Challenge May 2022
Medical Claim Processing

A solution with DT5GL by Jack Jansonius – 25 September 2022

Problem Statement (from the website):

This CSV file "ICD10Codes.csv" consists of two columns that represent different incompatible diagnoses:

Column 1,Column 2
A48.5,A05.1
K75.0,A06.4
K75.0,K83.09
K75.0,K75.1
G07,A06.6
G07,B43.1
...

The actual file contains around 70,000 code combinations. Your claim processing decision service receives a set of claim diagnosis codes such as {M43.6, F45.8} as its input. By comparing these codes with those found in Column 1 and Column 2, it should validate the claim by producing the errors "**Diagnosis Code [XXX] cannot be reported together with [YYY]**" in the following situations:

1. Diagnosis Code is found in Column 1 while another Diagnosis Code found in Column 2
2. Diagnosis Code is found in Column 2 while another Diagnosis Code found in Column 1.

Keep in mind that some diagnoses can be found in both columns, and your service should not produce duplicate errors.

Here are several test cases for your service to run:

1. D47.02,C94.32
2. E71.313,E72.3
3. R29.891, M43.6, F45.8
4. D75.81,C94.42

Tables in the database:

icd10codes:

```
SELECT *
  FROM icd10codes;
```

Grid view | Form view

Total rows loaded: 69763

	column1	column2
1	A48.5	A05.1
2	K75.0	A06.4
3	K75.0	K83.09
4	K75.0	K75.1
5	G07	A06.6
6	G07	B43.1
7	G07	A54.82
8	G07	A17.81
9	G07	A17.1
10	N51	A06.8
11	N51	B37.42
12	N51	A54.23
13	N51	A54.22
14	N51	A60.01
15	N51	A59.02
16	N51	A18.14
17	N48.1	A06.8
18	N48.1	N48.0
19	N48.1	B37.42
20	N48.1	A54.23
21	N48.1	A60.01
22	B60	A07.2
23	B60	A07.8
24	B60	A07.3
25	J98.11	A15
26	J60	A15
27	J60	J65
28	J62	A15
29	J62	J65

input:

```
SELECT *
  FROM input;
```

Grid view | Form view

	tc	code
1	1	D47.02
2	1	C94.32
3	2	E71.313
4	2	E72.3
5	3	R29.891
6	3	M43.6
7	3	F45.8
8	4	D75.81
9	4	C94.42
10	5	D75.81
11	5	C94.32
12	6	D47.02
13	6	C94.32
14	6	E71.313
15	6	E72.3
16	6	R29.891
17	6	M43.6
18	6	F45.8
19	6	D75.81
20	6	C94.42
21	6	D75.81
22	6	C94.32
23	7	D47.02
24	7	C94.32
25	7	C94.32
26	7	D47.02
27	7	D47.02
28	7	C94.32

and a table ‘found’ to record code combinations already found:

```
SELECT *
  FROM found;
```

