



Data Analysis and Integration

Lab 4: String matching

Using the Calculator to compare strings

1. Open a new terminal and change to the directory: **cd ~/Pentaho/data-integration**
2. Start Pentaho Data Integration (PDI) with: **./spoon.sh**
3. In the **File** menu, select **New > Transformation**.
4. In the **Design** tab, expand **Input**, and drag a **Data Grid** step to the canvas.
5. Double-click the **Data Grid** to configure it.
6. In the **Meta** tab, define the following columns:
 - Name: **string1** Type: **String**
 - Name: **string2** Type: **String**
7. In the **Data** tab, add the following line:
 - string1: **Austria** string2: **Autriche**
8. Press **OK** to close the **Data Grid** configuration.
9. In the **Design** tab, expand **Transform**, and drag a **Calculator** step to the canvas.
10. Create a hop from the **Data Grid** step to the **Calculator** step.
11. Double-click the **Calculator** step to configure it.
12. Add the following field:
 - New field: **levenshtein**
 - Calculation: **Levenshtein distance (source A and target B)**
 - Field A: **string1**
 - Field B: **string2**
13. Press **OK** to close the **Calculator** configuration.
14. Preview the **Calculator** step.
15. Check that the result agrees with what you would expect from this string measure.

Adding more strings

16. Double-click the **Data Grid** to configure it.

17. In the **Data** tab, add the following lines:

- string1: **Ireland** string2: **Ierland**
- string1: **Dinamarca** string2: **Danimarca**
- string1: **Chipre** string2: **Cypren**
- string1: **Österreich** string2: **Österrike**

18. Preview the **Calculator** step.

19. Check that the results agree with what you would expect from this string measure.

Adding other measures

20. Double-click the **Calculator** step to configure it.

21. Add the following fields:

- New field: **damerau**
- Calculation: **DamerauLevenshtein distance between String A and String B**
- Field A: **string1**
- Field B: **string2**

- New field: **needleman**
- Calculation: **NeedlemanWunsch distance between String A and String B**
- Field A: **string1**
- Field B: **string2**

- New field: **jaro**
- Calculation: **Jaro similitude between String A and String B**
- Field A: **string1**
- Field B: **string2**

- New field: **jarow**
- Calculation: **JaroWinkler similitude between String A and String B**
- Field A: **string1**
- Field B: **string2**

22. Preview the **Calculator** step.

23. Take a moment to recall how these measures work and to check that the results agree with what you would expect from these measures.

Names of European countries in different languages

24. Download the file **languages.csv** to your Downloads folder (/home/aid/Downloads).

25. Open the file in a text editor to inspect its format.

26. Open the same file in **LibreOffice Calc**.
27. Use **Semicolon** as the separator between fields.
28. Take a moment to check how country names are written in different languages.

