**Wikiprint Book**

**Title: Przykłady workflow**

**Subject: eDokumenty - elektroniczny system obiegu dokumentów, workflow i CRM - DeployerGuide/Customization/ProcessAutomation/Examples**

**Version: 68**

**Date: 10/08/25 04:03:38**

## Table of Contents

*Przykłady workflow* 3
*WARUNKI* 3
*PRZYPISANIA* 5
*KWERENDY DO PARAMETRU SELECT[]* 6
*DYNAMICZNE WARTOŚCI PARAMETRÓW KOMEND* 7
*Zapytania do parametrów Akcji* 7
*Różne* 8
*Konstrukcje z ANY* 9
*Zapytanie dla "moich zadań workflow"* 9
*Zapytanie dla moich zadań workflow (v>5)* 9
*Zapytanie dla wszystkich zadań workflow z modułu Dokumenty:* 12
*Linki przydatne* 12

## Przykłady workflow

W tym miejscu publikowane będą przykładowe kwerendy użyteczne przy budowaniu warunków, przekazywaniu parametrów do komend itp.

### WARUNKI

```
-- W1. Sprawdzenie czy jest plik
-- Brak
SELECT NOT EXISTS(
 SELECT doc_id FROM attachments WHERE doc_id =
(SELECT doc_id FROM documents WHERE procid = {PROCID}))

-- Jest
SELECT EXISTS(
 SELECT doc_id FROM attachments WHERE doc_id =
(SELECT doc_id FROM documents WHERE procid = {PROCID}))

-- W2. Sprawdzenie czy są zamieszczone opinie kierowników
-- pobierane jako parametry z bpm_properties.id____ = 1
-- brak
SELECT count(*) = 0 FROM bpm_property_values
WHERE id____ IN (12,13,14)
AND procid = {PROCID}
AND value_  != ''

-- są
SELECT count(*) > 0 FROM bpm_property_values
WHERE id____ IN (12,13,14)
AND procid = {PROCID}
AND value_  != ''

W3.
-- czy dokument JEST dołączony do sprawy i powiązany z innym dokumentem (zakładka Powiązania)
SELECT dlp.prc_id IS NOT NULL AND dld.doc_id IS NOT NULL
FROM documents_view dv
LEFT JOIN doc_link_proc dlp USING(doc_id)
LEFT JOIN doc_link_doc dld ON dv.doc_id = dld.doc_id
WHERE dv.doc_id = {DOC_ID} LIMIT 1

W4.
-- Sprawdzenie czy dokument jest powiązany ze sprawą i dokumentem (zakładka powiązania)
SELECT dlp.prc_id IS NOT NULL AND dld.doc_id IS NOT NULL
FROM documents_view dv
LEFT JOIN doc_link_proc dlp USING(doc_id)
LEFT JOIN doc_link_doc dld ON dv.doc_id = dld.rel_to
WHERE dv.doc_id = {DOC_ID}

W5.
-- Sprawdzenie czy dokument ma wypełnione pozycje (np. zapotrzebowanie, faktura)
SELECT EXISTS (SELECT fk.adddat FROM fk_elements fk INNER JOIN documents d USING (doc_id)
WHERE d.doc_id = 50856 AND fk.is_del IS FALSE)

W6.
-- Sprawdzenie czy w sprawie zamknięte są wszystkie zapytania ofertowe (kontrolowane procedurą o prtpid = 5
SELECT sum(res) = 0 FROM
(SELECT
(CASE WHEN s.ptsttp = 'END' THEN 0 ELSE 1 END) AS res, d.dscrpt, s.ptstnm, d.prtpid
FROM documents d
INNER JOIN stages s ON d.is_del IS NOT TRUE AND s.procid = d.procid AND ((s.is_act IS TRUE AND s.is_fix IS FALSE) OR (s.pt
WHERE d.prtpid = 5 AND d.prc_id = {PRC_ID}) AS x
```

```
W7.
-- Sprawdzenie czy osoba zalogowana jest z określonego działu (wg orunid np. 57)
select substring(get_org_path(orunid), '.57.') IS NOT NULL FROM orgtree_view WHERE usr_id = {LOGGED_USR_ID}

W8.
-- Sprawdzenie wartości faktury z rozpisanymi kosztami
SELECT sum(amount) = (SELECT COALESCE(sum(netto_),0) FROM vatnote WHERE doc_id = {DOC_ID}) FROM vatnote_costs WHERE doc_id

W9.
-- Sprawdzenie typu dokumentu księgowego - czy jest konkretny - 6
SELECT EXISTS(SELECT 1 FROM types_of_accountants_doc INNER JOIN vatnote USING (accdid) WHERE doc_id = {DOC_ID} AND accdid

W10.
-- Sprawdzenie czy cechy są ustawione w sprawie (jeśli są to TRUE) - dla warunku przeciwnego należy zamienić "IS NOT NULL
SELECT ftv1.data__ IS NOT NULL
AND ftv2.data__ IS NOT NULL
AND ftv3.data__ IS NOT NULL
AND ftv4.data__ IS NOT NULL
FROM processes p LEFT JOIN features_text_view ftv1 ON ftv1.tbl_id = p.prc_id AND ftv1.featid = 29
LEFT JOIN features_text_view ftv2 ON ftv2.tbl_id = p.prc_id AND ftv2.featid = 30
LEFT JOIN features_text_view ftv3 ON ftv3.tbl_id = p.prc_id AND ftv3.featid = 31
       LEFT JOIN features_text_view ftv4 ON ftv4.tbl_id = p.prc_id AND ftv4.featid = 32
WHERE p.prc_id = {PRC_ID}

W11
-- Sprawdzenie globalnej wartości własności procedury (z uwzględnieniem podprocesów).
Jeżeli choć jedna osoba w podprocesie nie wyraziła zgody (przeszło przez przypisanie do FALSE) to zwraca FALSE
SELECT count(*) > 0 AS res FROM (
SELECT CASE WHEN (value_::boolean = TRUE) THEN 1 ELSE 0 END AS result
FROM bpm_property_values bpv WHERE bpv.id____ = 88 AND bpv.procid IN
(SELECT p.procid
FROM documents d
INNER JOIN procedures p ON p.procid = d.procid OR d.procid = p.rootpr
WHERE d.doc_id = {DOC_ID})) x
WHERE result > 0;

W12.
-- Sprawdzenie czy zaznaczona jest cecha
-- typu Pole zaznaczane (jeśli 1 to zaznaczona)
SELECT COALESCE(f32.data__::int, 0) = 1
FROM documents d
LEFT JOIN features_text_view f32 ON d.doc_id = f32.tbl_id AND f32.featid = 255
WHERE d.doc_id = 541950


W13.
-- sprawdzenie czy zalogowana osoba nie ma ustawionego zastępstwa
SELECT EXISTS(SELECT * FROM global_sys_conf WHERE (SELECT regexp_matches(objnam,  '_(\d+)$'))[1]::int = {LOGGED_USR_ID})
```

