

Fixing JIRA graph gadgets broken by JRA-59364

JIRA 7 bug  [JRASERVER-59364](#) - Created vs. Resolved gadget should respect the Cumulative Totals option set before the upgrade CLOSED

causes most 'Created vs. Resolved Chart' gadgets on dashboards to break with the error:

Invalid field. Valid values are [count, cumulative].

Created vs. Resolved Chart



Unfortunately, one or more of your preferences are now unavailable. Please update your preferences, or remove gadget by clicking delete from the title bar above.

- Invalid field. Valid values are [count, cumulative].

Broken portlets can be fixed individually by editing them and setting `Collection Operation` to either `Cumulative` (emulating the old default behaviour) or `Count`:

Created vs. Resolved Chart

Project or Saved*

project-11353

Filter:

Search

Project or saved filter to use as the basis for the graph.

[Advanced Search](#)

Period:*

Daily

The length of periods represented on the graph.

Days Previously*

30

Days in the past to collect data for the selected period.

Collection Operation*

Count

Count

Cumulative

individual values

Display the Trend of Unresolved*

No

Show the number of unresolved issues over time in a subplot.

Display Versions*

Only major versions

Show when versions were released on the chart.

Auto refresh

☒ Update every 15 minutes

Save

Cancel

However only the dashboard owner can do this, and it's not exactly intuitive. This page describes a SQL fix to migrate all Created vs Resolved portlets to the new preference format.

Table of Contents

- [I haven't yet upgraded. Will I be affected?](#)
- [How do I fix it?](#)
- [What's going on?](#)
- [Deriving the SQL Fix](#)

- Background: database tables involved in JIRA portlets
- SQL part 1: collecting relevant old-portlet data
 - Recreating the missing parameters
- SQL part 2: inserting new rows
- Inserting data: dealing with IDs
 - Updating the sequence_value_item table
- Shameless Plug

I haven't yet upgraded. Will I be affected?

If  **JRASERVER-59364** - Created vs. Resolved gadget should respect the Cumulative Totals option set before the upgrade CLOSED isn't marked resolved, then probably, yes. Find out which dashboards will be affected by running the SQL:

```
jira=> SELECT DISTINCT portalpage.pagename AS "Dashboard",
        concat('http://jira.example.com/secure/Dashboard.jspa?selectPageId=', portalpage.id) AS URL,
        app_user.lower_user_name AS "Owner"
FROM portalpage JOIN portletconfiguration ON portalpage.id=portletconfiguration.portalpage
        JOIN cwd_user ON portalpage.username=cwd_user.user_name
        JOIN app_user ON cwd_user.user_name=app_user.user_key
WHERE portletconfiguration.gadget_xml ~'createdvsresolved-gadget.xml';
```

Dashboard	url	Owner
My Custom Dashboard	http://jira.example.com/secure/Dashboard.jspa?selectPageId=10200	jturner

(1 row)

How do I fix it?

With JIRA shut down, *****BACK UP YOUR JIRA DATABASE*****, and then run this SQL:

```
CREATE TEMP SEQUENCE prefid;
SELECT setval('prefid', max(id)::integer + 1) FROM gadgetuserpreference;
WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
    filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
    cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
    relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration)),
    prefs AS (select portletconfiguration, projectOrFilterId, regexp_replace(projectOrFilterId, '-\d+', ''))
AS type, regexp_replace(projectOrFilterId, '(filter|project)-', '') AS id, iscumulative FROM relevantprefs),
    newvals AS (select nextval('prefid') AS ID,
        portletconfiguration,
        t.userprefkey,
        CASE t.userprefkey
            WHEN 'operation' THEN (CASE prefs.isCumulative WHEN
'true' THEN 'cumulative' ELSE 'count' END)
            WHEN 'id' THEN prefs.id
            WHEN 'name' THEN CASE type WHEN 'project' THEN (SELECT
pname FROM project WHERE id=prefs.id::integer) WHEN 'filter' THEN prefs.projectOrFilterId END
            WHEN 'type' THEN prefs.type
        END AS userprefvalue
    FROM prefs,
    (VALUES ('name'), ('operation'), ('id'), ('type')) AS t(userprefkey))
INSERT INTO gadgetuserpreference SELECT * FROM newvals;
UPDATE sequence_value_item SET seq_id = (SELECT ceil(max(id)/100)*100 FROM gadgetuserpreference) WHERE
seq_name='GadgetUserPreference';
```



There may be Postgres'isms in this SQL. Proceed with caution.

