇨ **Save** to save the data job.

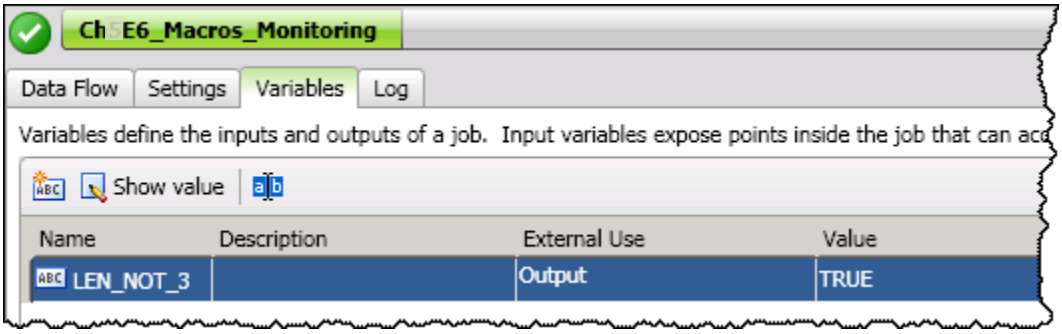
q. Update the OUT_KEY_VALUES attribute for the Data Monitoring node.

1) Click the **Data Flow** subtab.
2) Right-click the **Data Monitoring** node and select **Advanced Properties**.
   a) Right-click the **OUT_KEY_VALUES** attribute and select **Edit Default Value**.
      (1) Click **New Row**.
      (2) Enter **FOUND_LENGTH_NOT_THREE** as the KEY value.
      (3) Enter **LEN_NOT_3** as the OUT_KEY value.
      (4) Click **OK** to close the Default Values window.
   b) Click **OK** to close the Advanced Properties window.

r. Select **File** ⇨ **Save** to save the data job.

s. Select **Actions** ⇨ **Run Data Job**.

t. Click the **Variables** tab.



Question: Does the output variable created have a run-time value of *TRUE*?

Answer: **Yes**

Question: What would need to be done to replace the hardcoded length of 3 with a macro variable reference?

Answer: **The rule expression code would need to be edited, replacing the hardcoded value of 3 with a retrieved value of a macro variable. For example, if a macro variable named FIELDLENGTH was defined, the rule expression code would change from**

```
if len(`FIELD`) != 3 then
   return true
else
   return false
```

to the following:

```
if len(`FIELD`) != getvar("FIELDLENGTH") then
   return true
else
   return false
```

u. Select **File** ⇨ **Close** to close the data job.