

Variable-Related Analyses

Process variable template

Variables bundled in groups of three perform better. It is better to use multiple groups of three than one giant group. It is even better if the *values* (Indices) from the view_instance can be employed.

```
SELECT Variablen1.1,Variablen1.3,Variablen1.3
FROM
(
SELECT var1.stringvalue AS "1", var2.stringvalue AS "2", var3.stringvalue AS
"3", inst.ID
FROM view_instance inst
LEFT JOIN view_variable var1 ON inst.ID = var1.instanceID AND var1."name" =
''
LEFT JOIN view_variable var2 ON inst.ID = var2.instanceID AND var2."name" =
''
LEFT JOIN view_variable var3 ON inst.ID = var3.instanceID AND var3."name" =
''
)[] AS Variablen1
INNER JOIN
(
SELECT var1.stringvalue AS "1", var2.stringvalue AS "2", var3.stringvalue AS
"3", inst.ID
FROM view_instance inst
LEFT JOIN view_variable var1 ON inst.ID = var1.instanceID AND var1."name" =
''
LEFT JOIN view_variable var2 ON inst.ID = var2.instanceID AND var2."name" =
''
LEFT JOIN view_variable var3 ON inst.ID = var3.instanceID AND var3."name" =
''
)[] AS Variablen2
ON Variablen1.ID = Variablen2.ID
```

Prozessinstanzen inklusive beliebig vielen Prozessvariablen auflisten

Listing Processinstances as well as any amount of processvariables

With the following statement a list with processinstances as well as a presentation of indexvariables and any processvariables can be created.

Notice: To presort the resulting list the whole SQL-Query has to be based as a INNER-SELECT and then sorted.

```
SELECT DISTINCT
pi."name",
/*
```

Selecting the indexvariables from 1 to 10, if desired:

```
*/
CASE WHEN pi.value1 IS NOT NULL THEN pi.value1 ELSE '-' END AS "Index 1",
CASE WHEN pi.value2 IS NOT NULL THEN pi.value2 ELSE '-' END AS "Index 2",
CASE WHEN pi.value3 IS NOT NULL THEN pi.value3 ELSE '-' END AS "Index 3",
/*
```

Selecting the required processvariables:

- A processVariables.- Line has to be added for every variable

```
*/
processVariables.VARIABLE_1_WITHOUT_SPACE AS "Header Variable 1",
processVariables.VARIABLE_2_WITHOUT_SPACE AS "Header Variable 2",
processVariables.VARIABLE_3_WITHOUT_SPACE AS "Header Variable 3",
identP.name AS "StarterID",
pi.definitionName AS "Prozessdefinition",
pi.id AS "ProzessID"
```

FROM view_instance pi

LEFT JOIN view_identity identP ON pi.creationUserId = identP.id

LEFT JOIN (

SELECT piInner.id,

/*

Creating a MAX(...) line for every desired variable

*/

MAX(CASE WHEN var.name = 'PROZESSVARIABLE 1' THEN

var.stringValue ELSE '-' END) AS VARIABLE_1_WITHOUT_SPACE,

MAX(CASE WHEN var.name = 'PROZESSVARIABLE 2' THEN

var.stringValue ELSE '-' END) AS VARIABLE_2_WITHOUT_SPACE,

MAX(CASE WHEN var.name = 'PROZESSVARIABLE 3' THEN

var.stringValue ELSE '-' END) AS VARIABLE_3_WITHOUT_SPACE

FROM view_instance piInner

/*

In the following IN(...) statement all variablenames have to be listed, that are to be selected:

*/

LEFT JOIN view_variable var ON var.instanceId = piInner.id AND

var.name IN ('PROZESSVARIABLE 1', 'PROZESSVARIABLE 2', 'PROZESSVARIABLE 3')

GROUP BY piInner.id

) processVariables ON pi.id = processVariables.id

From:
<https://wiki.tim-solutions.de/> - **TIM Wiki** / [NEW TIM 6 Documentation](#)

Permanent link:
https://wiki.tim-solutions.de/doku.php?id=en:software:dashboard:analyses:variable_analyses_oracle

Last update: **2021/07/01 09:52**