W14. -- sprawdzenie czy są uzupełnione cechy na kliencie, do którego jest kierowana oferta SELECT x.potrzeba IS NOT NULL AND x.segment IS NOT NULL AND x.zrodlo IS NOT NULL AND x.branza IS NOT NULL FROM (

SELECT (SELECT ftopnm FROM features_opt_view fop WHERE fop.featid = 11 AND fop.tbl_id = fod.contid) AS segment, (SELECT ftopnm FROM features_opt_view fop WHERE fop.featid = 17 AND fop.tbl_id = fod.contid) AS zrodlo, (SELECT ftopnm FROM features_opt_view fop WHERE fop.featid = 13 AND fop.tbl_id = fod.contid) AS branza, (SELECT data FROM features_text_view ftv WHERE ftv.featid = 26 AND ftv.tbl_id = fod.contid) AS potrzeba FROM documents d INNER JOIN fk_offer_documents fod USING(doc_id) INNER JOIN contacts c1 ON c1.contid = fod.contid WHERE d.doc_id = {DOC_ID}) x

W15. -- sprawdzenie czy są uzupełnione cechy na fakturze do którego nabywcy (lub odbiorcy jeśli jest uzupełniony) , albo klienta końcowego oznaczonego na cesze faktury jako klient z bazy - jest kierowana faktura - są uzupełnione cechy typ. segment, źródło, branża SELECT x.typ IS NOT NULL AND x.segment IS NOT NULL AND x.zrodlo IS NOT NULL AND x.branza IS NOT NULL FROM ( SELECT

(SELECT text_sum(ftopnm) FROM features_opt_view fop WHERE fop.featid = 12 AND fop.ftopid IN (7,9, 28) AND fop.tbl_id = (COALESCE(ftv.data::int, c1.contid))) AS typ, (SELECT ftopnm FROM features_opt_view fop WHERE fop.featid = 11 AND fop.tbl_id = (COALESCE(ftv.data::int, c1.contid))) AS segment, (SELECT ftopnm FROM features_opt_view fop WHERE fop.featid = 17 AND fop.tbl_id = (COALESCE(ftv.data::int, c1.contid))) AS zrodlo, (SELECT ftopnm FROM features_opt_view fop WHERE fop.featid = 13 AND fop.tbl_id = (COALESCE(ftv.data::int, c1.contid))) AS branza FROM documents d INNER JOIN vatnote v USING(doc_id) INNER JOIN contacts c1 ON c1.contid = COALESCE(v.rectid, v.toctid) LEFT JOIN features_text_view ftv ON ftv.tbl_id = d.doc_id AND ftv.featid = 15 -- klient końcowy LEFT JOIN contacts c2 ON c2.contid = ftv.data::int WHERE d.doc_id = {DOC_ID}) x

### PRZYPISANIA

```
-- P1. Przypisanie jako osoby tworzącej dokument
SELECT ARRAY[o.orunid] FROM orgtree_view o INNER JOIN documents d ON d.adduid = o.usr_id
WHERE d.procid = {PROCID}

--
-- P2. Przypisanie akceptanta (pobierany z właściwości)
SELECT ARRAY[orunid] FROM organization_units WHERE orunid = {procedures.AKCEPTANT CZŁONEK ZARZĄDU}

--
-- P3. Przypisanie osób które zaakceptowały określony etap (np. 44)
SELECT ARRAY[orunid] FROM stages WHERE ptstid = 44 AND procid = {PROCID}


--
-- P4. Przypisanie osoby odpowiedzialnej ze sprawy
SELECT ARRAY(SELECT o.orunid FROM processes p INNER JOIN orgtree_view o ON p.rspuid = o.usr_id WHERE prc_id = {PRC_ID})

--