What's going on?

Each gadget has a bunch of parameters:

Created vs. Resolved Chart

Project or Saved

project-11353

Filter:

Search

Project or saved filter to use as the basis for the graph.

Advanced Search

Period:

Daily

The length of periods represented on the graph.

Days Previously

30

Days in the past to collect data for the selected period.

Collection Operation

Count

Count

Cumulative

individual values

Display the Trend of Unresolved

No

Show the number of unresolved issues over time in a subplot.

Display Versions

Only major versions

Show when versions were released on the chart.

Auto refresh

☒ Update every 15 minutes

Save

Cancel

These parameters are saved in the gadgetuserpreferences database table.

In JIRA 7.x, Atlassian reimplemented the Created vs Resolved gadget, and **changed the parameters**. Here is a before/after diff on the params for a filter-using portlet (see [Using database diffs to see what JIRA is doing](#) for details on the technique):

id	portletconfiguration	userprefkey	userprefvalue
12398	10801	daysprevious	600
12390	10801	isConfigured	true
12393	10801	isCumulative	true
12392	10801	isPopup	false
12396	10801	periodName	monthly
12397	10801	projectOrFilterId	filter-10601
12394	10801	refresh	false
12395	10801	showUnresolvedTrend	false
12391	10801	versionLabel	major

9 rows

id	portletconfiguration	userprefkey	userprefvalue
12625	10801	daysprevious	600
12626	10801	id	10601
12615	10801	isConfigured	true
12623	10801	isCumulative	true
12616	10801	isPopup	false
12624	10801	name	filter-10601
12627	10801	operation	cumulative
12619	10801	periodName	monthly
12620	10801	projectOrFilterId	filter-10601
12617	10801	refresh	false
12618	10801	showUnresolvedTrend	false
12621	10801	type	filter
12622	10801	versionLabel	major

13 rows

and a project-using portlet:

id	portletconfiguration	userprefkey	userprefvalue
12526	10900	daysprevious	500
12518	10900	isConfigured	true
12521	10900	isCumulative	true
12520	10900	isPopup	false
12524	10900	periodName	monthly
12525	10900	projectOrFilterId	project-10800
12522	10900	refresh	false
12523	10900	showUnresolvedTrend	false
12519	10900	versionLabel	major

9 rows

id	portletconfiguration	userprefkey	userprefvalue
12651	10900	daysprevious	500
12652	10900	id	10800
12641	10900	isConfigured	true
12649	10900	isCumulative	true
12642	10900	isPopup	false
12650	10900	name	IT Tasks
12653	10900	operation	cumulative
12645	10900	periodName	monthly
12646	10900	projectOrFilterId	project-10800
12643	10900	refresh	false
12644	10900	showUnresolvedTrend	false
12647	10900	type	project
12648	10900	versionLabel	major

13 rows

Specifically, they introduced 4 new parameters, type, id, name and operation, whose values are derived from the old parameters.

Atlassian **should** have written an 'upgrade task' to derive these new parameters from the old, and delete the old (to avoid confusing future generations). That is the topic of

 [JRASERVER-59364](#) - Created vs. Resolved gadget should respect the Cumulative Totals option set before the upgrade

CLOSED

But we can derive the new params ourselves with a bit of SQL.

Deriving the SQL Fix

Background: database tables involved in JIRA portlets

In the JIRA database:

- The `portalpage` table stores user's dashboards, e.g. "My Custom Dashboard".
- The `portletconfiguration` table stores individual portlets on dashboards. E.g. it records that there are N gadgets on "My Custom Dashboard", as well as their position and type.
- The `gadgetuserpreference` table stores portlet configuration details. E.g. for a "Created vs Resolved" portlet it stores the filter ID the portlet displays data from.