Grid view | Form view

code1	code2

Implementation of the decision model in DT5GL (Solution 1):

```
SQLite_database: "Database/ClaimTC.db"

Table 0:
If:                                              | 0| 1|
'next testcase number selected'                  | Y| N|
Then:
NextTC is Selected                               | X|  |
NextTC is Finished                               |  | X|
# .......
# Repeat until: Finished

Proposition: 'next testcase number selected'
Obtain_instance_from_database_view: testcase


Table 1:
If:                                              | 0| 1|
'next code 1 selected'                           | Y| N|
Then:
FirstCode is Selected                            | X|  |
FirstCode is Finished                            |  | X|
# .......
# Repeat until: Finished

Proposition: 'next code 1 selected'
Obtain_instance_from_database_view: input1


Table 2:
If:                                              | 0| 1| 2| 3| 4|
'next code 2 selected'                           | Y| Y| Y| Y| N|
input1.code <> input2.code                       | Y| Y| Y| N| -|
'combo already reported'                         | Y| N| N| -| -|
'combo present in codes'                         | -| Y| N| -| -|
Then:
EvalCombo is NotReported                         | X|  | X| X|  |
EvalCombo is Reported                            |  | X|  |  |  |
EvalCombo is Finished                            |  |  |  |  | X|
# .......
# Repeat until: Finished


Proposition: 'next code 2 selected'
Obtain_instance_from_database_view: input2

Proposition: 'combo present in codes'
Obtain_instance_from_database_view: code

Proposition: 'combo already reported'
Obtain_instance_from_database_view: already_found_combos

Attribute: input1.code      Type: Text
Attribute: input2.code      Type: Text
```

```
########################## Database views ############################


Database_view: testcase
With_attributes:
number
Query:
 SELECT distinct tc
  FROM input
 LIMIT 1 OFFSET %s
With_arguments: testcase.auto_index


Database_view: input1
With_attributes:
code
Query:
SELECT distinct code
  FROM input
 WHERE tc = %s
 LIMIT 1 OFFSET %s
With_arguments: testcase.number, input1.auto_index

Database_view: input2
With_attributes:
code
Query:
SELECT distinct code
  FROM input
 WHERE tc = %s
 LIMIT 1 OFFSET %s
With_arguments:  testcase.number, input2.auto_index


Database_view: code
With_attributes:
column1, column2
Query:
SELECT column1, column2
  FROM icd10codes
 WHERE column1 = '%s' AND column2 = '%s'
       OR
       column1 = '%s' AND column2 = '%s'
 LIMIT 1
With_arguments: input1.code, input2.code, input2.code, input1.code


Database_view: already_found_combos
With_attributes:
code1, code2
Query:
SELECT code1, code2
  FROM found
 WHERE code1 = '%s' AND
       code2 = '%s'
 LIMIT 1
With_arguments: input2.code, input1.code
```

```
########################## GoalAttributes ############################
GoalAttribute: NextTC
Repeat_until: Finished

Case: Finished
Print: "Finished"

Case: Selected
Print: "------- Testcase: %s --------------"       testcase.number


GoalAttribute: FirstCode
Repeat_until: Finished

Case: Finished
Print: "   "
>SQL:  "DELETE FROM found  "
<SQL:  "   "

Case: Selected
Print: "#REM# -- print nothing"

GoalAttribute: EvalCombo
Repeat_until: Finished

Case: Finished
Print: "#REM# -- print nothing"

Case: NotReported
Print: "#REM# -- print nothing"

Case: Reported
Print: "Diagnosis Code [%s] cannot be reported together with [%s]" input1.code
input2.code
>SQL:  "INSERT INTO found (code1, code2) "
<SQL:  "VALUES ('%s', '%s') "                                      input1.code
input2.code


Initial_database_setup: empty_table_found
Query:
    DELETE FROM found
End_Query
```

Testrun solution 1

```
SQLite_database: "Database/ClaimTC.db"
Proposition: 'next testcase number selected'
Proposition: 'next code 1 selected'
Proposition: 'next code 2 selected'
Proposition: 'combo present in codes'
Proposition: 'combo already reported'
Attribute: input1.code      Type: Text
Attribute: input2.code      Type: Text
Database_view: testcase
Database_view: input1
Database_view: input2
Database_view: code
Database_view: already_found_combos
GoalAttribute: NextTC
GoalAttribute: FirstCode
GoalAttribute: EvalCombo
Initial_database_setup: empty_table_found


------- Testcase: 1 --------------
Diagnosis Code [D47.02] cannot be reported together with [C94.32]

------- Testcase: 2 --------------
Diagnosis Code [E71.313] cannot be reported together with [E72.3]

------- Testcase: 3 --------------
Diagnosis Code [R29.891] cannot be reported together with [M43.6]
Diagnosis Code [R29.891] cannot be reported together with [F45.8]
Diagnosis Code [M43.6] cannot be reported together with [F45.8]

------- Testcase: 4 --------------
Diagnosis Code [D75.81] cannot be reported together with [C94.42]

------- Testcase: 5 --------------

------- Testcase: 6 --------------
Diagnosis Code [D47.02] cannot be reported together with [C94.32]
Diagnosis Code [E71.313] cannot be reported together with [E72.3]
Diagnosis Code [R29.891] cannot be reported together with [M43.6]
Diagnosis Code [R29.891] cannot be reported together with [F45.8]
Diagnosis Code [M43.6] cannot be reported together with [F45.8]
Diagnosis Code [D75.81] cannot be reported together with [C94.42]

------- Testcase: 7 --------------
Diagnosis Code [D47.02] cannot be reported together with [C94.32]

Finished
Time elapsed: 0:00:02.436839
```

A brief note on the order of conditions within decision tables.