-- P5. Przypisanie osób z parametru typu usr_ids[]
SELECT ARRAY(SELECT o.orunid FROM orgtree_view o WHERE o.usr_id IN ({procedures.OSOBY}))

--
-- P6. Przypisanie osoby odpowiedzialnej za MPK wpisane w fakturze (do etapu MULTI)
SELECT ARRAY (
 SELECT CASE
   WHEN x.ndetpe = 'POST' THEN x.orunid
   WHEN x.ndetpe = 'ORGCELL' THEN (SELECT o2.orunid FROM orgtree_view o2 WHERE o2.prn_id = x.orunid LIMIT 1) END
 FROM
 (SELECT DISTINCT mpk.orunid, o.ndetpe FROM vatnote_costs
  LEFT JOIN places_of_vcosts AS mpk USING (povcid)
  LEFT JOIN orgtree_view AS o ON mpk.orunid = o.orunid
  WHERE doc_id = {DOC_ID})
 AS x)


--
-- P7. Przypisanie osoby zalogowanej
SELECT ARRAY(SELECT o.orunid FROM orgtree_view o WHERE o.usr_id = {LOGGED_USR_ID})

--
-- P8. Przypisanie osoby wybranej w liście wyboru (lista typu select z wartościami orunid, przypisana do zmiennej typu Int
SELECT array[orunid] FROM organization_units WHERE orunid = {$DYREKTOR_HANDLOWY}

--
-- P9. Przypisanie orunid w zależności od accdid (jednostki org. w zależności od typu dokumentu księgowego)
SELECT
CASE WHEN v.accdid = 1 THEN [62]
```

```
       WHEN v.accdid = 2 THEN 56
       WHEN v.accdid = 3 THEN 61
       WHEN v.accdid = 4 THEN 60
       WHEN v.accdid = 5 THEN 63
END
FROM vatnote v  WHERE v.doc_id = {DOC_ID}

-- P10. Przypisanie osoby, która załatwiła poprzedni etap
SELECT ARRAY[s.orunid]
FROM stages s
LEFT JOIN workflow_log wl USING(sop_id)
WHERE s.is_fix IS TRUE AND s.procid = {procedures.procid} AND wl.chloid = (SELECT max(chloid) FROM workflow_log WHERE proc

-- P11
-- Przypisanie osób do etapu z tablicy VAR_OSOBY_OPISUJACE, które nie są w tablicy VAR_OSOBY_FIXED.

SELECT array(
SELECT orunid FROM orgtree_view
WHERE orunid IN (
SELECT * FROM bs_unnest({$VAR_OSOBY_OPISUJACE}::int[]::int[]) as qq where not (array[qq] <@ {$VAR_OSOBY_FIXED}::int[])
   )
)

-- P12
-- Przypisanie osób z określonej grupy (grp_id = 9)
SELECT ARRAY(SELECT o.orunid FROM orgtree_view o WHERE o.orunid IN (SELECT o.orunid FROM users_link_group ulg INNER JOIN o

--
-- P13
-- Przypisanie wartości cechy - pracownik

SELECT ARRAY[o.orunid]
FROM documents d
LEFT JOIN features_text_view f26 ON d.doc_id = f26.tbl_id AND f26.featid = 26
LEFT JOIN orgtree_view o ON o.usr_id = f26.data__::int
WHERE d.doc_id = {DOC_ID}

-- P14
-- PRZYPISANIE przełożonego osoby wybranej w parametrze $WNIOSKODAWCA
SELECT (get_superior( (SELECT orunid FROM orgtree_view WHERE orunid = ANY({$WNIOSKODAWCA}))))

-- lub dla opcji kiedy własność jest typu orunid[]
SELECT ARRAY(SELECT get_superior( (SELECT orunid FROM orgtree_view WHERE orunid = ANY({$WNIOSKODAWCA}::int[]))))


-- P15
-- Przypisanie do własności typu orunid[]
SELECT {$KTO_OSOBA}::int[]
```