Database table	Sample data
portalpage	<pre>select * from portalpage where id=10200; id username pagename description sequence fav_count layout ppversion 10200 jturner My Custom Dashboard (1 row)</pre>
portletconfiguration	<pre>select * from portletconfiguration where portalpage=10200; id portalpage portlet_id column_number positionseq gadget_xml color dashboard_module_complete_key 10801 10200 1 0 rest/gadgets/1.0/g/com. atlassian.jira.gadgets:created-vs-resolved-issues-chart-gadget/gadgets/createdvsresolved- gadget.xml color1 10803 10200 0 0 rest/gadgets/1.0/g/com. atlassian.jira.gadgets:created-vs-resolved-issues-chart-gadget/gadgets/createdvsresolved- gadget.xml color1 10802 10200 0 1 rest/gadgets/1.0/g/com. atlassian.jira.gadgets:pie-chart-gadget/gadgets/piechart-gadget. xml color1 10800 10200 0 2 rest/gadgets/1.0/g/com. atlassian.jira.gadgets:assigned-to-me-gadget/gadgets/assigned-to-me-gadget. xml color1 (4 rows)</pre>

gadgetuserpreference	Pre-JIRA 7.0.5	JIRA 7.0.5+																																																																																																
	<pre>select * from gadgetuserpreference where portletconfiguration=10801 order by userprefkey;</pre>	<pre>select * from gadgetuserpreference where portletconfiguration=10801 order by userprefkey;</pre>																																																																																																
	<table><tr><th>id</th><th>portletconfiguration</th></tr><tr><th>userprefkey</th><th>userprefvalue</th></tr><tr><td>12398</td><td>10801</td></tr><tr><td>daysprevious</td><td>600</td></tr><tr><td>12390</td><td>10801</td></tr><tr><td>isConfigured</td><td>true</td></tr><tr><td>12393</td><td>10801</td></tr><tr><td>isCumulative</td><td>true</td></tr><tr><td>12392</td><td>10801</td></tr><tr><td>isPopup</td><td>false</td></tr><tr><td>12396</td><td>10801</td></tr><tr><td>periodName</td><td>monthly</td></tr><tr><td>12397</td><td>10801</td></tr><tr><td>projectOrFilterId</td><td>filter-10601</td></tr><tr><td>12394</td><td>10801</td></tr><tr><td>refresh</td><td>false</td></tr><tr><td>12395</td><td>10801</td></tr><tr><td>showUnresolvedTrend</td><td>false</td></tr><tr><td>12391</td><td>10801</td></tr><tr><td>versionLabel</td><td>major</td></tr></table>	id	portletconfiguration	userprefkey	userprefvalue	12398	10801	daysprevious	600	12390	10801	isConfigured	true	12393	10801	isCumulative	true	12392	10801	isPopup	false	12396	10801	periodName	monthly	12397	10801	projectOrFilterId	filter-10601	12394	10801	refresh	false	12395	10801	showUnresolvedTrend	false	12391	10801	versionLabel	major	<table><tr><th>id</th><th>portletconfiguration</th></tr><tr><th>userprefkey</th><th>userprefvalue</th></tr><tr><td>12525</td><td>10801</td></tr><tr><td>daysprevious</td><td>600</td></tr><tr><td>12526</td><td>10801</td></tr><tr><td>id</td><td>10601</td></tr><tr><td>12515</td><td>10801</td></tr><tr><td>isConfigured</td><td>true</td></tr><tr><td>12523</td><td>10801</td></tr><tr><td>isCumulative</td><td>true</td></tr><tr><td>12516</td><td>10801</td></tr><tr><td>isPopup</td><td>false</td></tr><tr><td>12524</td><td>10801</td></tr><tr><td>name</td><td>filter-10601</td></tr><tr><td>12527</td><td>10801</td></tr><tr><td>operation</td><td>cumulative</td></tr><tr><td>12519</td><td>10801</td></tr><tr><td>periodName</td><td>monthly</td></tr><tr><td>12520</td><td>10801</td></tr><tr><td>projectOrFilterId</td><td>filter-10601</td></tr><tr><td>12517</td><td>10801</td></tr><tr><td>refresh</td><td>false</td></tr><tr><td>12518</td><td>10801</td></tr><tr><td>showUnresolvedTrend</td><td>false</td></tr><tr><td>12521</td><td>10801</td></tr><tr><td>type</td><td>filter</td></tr><tr><td>12522</td><td>10801</td></tr><tr><td>versionLabel</td><td>major</td></tr></table>	id	portletconfiguration	userprefkey	userprefvalue	12525	10801	daysprevious	600	12526	10801	id	10601	12515	10801	isConfigured	true	12523	10801	isCumulative	true	12516	10801	isPopup	false	12524	10801	name	filter-10601	12527	10801	operation	cumulative	12519	10801	periodName	monthly	12520	10801	projectOrFilterId	filter-10601	12517	10801	refresh	false	12518	10801	showUnresolvedTrend	false	12521	10801	type	filter	12522	10801	versionLabel	major
	id	portletconfiguration																																																																																																
	userprefkey	userprefvalue																																																																																																
	12398	10801																																																																																																
	daysprevious	600																																																																																																
	12390	10801																																																																																																
	isConfigured	true																																																																																																
	12393	10801																																																																																																
isCumulative	true																																																																																																	
12392	10801																																																																																																	
isPopup	false																																																																																																	
12396	10801																																																																																																	
periodName	monthly																																																																																																	
12397	10801																																																																																																	
projectOrFilterId	filter-10601																																																																																																	
12394	10801																																																																																																	
refresh	false																																																																																																	
12395	10801																																																																																																	
showUnresolvedTrend	false																																																																																																	
12391	10801																																																																																																	
versionLabel	major																																																																																																	
id	portletconfiguration																																																																																																	
userprefkey	userprefvalue																																																																																																	
12525	10801																																																																																																	
daysprevious	600																																																																																																	
12526	10801																																																																																																	
id	10601																																																																																																	
12515	10801																																																																																																	
isConfigured	true																																																																																																	
12523	10801																																																																																																	
isCumulative	true																																																																																																	
12516	10801																																																																																																	
isPopup	false																																																																																																	
12524	10801																																																																																																	
name	filter-10601																																																																																																	
12527	10801																																																																																																	
operation	cumulative																																																																																																	
12519	10801																																																																																																	
periodName	monthly																																																																																																	
12520	10801																																																																																																	
projectOrFilterId	filter-10601																																																																																																	
12517	10801																																																																																																	
refresh	false																																																																																																	
12518	10801																																																																																																	
showUnresolvedTrend	false																																																																																																	
12521	10801																																																																																																	
type	filter																																																																																																	
12522	10801																																																																																																	
versionLabel	major																																																																																																	
	(9 rows)	(13 rows)																																																																																																