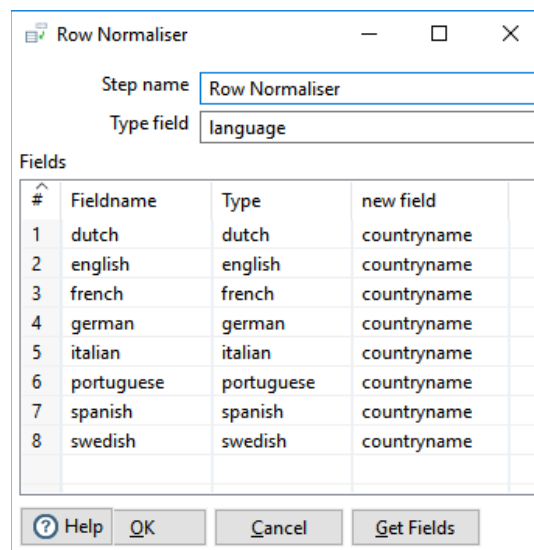
Reading the CSV file in PDI

29. In Pentaho Data Integration (PDI), create a new transformation.
30. In the **Design** tab, expand **Input**, and drag a **CSV file input** step to the canvas.
31. Double-click the **CSV file input** and configure it as follows:
 - In **Filename**, write **/home/aid/Downloads/languages.csv** (if you are on the VM)
 - In **Delimiter**, replace the comma (,) with a semicolon (;)
 - Uncheck the option **Lazy conversion**.
 - Leave the option **Header row present** checked.
 - Set the **File encoding** to **UTF-8**.Do not close the configuration dialog just yet.
32. Press the **Get Fields** button. This will sample some lines from the CSV file.
33. In the **Sample size** window, you can leave the default size of 100. Press **OK**.
34. Check that all fields have been identified correctly.
35. Press **OK** to close the **CSV file input** configuration.
36. Preview the **CSV file input** step and check that the file contents have been read correctly.

Changing the structure of the rowset

37. In the **Design** tab, expand **Transform**, and drag a **Row Normaliser** step to the canvas.
38. Connect the **CSV file input** step to the **Row Normaliser** step (choose **Main output of step**).
39. Double-click the **Row Normaliser** step and configure it as follows:
 - In **Type field** write **language**.
 - Click the **Get Fields** button to get the fields.
 - Right-click and delete the **countrycode** field from the list (line 1).

- Set **new field** to **countryname** in every field.



Row Normaliser

Step name: Row Normaliser

Type field: language

Fields

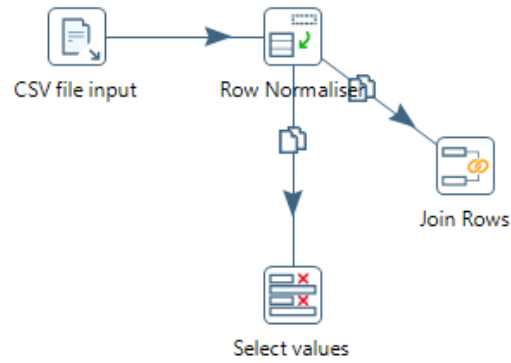
#	Fieldname	Type	new field
1	dutch	dutch	countryname
2	english	english	countryname
3	french	french	countryname
4	german	german	countryname
5	italian	italian	countryname
6	portuguese	portuguese	countryname
7	spanish	spanish	countryname
8	swedish	swedish	countryname

Buttons: Help, OK, Cancel, Get Fields

40. Preview the **Row Normaliser** step and check the new structure of the rowset. What has changed?
41. Without closing the **Examine preview data** window, preview the **CSV file input** step and compare the results side by side.

Generating all combinations of rows

42. In the **Design** tab, expand **Joins**, and drag a **Join Rows (cartesian product)** step to the canvas.
43. Double-click the **Join Rows (cartesian product)** and change the **Step name** to simply **Join Rows**.
44. Connect the **Row Normaliser** step to the **Join Rows** step.
45. In the **Design** tab, expand **Transform**, and drag a **Select values** step to the canvas.
46. Connect the **Row Normaliser** step to the **Select values** step. A warning message will appear asking how the rows should be sent to the two destination steps. Choose **Copy**.



47. Double-click the **Select values** and configure it as follows:

- In the **Select & Alter** tab, press the **Get fields to select** button.
- Next to **countrycode**, write **countrycode2** in the second column (**Rename to**)
- Next to **language**, write **language2** in the second column (**Rename to**)
- Next to **countryname**, write **countryname2** in the second column (**Rename to**)

48. Preview the **Select values** step and check that the fields are being renamed as specified above.

49. Connect the **Select values** step to the **Join Rows** step (choose **Main output of step**).

50. Preview the **Join Rows** step and check that it is generating all combinations of rows.

51. Use **Stop** to close the **Examine preview data** window (because there are many more rows).

52. Configure the **Join Rows** step with the two conditions:

countrycode = countrycode2

AND

language <> language2

(See the following figure.)