### KWERENDY DO PARAMETRU SELECT[]

```
-- Członkowie zarządu
SELECT orunid as value, fullnm || ' - ' || ndenam as caption FROM orgtree_view WHERE orunid IN (3,14,15,16)

-- Członkowie grupy o id 9
SELECT o.orunid AS value, o.fullnm AS caption FROM users_link_group ulg INNER JOIN orgtree_view o ON o.usr_id = ulg.usr_id

--
-- Wszyscy pracownicy (lista jednokrotnego wyboru)
SELECT orunid as value, lasfir || ' - ' || ndenam as caption FROM orgtree_view WHERE ndetpe = 'POST' AND is_del IS FALSE A
```

## DYNAMICZNE WARTOŚCI PARAMETRÓW KOMEND

```
-- Utwórz komentarz
dscrpt="SQL::SELECT CASE WHEN (SELECT EXISTS(
  SELECT value_
  FROM bpm_property_values WHERE id____ = 20 AND sop_id = {SOP_ID})) THEN
(SELECT value_
  FROM bpm_property_values WHERE id____ = 20 AND sop_id = {SOP_ID})
ELSE 'Bez uwag' END AS result
FROM stages WHERE sop_id = {SOP_ID}"

-- Utwórz przypomnienie w sprawie windykacyjnej
-- dla handlowca
usr_id="SQL::SELECT seller FROM vindication.vind_proc_view WHERE prc_id = {PRC_ID}",
start_="SQL::SELECT fxterm - interval '3 days' FROM vindication.vind_proc_view WHERE prc_id = {PRC_ID}",dscrpt="Uwaga! Za
```