SQL part 1: collecting relevant old-portlet data

Let us look at the newly introduced preferences in more detail:

id	portletconfiguration	userprefkey	userprefvalue
12398	10801	daysprevious	600
12390	10801	isConfigured	true
12393	10801	isCumulative	true
12392	10801	isPopup	false
12396	10801	periodName	monthly
12397	10801	projectOrFilterId	filter-10601
12394	10801	refresh	false
12395	10801	showUnresolvedTrend	false
12391	10801	versionLabel	major

(9 rows)

id	portletconfiguration	userprefkey	userprefvalue
12625	10801	daysprevious	600
12626	10801	id	10601
12615	10801	isConfigured	true
12623	10801	isCumulative	true
12616	10801	isPopup	false
12624	10801	name	filter-10601
12627	10801	operation	cumulative
12619	10801	periodName	monthly
12620	10801	projectOrFilterId	filter-10601
12617	10801	refresh	false
12618	10801	showUnresolvedTrend	false
12621	10801	type	filter
12622	10801	versionLabel	major

(13 rows)

The new prefs are:

- **type** is project or filter, depending on the portlet's data source
- **id** is the ID of the filter or project
- **name** is a human-readable project or filter name, where:
 - if **type** is a project, then **name** should be the project name, and is actually displayed in the portlet.
 - if **type** is a filter, it doesn't really matter what **name** is – JIRA set it to filter-xxxxx and doesn't use it.
- **operation** has value cumulative or count

Recreating the missing parameters

Notice that all of the 4 new parameter values can be derived just by looking at the old **projectOrFilterId** and **isCumulative** parameters.

So first we write some SQL to print the **projectOrFilterId** and **isCumulative** parameters for each Created vs Resolved portlet:

```
WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration))
SELECT * From relevantprefs;
```

portletconfiguration	projectorfilterid	iscumulative
10801	filter-10601	true
10803	filter-10601	false
10900	project-10800	true

(3 rows)

The same, but using `regexp_replace()` to break **projectorfilterid** into its constituent parts:

```
jira=> WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration)),
prefs AS (select portletconfiguration, projectOrFilterId, regexp_replace(projectOrFilterId, '-\d+', '')
AS type, regexp_replace(projectOrFilterId, '(filter|project)-', '') AS id, iscumulative FROM relevantprefs)
SELECT * from prefs;
```

portletconfiguration	projectorfilterid	type	id	iscumulative
10801	filter-10601	filter	10601	true
10803	filter-10601	filter	10601	false
10900	project-10800	project	10800	true

(3 rows)