In creating the design for solution 1, I initially created the following decision table:

```
Table 2:
If:                                          | 0| 1| 2| 3| 4|
'next code 2 selected'                       | Y| Y| Y| Y| N|
input1.code <> input2.code                   | Y| Y| Y| N| -|
'combo present in codes'                     | Y| Y| N| -| -|
'combo already reported'                     | Y| N| -| -| -|
Then:
EvalCombo is NotReported                     | X|  | X| X|  |
EvalCombo is Reported                        |  | X|  |  |  |
EvalCombo is Finished                        |  |  |  |  | X|
# .......
# Repeat until: Finished
```

That gave the expected results, but when I looked at this table again, it turned out to be less logical: if a combo has already been reported as forbidden once, you don't have to search the code table again first! So conditions 3 and 4 must be reversed:

```
Table 2:
If:                                          | 0| 1| 2| 3| 4|
'next code 2 selected'                       | Y| Y| Y| Y| N|
input1.code <> input2.code                   | Y| Y| Y| N| -|
'combo already reported'                     | Y| N| N| -| -|
'combo present in codes'                     | -| Y| N| -| -|
Then:
EvalCombo is NotReported                     | X|  | X| X|  |
EvalCombo is Reported                        |  | X|  |  |  |
EvalCombo is Finished                        |  |  |  |  | X|
# .......
# Repeat until: Finished
```

with same results and improved performance!

Brief introduction to solution 2.

I am not exaggerating when I state that there is much resemblance between my first solution and Jacob Feldman's second solution[1]. With which it is therefore time to look at his excellent first solution.

This solution is excellent because it eliminates as many as 2 conditions in the decision table given earlier:

```
Table 2:
If:                                          | 0| 1| 2| 3| 4|
'next code 2 selected'                       | Y| Y| Y| Y| N|
~~input1.code <> input2.code~~                | ~~Y~~| ~~Y~~| ~~Y~~| ~~N~~| ~~-~~|
~~'combo already reported'~~                 | ~~Y~~| ~~N~~| ~~N~~| ~~-~~| ~~-~~|
'combo present in codes'                     | -| Y| N| -| -|
Then:
EvalCombo is NotReported                     | X|  | X| X|  |
EvalCombo is Reported                        |  | X|  |  |  |
EvalCombo is Finished                        |  |  |  |  | X|
# .......
# Repeat until: Finished
```

This is made possible by starting the inner loop not from the beginning of the input codes again and again, but with the next code, following the code selected in the outer loop. In the java code can thus be recognized by: for (int j=i+1, ...).

Without much further explanation, I realized this in my tool in the following way:

```
Attribute: restricted_input2_index  Type: Integer
Equals: input1.auto_index + input2.auto_index

Database_view: input1
With_attributes:
code
Query:
SELECT distinct code
  FROM input
 WHERE tc = %s
 LIMIT 1 OFFSET %s
With_arguments: testcase.number, input1.auto_index

Database_view: input2
With_attributes:
code
Query:
SELECT distinct code
  FROM input
 WHERE tc = %s
 LIMIT 1 OFFSET %s
With_arguments:  testcase.number, restricted_input2_index
```

The attribute auto_index of a database view always starts with 0 and is incremented by 1 when data has been retrieved from the database for the view.

[1] See: https://openrules.wordpress.com/2022/06/22/decision-model-for-medical-claim-processing/