### Zapytania do parametrów Akcji

Tworzenie dokumentu:

```
Tworzenie wydania z dokumentu przyjęcia
dctpid="17",dscrpt="Wydanie zewnętrzne",do="SQL::SELECT orunid FROM orgtree_view WHERE usr_id = {LOGGED_USER}"

# tworzenie zamówienia z zapotrzebowania
dctpid="41", dscrpt="Zamówienie do:{spller}",map="adddat=crtdat,dlvdat=orddat,acorid=acorid,spadid=spadid,pchaid=pchaid",s

#przepisz pozycje z zapotrzebowania do zamówienia
from__="SQL::SELECT doc_id FROM documents WHERE doc_id={DOC_ID}", to____="SQL::SELECT doc_id FROM documents WHERE rel_to={

# przepisz pozycje z zapotrzebowania do istniejącego zamówienia
from__="SQL::SELECT doc_id FROM documents WHERE doc_id={DOC_ID}", to____="{procedures.ZAMÓWIENIE}"

# czy pozycje uzupełnione
query="SELECT (SUM(CASE WHEN depnam = depsym THEN 1 ELSE 0 END))=0 FROM fk_elements WHERE doc_id={doc_id} AND is_del=FALSE

# Utwórz sprawę (dsexid teczki, opis z opisu dokumentu, kontrahent z nadawcy, procedura o id 2
dsexid="107",dscrpt="{documents.dscrpt}",contid="SQL::SELECT contid FROM doc_link_cont WHERE doc_id = {DOC_ID} AND role__

# Zarejestruj dokument w zalezności od tego czy dokument przyszedł z emaila
reg_id="
SELECT
CASE WHEN (SELECT EXISTS (SELECT e.doc_id FROM emails e WHERE e.doc_id = d.doc_id)) THEN 6
WHEN (SELECT NOT EXISTS (SELECT e.doc_id FROM emails e WHERE e.doc_id = d.doc_id)) THEN 4
END
FROM documents d WHERE d.doc_id = {DOC_ID}"


# Sprawdź czy workflow utworzonego z procedury dokumentu (o typie dctpid 10) został zakończony
query="SELECT EXISTS(
SELECT doc_id FROM documents d INNER JOIN stages s ON s.procid = d.procid AND s.ptsttp = 'END'
AND d.doc_id = (SELECT dld.doc_id FROM documents d2 INNER JOIN doc_link_doc dld ON d2.doc_id = dld.doc_id WHERE d2.doc_id

# Sprawdź czy wypełnione są na opisie kosztowym konta grupy 4XX
SELECT EXISTS(SELECT substring(type__,1,1) = '4' AS res
FROM vatnote_costs INNER JOIN types_of_vcosts USING (tovcid)
WHERE doc_id = {DOC_ID} GROUP BY res HAVING substring(type__,1,1) = '4' )
```

```
# Wyślij email (do parametry Do - ustawienie emaili osób przypisanych do etapu)
SELECT text_sum(o2.e_mail) FROM
stages s3 INNER JOIN orgtree_view o2 ON o2.orunid = ANY(s3.orgarr)
WHERE s3.sop_id = {SOP_ID}

# wybierz użytkowników uprawnionych w sprawie nie będących opiekunem
SELECT text_sum(usr_id::text) FROM proc_link_users WHERE prc_id = {processes.prc_id} AND type__ = 'USER' AND usr_id != {pr

# Ustaw przypomnienie na termin obowiązywania umowy - podana w parametrze l. miesięcy wypowiedzienia - 1

SELECT COALESCE(enddat, strdat + interval '1 year') - (fov.ftopnm::int + 1 || 'month')::interval AS termin
FROM documents d
INNER JOIN contract USING(doc_id)
LEFT JOIN features_opt_view fov ON fov.tbl_id = d.doc_id AND fov.featid = 13
WHERE dctpid = 3 AND prc_id = {PRC_ID}

-- Uprawnij pracownika z parametru do sprawy
SQL::SELECT text_sum(usr_id::text) FROM orgtree_view WHERE orunid = ANY({$OSOBA_ZAINTERESOWANA_SPRAWA})

-- Ustawienie wartości ftopid cechy w komendzie ustaw cechę
-- na podstawie orunid pobranego z danych wejściowych
-- W tym przykładzie Cecha zawiera listę wartości zgodnych z inicjałami
SELECT ftopid
FROM
features_options
INNER JOIN features_options_def USING(fodfid)
WHERE featid = 91
AND ftopnm = (SELECT initls FROM users join users_link_org_units using (usr_id) join organization_units using(orunid) WHER
LIMIT 1

-- Ustawienie w komendzie Wyślij powiadomienie Pracowników wybranych z tablicy
-- użytkowników w paramatrze w danych wejsciowych (typ tablicowy orunid[])
SELECT text_sum(usr_id::text) FROM orgtree_view WHERE orunid = ANY ('{$OSOBY_UPRAWNIONE}'::int[])

-- jeśli typ tablicowy to nie używamy cudzysłowia
SQL::SELECT text_sum(usr_id::text) FROM orgtree_view WHERE orunid = ANY ({$WNIOSKODAWCA}::int[])

w wykonaniu wygląda to tak:
SELECT text_sum(usr_id::text) FROM orgtree_view WHERE orunid = ANY ('{14,15}'::int[])

-- Ustawienie wartości cechy
-- jeśli z wartości tekstowej chcemy usunać cudzysłowia
SQL::SELECT replace({$KOSZTY_NAPRAWY}, '''','')
```