Join rows

Step name: Join Rows

Temp directory: %%java.io.tmpdir%% [Browse...](#)

TMP-file prefix: out

Max. cache size (in rows): 500

Main step to read from: [▼](#)

The condition:

To edit a subcondition, simply click on it [+](#)

countrycode = countrycode2

AND

language <> language2

[? Help](#) [OK](#) [Cancel](#)

53. Why is the first condition being imposed? And why is the second one being imposed?

54. Preview the **Join Rows** step and check that the results agree with the two conditions above.

55. Again, use **Stop** to close the **Examine preview data** window.

Comparing the country names

56. In the **Design** tab, expand **Transform**, and drag a **Calculator** step to the canvas.

57. Connect the **Join Rows** step to the **Calculator** step.

58. Double-click the **Calculator** step to configure it.

59. Add the following field:

- New field: **measure**
- Calculation: **Levenshtein distance (source A and target B)**
- Field A: **countryname**
- Field B: **countryname2**
- Value type: **Number**

60. Preview the **Calculator** step and check the results.

Comparing the languages by average measure

61. In the **Design** tab, expand **Transform**, and drag a **Sort rows** step to the canvas.

62. Connect the **Calculator** step to the **Sort rows** step.
63. Configure the **Sort rows** step to sort by **language** and **language2**, both in ascending order.
64. Preview the **Sort rows** step and check the results are being sorted as desired.
65. In the **Design** tab, expand **Statistics**, and drag a **Group by** step to the canvas.
66. Connect the **Sort rows** step to the **Group by** step.
67. Configure the **Group by** step as follows:
- The **Group fields** are **language** and **language2**.
 - The **Aggregate** is:
 - o Name: **avg_measure**
 - o Subject: **measure**
 - o Type: **Average (Mean)**
68. Preview the **Group by** step and check the results for each pair of languages.

Finding the most similar languages

69. From the **Design** tab, drag another **Sort rows** step to the canvas (**Sort rows 2**).
70. Connect the **Group by** step to the **Sort rows 2** step.
71. Configure the **Sort rows 2** step to sort by **language** and **avg_measure**, both in ascending order.
72. Preview the **Sort rows 2** step and check that there is a separate group for each language.
73. From the **Design** tab, drag another a **Group by** step to the canvas (**Group by 2**).
74. Connect the **Sort rows 2** step to the **Group by 2** step.
75. Configure the **Group by 2** step as follows:
- The **Group field** is **language**.
 - The **Aggregates** are:

o Name: language2	Subject: language2	Type: First value
o Name: avg_measure	Subject: avg_measure	Type: First value
76. Preview the **Group by 2** step and check that now there is only one match (the best match) for each language.
77. From the **Design** tab, drag another **Sort rows** step to the canvas (**Sort rows 3**).

78. Connect the **Group by 2** step to the **Sort rows 3** step.
79. Configure the **Sort rows 3** step to sort by **avg_measure** in ascending order.
80. Preview the **Sort rows 3** step. You should get the following results:

#	language	language2	avg_measure
1	portuguese	spanish	0.8571428571
2	spanish	portuguese	0.8571428571
3	italian	spanish	1.75
4	german	swedish	2.5
5	swedish	german	2.5
6	dutch	swedish	2.9285714286
7	french	spanish	3.0357142857
8	english	italian	3.1071428571

Exercise

81. Change the transformation to use the Damerau-Levenshtein distance. Do the results change? What conclusions can you draw from here?
82. Change the transformation to use the Needleman-Wunsch measure. Do the results change?
Note: The Needleman-Wunsch measure is better if its values are higher. So you will have to change the way the avg_measure is being sorted at two different places in the transformation.
83. Change the transformation to use the Jaro measure. Is there any change in the results, besides the measure values?



84. Change the transformation to use the Jaro-Winkler measure. How do they compare to the results of the Jaro measure? Why the difference?
Note: You need to do the previous steps for this exercise to work correctly.