SQL part 2: inserting new rows

The trick here is generating 4 *new* rows for each matching row above. We can create a 'fake' table to cross-join with our **relevantprefs** using a **VALUES** clause:

```
jira=> select * from (VALUES ('name'), ('operation'), ('id'), ('type')) AS t(userprefkey);
```

userprefkey
name
operation
id
type

(4 rows)

Then 4 new rows for each of our portlets yields:

```

WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration)),
prefs AS (select portletconfiguration, projectOrFilterId, regexp_replace(projectOrFilterId, '-\d+', '')
AS type, regexp_replace(projectOrFilterId, '(filter|project)-', '') AS id, iscumulative FROM relevantprefs)
SELECT * from prefs,
(VALUE ('name'), ('operation'), ('id'), ('type')) AS t(userprefkey);

```

portletconfiguration	projectorfilterid	type	id	iscumulative	userprefkey
10801	filter-10601	filter	10601	true	name
10801	filter-10601	filter	10601	true	operation
10801	filter-10601	filter	10601	true	id
10801	filter-10601	filter	10601	true	type
10803	filter-10601	filter	10601	false	name
10803	filter-10601	filter	10601	false	operation
10803	filter-10601	filter	10601	false	id
10803	filter-10601	filter	10601	false	type
10900	project-10800	project	10800	true	name
10900	project-10800	project	10800	true	operation
10900	project-10800	project	10800	true	id
10900	project-10800	project	10800	true	type

(12 rows)

We now have a row containing all relevant old params, plus the new parameter name (name, operation, id and type). Now we just insert a SELECT clause to derive the new userprefvalue from the old params:

```

WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
    filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
    cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
    relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration)),
    prefs AS (select portletconfiguration, projectOrFilterId, regexp_replace(projectOrFilterId, '-\d+', '')
AS type, regexp_replace(projectOrFilterId, '(filter|project)-', '') AS id, iscumulative FROM relevantprefs)
SELECT *, CASE t.userprefkey
                                WHEN 'operation' THEN (CASE prefs.isCumulative WHEN
'true' THEN 'cumulative' ELSE 'count' END)
                                WHEN 'id' THEN prefs.id
                                WHEN 'name' THEN CASE type WHEN 'project' THEN (SELECT
pname FROM project WHERE id=prefs.id::integer) WHEN 'filter' THEN prefs.projectOrFilterId END
                                WHEN 'type' THEN prefs.type
                                END AS userprefvalue
FROM prefs,
    (VALUES ('name'), ('operation'), ('id'), ('type')) AS t(userprefkey);

portletconfiguration  projectorfilterid  type      id      iscumulative  userprefkey  userprefvalue
10801  filter-10601      filter    10601  true          name         filter-10601
10801  filter-10601      filter    10601  true          operation    cumulative
10801  filter-10601      filter    10601  true          id           10601
10801  filter-10601      filter    10601  true          type         filter
10803  filter-10601      filter    10601  false         name         filter-10601
10803  filter-10601      filter    10601  false         operation    count
10803  filter-10601      filter    10601  false         id           10601
10803  filter-10601      filter    10601  false         type         filter
10900  project-10800      project   10800  true          name         CrowdFlower
10900  project-10800      project   10800  true          operation    cumulative
10900  project-10800      project   10800  true          id           10800
10900  project-10800      project   10800  true          type         project

```

(12 rows)

Inserting data: dealing with IDs

We've successfully derived the new preferences needed. Now we just need to insert it into the gadgetuserpreferences table.

The gadgetuserpreferences table has an ID field which must be unique:

```

redradish_jira=> \d gadgetuserpreference
Table "public.gadgetuserpreference"

   Column          |      Type      | Modifiers
id                  | numeric(18,0)   | not null
portletconfiguration | numeric(18,0)   |
userprefkey        | character varying(255) |
userprefvalue       | text            |

Indexes:
    "pk_gadgetuserpreference" PRIMARY KEY, btree (id)
    "userpref_portletconfiguration" btree (portletconfiguration)

```

So we need to generate a sequence of IDs, starting at 1 more than the current max. This is done with a Postgres sequence:

```
CREATE TEMP SEQUENCE prefid;
SELECT setval('prefid', max(id)::integer + 1) FROM gadgetuserpreference;

setval

12712

(1 row)
```