### Różne

```
-- Okreslenie stanowiska (orunid) na podstawie wartości cechy produktu w sprawie - listy wyboru
SELECT CASE WHEN fop.ftopid = 206 THEN ARRAY[79]
WHEN fop.ftopid = 207 THEN ARRAY[95]
WHEN fop.ftopid = 208 THEN ARRAY[100]
WHEN fop.ftopid = 209 THEN ARRAY [103]
WHEN fop.ftopid = 210 THEN ARRAY [70]
WHEN fop.ftopid = 211 THEN ARRAY [51]
END
FROM fk_elements_view fk
INNER JOIN features_opt_view fop ON fop.tbl_id = fk.fkelid
WHERE fop.tblnam = 'fk_elements' AND fk.doc_id = 14949
LIMIT 1
```

```
-- Sprawdzenie czy cecha jest wypełniona
SELECT NOT EXISTS(
SELECT CASE WHEN fop.ftopid IN (206, 207, 208, 209, 210, 211) THEN TRUE
ELSE FALSE END
FROM fk_elements_view fk
LEFT JOIN features_opt_view fop ON fop.tbl_id = fk.fkelid
WHERE fop.tblnam = 'fk_elements' AND fk.doc_id = 15815) AS r

-- Sprawdzenie czy wybrany został termin dostawy w zapotrzebowaniu
SELECT dlvdat IS NOT NULL FROM demand WHERE doc_id = 14949

-- Ustawienie nazwy dla podsprawy zakładanej komendą Utwórz sprawę
SELECT 'Zlecenie realizacji :' || dscrpt FROM processes WHERE prc_id = 10

-- sprawdzenie czy jest uzupełniona cecha w sprawie do której należy aktualny dokument {DOC_ID}
SELECT NOT EXISTS(SELECT CAST (bpv.value_ AS int)
FROM documents doc
INNER JOIN processes pr USING (prc_id)
INNER JOIN bpm_property_values bpv ON (bpv.procid = pr.procid AND bpv.id____ = 32)
WHERE doc.doc_id = 14949)

-- Wybór adresu - domyślnego kontaktu
SELECT mainad FROM contacts_view WHERE contid = 2613

-- Sprawdzenie czy termin wprowadzony w cesze nie jest zbyt krótki
SELECT (f32.data__::date - interval '3 days') >= CURRENT_DATE
FROM documents d
LEFT JOIN features_text_view f32 ON d.doc_id = f32.tbl_id AND f32.featid = 263
WHERE d.doc_id = {DOC_ID}

-- Sprawdzenie czy zaznaczona jest określona OPCJA cechy
SELECT f32.ftopid = 230
FROM documents d
LEFT JOIN features_opt_view f32 ON d.doc_id = f32.tbl_id AND f32.featid = 263
WHERE d.doc_id = {DOC_ID}
```