Implementation of the decision model in DT5GL (Solution 2):

```
SQLite_database: "Database/ClaimTC.db"


Table 0:
If:                                              | 0| 1|
'next testcase number selected'                  | Y| N|
Then:
NextTC is Selected                               | X|  |
NextTC is Finished                               |  | X|
# .......
# Repeat until: Finished

Proposition: 'next testcase number selected'
Obtain_instance_from_database_view: testcase


Table 1:
If:                                              | 0| 1|
'next code 1 selected'                           | Y| N|
Then:
FirstCode is Selected                            | X|  |
FirstCode is Finished                            |  | X|
# .......
# Repeat until: Finished

Proposition: 'next code 1 selected'
Obtain_instance_from_database_view: input1


Table 2:
If:                                              | 0| 1| 2|
'next code 2 selected'                           | Y| Y| N|
'combo present in codes'                         | Y| N| -|
Then:
EvalCombo is Reported                            | X|  |  |
EvalCombo is NotReported                         |  | X|  |
EvalCombo is Finished                            |  |  | X|
# .......
# Repeat until: Finished


Proposition: 'next code 2 selected'
Obtain_instance_from_database_view: input2

Proposition: 'combo present in codes'
Obtain_instance_from_database_view: code

Attribute: restricted_input2_index  Type: Integer
Equals: input1.auto_index + input2.auto_index
```

```
########################## Database views ############################

Database_view: testcase
With_attributes:
number
Query:
 SELECT distinct tc
  FROM input
 LIMIT 1 OFFSET %s
With_arguments: testcase.auto_index


Database_view: input1
With_attributes:
code
Query:
SELECT distinct code
  FROM input
 WHERE tc = %s
 LIMIT 1 OFFSET %s
With_arguments: testcase.number, input1.auto_index

Database_view: input2
With_attributes:
code
Query:
SELECT distinct code
  FROM input
 WHERE tc = %s
 LIMIT 1 OFFSET %s
With_arguments:  testcase.number, restricted_input2_index


Database_view: code
With_attributes:
column1, column2
Query:
SELECT column1, column2
  FROM icd10codes
 WHERE column1 = '%s' AND column2 = '%s'
       OR
       column1 = '%s' AND column2 = '%s'
 LIMIT 1
With_arguments: input1.code, input2.code, input2.code, input1.code
```

```
########################## GoalAttributes ############################
GoalAttribute: NextTC
Repeat_until: Finished

Case: Finished
Print: "Finished"

Case: Selected
Print: "------- Testcase: %s --------------"       testcase.number

GoalAttribute: FirstCode
Repeat_until: Finished

Case: Finished
Print: "   "

Case: Selected
Print: "#REM# -- print nothing"

GoalAttribute: EvalCombo
Repeat_until: Finished

Case: Finished
Print: "#REM# -- print nothing"

Case: NotReported
Print: "#REM# -- print nothing"

Case: Reported
Print: "Diagnosis Code [%s] cannot be reported together with [%s]" input1.code
input2.code
```

Testrun solution 2

```
SQLite_database: "Database/ClaimTC.db"
Proposition: 'next testcase number selected'
Proposition: 'next code 1 selected'
Proposition: 'next code 2 selected'
Proposition: 'combo present in codes'
Attribute: restricted_input2_index  Type: Integer
Database_view: testcase
Database_view: input1
Database_view: input2
Database_view: code
GoalAttribute: NextTC
GoalAttribute: FirstCode
GoalAttribute: EvalCombo


------- Testcase: 1 --------------
Diagnosis Code [D47.02] cannot be reported together with [C94.32]

------- Testcase: 2 --------------
Diagnosis Code [E71.313] cannot be reported together with [E72.3]

------- Testcase: 3 --------------
Diagnosis Code [R29.891] cannot be reported together with [M43.6]
Diagnosis Code [R29.891] cannot be reported together with [F45.8]
Diagnosis Code [M43.6] cannot be reported together with [F45.8]

------- Testcase: 4 --------------
Diagnosis Code [D75.81] cannot be reported together with [C94.42]

------- Testcase: 5 --------------

------- Testcase: 6 --------------
Diagnosis Code [D47.02] cannot be reported together with [C94.32]
Diagnosis Code [E71.313] cannot be reported together with [E72.3]
Diagnosis Code [R29.891] cannot be reported together with [M43.6]
Diagnosis Code [R29.891] cannot be reported together with [F45.8]
Diagnosis Code [M43.6] cannot be reported together with [F45.8]
Diagnosis Code [D75.81] cannot be reported together with [C94.42]

------- Testcase: 7 --------------
Diagnosis Code [D47.02] cannot be reported together with [C94.32]

Finished
Time elapsed: 0:00:01.262903
```