Now we add a nextval('prefid') SELECT clause to our SQL, and trim down the results to match the gadgetuserprefs columns:

```
WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration)),
prefs AS (select portletconfiguration, projectOrFilterId, regexp_replace(projectOrFilterId, '-\d+', '')
AS type, regexp_replace(projectOrFilterId, '(filter|project)-', '') AS id, iscumulative FROM relevantprefs),
newvals AS (select nextval('prefid') AS ID,
portletconfiguration,
t.userprefkey,
CASE t.userprefkey
WHEN 'operation' THEN (CASE prefs.isCumulative WHEN
'true' THEN 'cumulative' ELSE 'count' END)
WHEN 'id' THEN prefs.id
WHEN 'name' THEN CASE type WHEN 'project' THEN (SELECT
pname FROM project WHERE id=prefs.id::integer) WHEN 'filter' THEN prefs.projectOrFilterId END
WHEN 'type' THEN prefs.type
END AS userprefvalue
FROM prefs,
(VALUE ('name'), ('operation'), ('id'), ('type')) AS t(userprefkey))
SELECT * FROM newvals;

id portletconfiguration userprefkey userprefvalue

12725 10801 name filter-10601
12726 10801 operation cumulative
12727 10801 id 10601
12728 10801 type filter
12729 10803 name filter-10601
12730 10803 operation count
12731 10803 id 10601
12732 10803 type filter
12733 10900 name CrowdFlower
12734 10900 operation cumulative
12735 10900 id 10800
12736 10900 type project

(12 rows)
```

And there we have exactly what we want to INSERT:

```

WITH gadgetprefs AS (select gup.* from gadgetuserpreference gup JOIN portletconfiguration pc ON gup.
portletconfiguration=pc.id WHERE pc.gadget_xml~'createdvsresolved-gadget.xml'),
    filterprefs AS (select * from gadgetprefs WHERE userprefkey='projectOrFilterId'),
    cumulativeprefs AS (select * from gadgetprefs WHERE userprefkey='isCumulative'),
    relevantprefs AS (select portletconfiguration, filterprefs.userprefvalue AS projectOrFilterId,
cumulativeprefs.userprefvalue AS isCumulative FROM filterprefs JOIN cumulativeprefs USING
(portletconfiguration)),
    prefs AS (select portletconfiguration, projectOrFilterId, regexp_replace(projectOrFilterId, '-\d+', ''))
AS type, regexp_replace(projectOrFilterId, '(filter|project)-', '') AS id, iscumulative FROM relevantprefs),
    newvals AS (select nextval('prefid') AS ID,
                    portletconfiguration,
                    t.userprefkey,
                    CASE t.userprefkey
                        WHEN 'operation' THEN (CASE prefs.isCumulative WHEN
'true' THEN 'cumulative' ELSE 'count' END)
                        WHEN 'id' THEN prefs.id
                        WHEN 'name' THEN CASE type WHEN 'project' THEN (SELECT
pname FROM project WHERE id=prefs.id::integer) WHEN 'filter' THEN prefs.projectOrFilterId END
                        WHEN 'type' THEN prefs.type
                    END AS userprefvalue
                FROM prefs,
                (VALUES ('name'), ('operation'), ('id'), ('type')) AS t(userprefkey))
    INSERT INTO gadgetuserpreference SELECT * FROM newvals;

INSERT 0 12
Time: 30.402 ms

```

Updating the sequence_value_item table

JIRA's database library (Ofbiz) has an odd way of tracking the maximum ID of a table. It has a table called sequence_value_item which stores the max ID of each table, rounded up to the nearest 100:

```

SELECT max(id) FROM gadgetuserpreference;

    max

12711

(1 row)

SELECT * from sequence_value_item WHERE seq_name='GadgetUserPreference';

    seq_name      seq_id

GadgetUserPreference  12800

(1 row)

```

So after manually inserting rows in a JIRA table, one must always update sequence_value_item. The new value can be derived from max(id) by a divide-then-round-up-then-multiple trick:

```

UPDATE sequence_value_item SET seq_id = (SELECT ceil(max(id)/100)*100 FROM gadgetuserpreference) WHERE
seq_name='GadgetUserPreference' RETURNING seq_id;

    seq_id

    12800

(1 row)

```

Shameless Plug

My business, Red Radish Consulting, offers a quarterly JIRA/Confluence upgrade service. The fix described here was derived on-the-fly for a client. Save yourself the hassle of dealing with exciting bugs like this, and [contact us](#) for a quote!