### Konstrukcje z ANY

```
-- Sprawdzenie czy wybrany pracownik w parametrze jest 12
select orunid = ANY('{12}') from orgtree_view

-- Pobranie orunid z danej wejściowej typu tablicowego
select orunid from orgtree_view where orunid = ANY('{12}') limit 1

-- Pobranie listy pracowników usr_id na podstawie zawartosci tablicy orunid[]
SELECT text_sum(usr_id::text) FROM orgtree_view WHERE orunid = ANY({$OSOBA_ZAINTERESOWANA_SPRAWA})
```

### Zapytanie dla "moich zadań workflow"

#### Zapytanie dla moich zadań workflow (v>5)

```
WITH RECURSIVE user_all_replacements (who___, bywhom, path)
AS (
   SELECT r.who___,
       r.bywhom,
       array [r.who___]::INT [] AS path
   FROM replacements r
   WHERE r.bywhom IN ({ORUNID})
       AND (now() BETWEEN r.from__ AND r.to____)
```

```
        AND NOT r.suspen

    UNION ALL

    SELECT r.who___,
        r.bywhom,
        uar.path || r.who___ AS path
    FROM replacements r,
        user_all_replacements uar
    WHERE NOT (r.who___ = ANY (path))
        AND r.bywhom = uar.who___
        AND (now() BETWEEN r.from__ AND r.to____)
        AND NOT r.suspen
),
all_orunids as (
        SELECT array_agg(who___)||{ORUNID} as val
        FROM (
                SELECT DISTINCT r.who___ FROM user_all_replacements r
        ) foo
)
SELECT keyval, dscrpt, clsnam, ptstnm, dctpid, prtpnm, end___, ptstid, prior_, prtpid
FROM (
    SELECT keyval, dscrpt, clsnam, ptstnm, dctpid, prtpnm, end___, ptstid, aa.prior_, pd.prtpid, orgarr IS NOT NULL as orga
    FROM procedures_def pd
    RIGHT JOIN (
        SELECT (doc_id) AS keyval,
            p2.prtpid,
            substr(d.dscrpt, 0, 100) AS dscrpt,
            'DOCUMENT' AS clsnam,
            end___,
            s.ptstnm,
            d.dctpid,
            ptstid,
            orgarr,
            s.actdat,
            d.prior_
        FROM procedures pd
        LEFT JOIN stages s USING (PROCID)
        LEFT JOIN procedures p2 ON (p2.PROCID = pd.rootpr)
        LEFT JOIN bpm_loops_def bld USING (ptstid)
        RIGHT JOIN documents d ON (d.PROCID = p2.PROCID)
        WHERE (
                bld.multii IS NOT TRUE
                OR s.prn_id IS NOT NULL
                )
            AND (
                ((SELECT val FROM all_orunids) && s.orgarr)
                OR (
                    s.orgarr IS NULL
                    AND ARRAY[d.target] <@ (SELECT val FROM all_orunids)
                    )
                )
            AND NOT d.is_del
            AND gostof IS NULL
            AND sop_id IS NOT NULL
            AND s.is_act
            AND NOT (p2.comple OR p2.cancel)
            AND ptsttp NOT IN ('SUBPROCESS', 'START', 'END')

        UNION
```

```
SELECT (prc_id) AS keyval,
    p2.prtpid,
    substr(p.dscrpt, 0, 100) AS dscrpt,
    'PROCESS' AS clsnam,
    end___,
    s.ptstnm,
    0 AS dctpid,
    ptstid,
    orgarr,
    s.actdat,
    NULL AS prior_
FROM procedures pd
LEFT JOIN stages s USING (PROCID)
LEFT JOIN procedures p2 ON (p2.PROCID = pd.rootpr)
LEFT JOIN bpm_loops_def bld USING (ptstid)
RIGHT JOIN processes p ON (p.PROCID = p2.PROCID)
WHERE (
        bld.multii IS NOT TRUE
        OR s.prn_id IS NOT NULL
        )
    AND p.is_fix IS FALSE
    AND p.is_del IS FALSE
    AND ((SELECT val FROM all_orunids) && s.orgarr)
    AND sop_id IS NOT NULL
    AND s.is_act IS TRUE
    AND NOT (p2.comple OR p2.cancel)
    AND ptsttp NOT IN ('SUBPROCESS', 'START', 'END')

UNION

SELECT (rcp_id) AS keyval,
    p2.prtpid,
    substr(p.dscrpt, 0, 100) AS dscrpt,
    'RCP' AS clsnam,
    end___,
    s.ptstnm,
    0 AS dctpid,
    ptstid,
    orgarr,
    s.actdat,
    NULL AS prior_
FROM procedures pd
LEFT JOIN stages s USING (PROCID)
LEFT JOIN procedures p2 ON (p2.PROCID = pd.rootpr)
LEFT JOIN bpm_loops_def bld USING (ptstid)
RIGHT JOIN rcp_cards p ON (p.PROCID = p2.PROCID)
WHERE (
        bld.multii IS NOT TRUE
        OR s.prn_id IS NOT NULL
        )
    AND p.is_fix IS FALSE
    AND p.is_del IS FALSE
    AND (
        ((SELECT val FROM all_orunids) && s.orgarr)
        OR (
            s.orgarr IS NULL
            AND p.emp_id = {USR_ID}
            )
        )
    AND sop_id IS NOT NULL
    AND s.is_act IS TRUE
```

```
            AND NOT (p2.comple OR p2.cancel)
            AND ptsttp NOT IN ('SUBPROCESS', 'START', 'END')
    ) AS aa USING (prtpid)
) res
GROUP BY prtpid, keyval, dscrpt, clsnam, ptstnm, dctpid, prtpnm, end___, ptstid, prior_, orgarr, actdat
ORDER BY prtpid, ptstid, (end___ IS NOT NULL AND orgarr) DESC, end___ IS NOT NULL DESC, end___ ASC, actdat ASC, orgarr DES
```

**Zapytanie dla wszystkich zadań workflow z modułu Dokumenty:**

```
SELECT doc.*, reg.regtyp,reg.ndenam, type.dctptp,type.dctpnm,type.dctpic,
    (SELECT ARRAY[count(f.fileid),min(f.fileid)] FROM attachments LEFT JOIN files f USING(fileid) WHERE is_del IS NOT TRUE
    (EXISTS (SELECT 1 FROM documents_history WHERE doc_id=doc.doc_id AND orunid = 1 AND strpos(dscrpt, 'Otwarcie')=1)) as i
FROM documents doc
INNER JOIN (
    SELECT doc_id, (array_agg(attrib))[1] as attrib
    FROM (
        SELECT documents_view.doc_id, (CASE WHEN dlu_doc_id IS NULL THEN NULL ELSE coalesce(dlu.attrib, '') END) as attrib
        FROM documents documents_view
        LEFT JOIN storage_places stp using(strpid)
        LEFT JOIN (
            SELECT dlu.doc_id as dlu_doc_id, dlu.attrib, dlu.usr_id, ul.prior_
            FROM doc_link_users dlu
            LEFT JOIN users_link_group ul on(ul.grp_id = dlu.grp_id AND ul.usr_id = 2)
            WHERE dlu.usr_id = 2 OR ul.usr_id = 2
        ) dlu ON (dlu_doc_id = documents_view.doc_id)
        WHERE is_del IS NOT TRUE AND gostof IS NULL
            AND procid IN (
                SELECT p.rootpr
                FROM procedures p
                LEFT JOIN stages s USING(procid)
                WHERE (ARRAY(SELECT orunid FROM orgtree_view LEFT JOIN users_link_org_units ulo USING(orunid) WHERE orunid
            )
            AND is_iso IS FALSE AND partof IS NULL ORDER BY dlu_doc_id, dlu.usr_id IS NOT NULL DESC, dlu.prior_ ASC
    ) foo2
    GROUP BY doc_id
) dolu USING (doc_id)
LEFT JOIN types_of_documents type USING(dctpid)
LEFT JOIN registers reg USING(reg_id)
WHERE (substr(dolu.attrib, 1, 1) = 'r') OR (dolu.attrib IS NULL AND (prionl IS NOT TRUE OR doc.adduid = 2))
ORDER BY timest DESC,doc_id DESC
```

## Linki przydatne

Funkcje tablicowe Postgres: http://www.postgresql.org/docs/8.4/static/intarray.html