

Using database diffs to see what JIRA is doing

This is a small but very, very useful tip. Often there's situations where you **need to know what happened in the database**, for a particular JIRA operation (or any database-using webapp). This is how you do it.

It's pretty straightforward: generate a SQL dump of the JIRA database before and after, and then use `diff` to see the changes.

The only trick to know is that *for the diff to be legible, you need to generate SQL dumps with full INSERT statements on every line*. By default this is not true: Postgres emits a giant `COPY` block for each table, for instance. So:

- When using PostgreSQL, use `pg_dump --inserts`
- When using MySQL, use `mysqldump --skip-extended-insert`

To spell it out:

PostgreSQL

1. Generate a pre-change dump of the JIRA database:

```
pg_dump --inserts > ~/pre.sql
```

2. Make the JIRA change
3. Generate a post-change dump of the JIRA database:

```
pg_dump --inserts > ~/post.sql
```

4. Generate a diff of the changes:

```
diff ~/pre.sql ~/post.sql
```

MySQL

1. Generate a pre-change dump of the JIRA database:

```
mysqldump --skip-extended-insert > ~/pre.sql
```

2. Make the JIRA change
3. Generate a post-change dump of the JIRA database:


```
mysqldump --skip-extended-insert > ~/post.sql
```

4. Generate a diff of the changes:

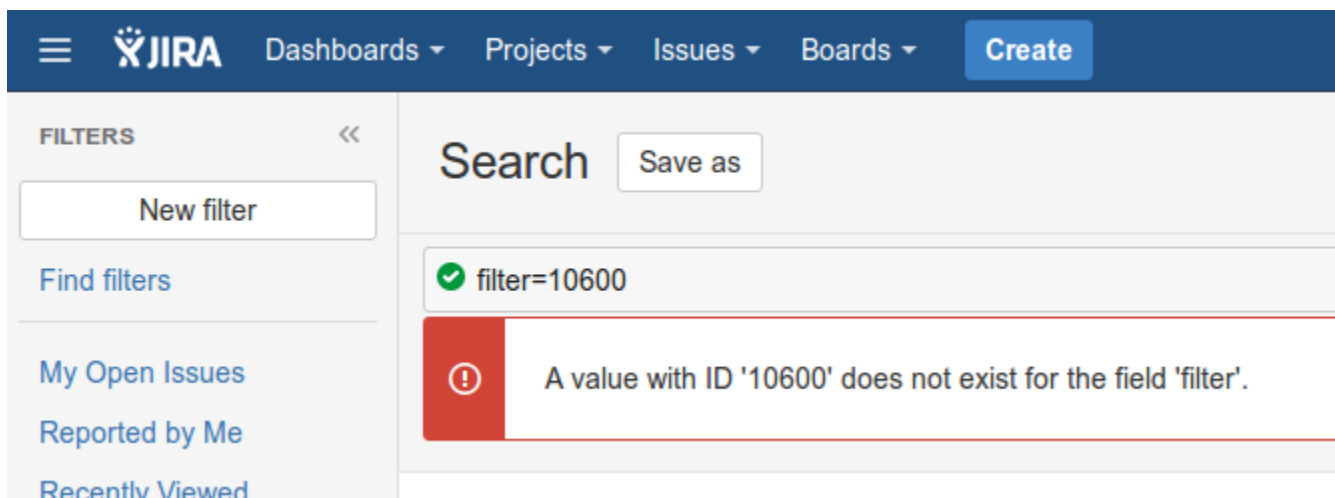
```
diff ~/pre.sql ~/post.sql
```

Example: finding private filters

Have you ever see this in Confluence?

Key	Summary	Type	Created
	JIRA project doesn't exist or you don't have permission to view it. View these issues in JIRA		

or been redirected to a JIRA search, only to see:

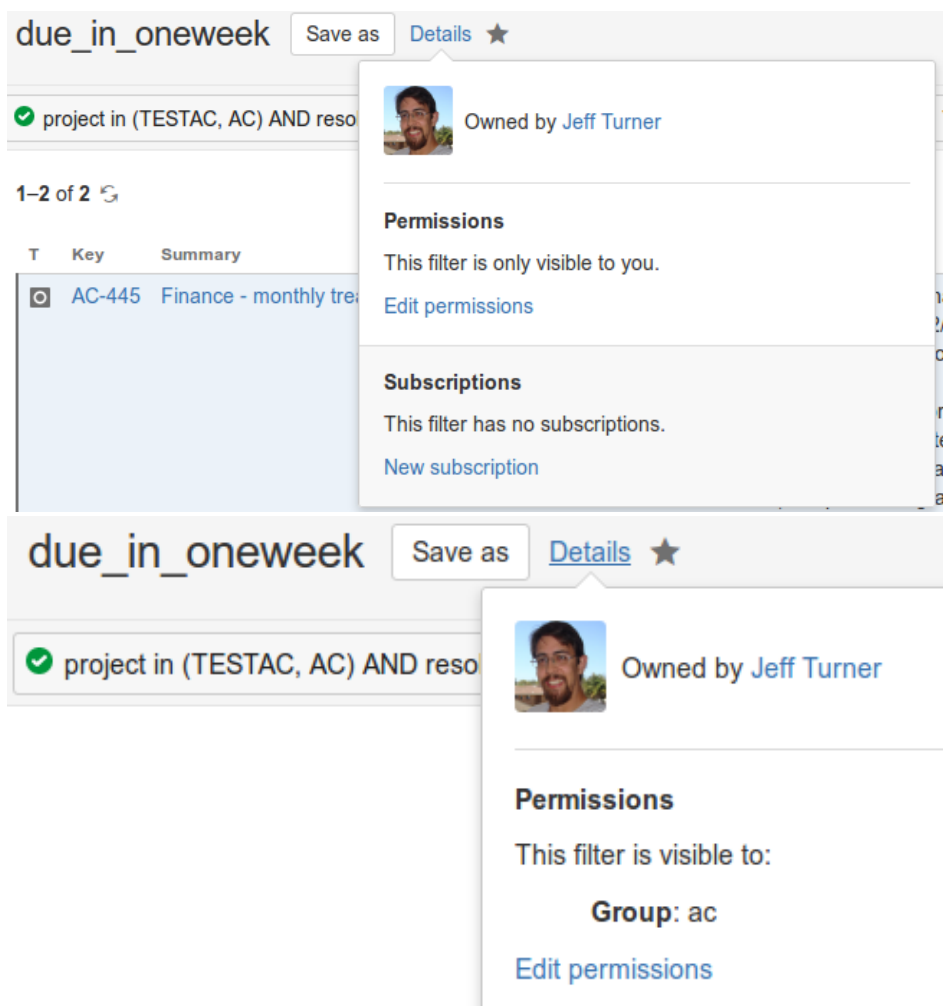


This is a common situation: someone has created a filter in JIRA, but forgotten to 'share' it (filters being private by default) before using it in a Confluence page or email link. The page looks fine to the sender, but not to anyone else.

How can we, as JIRA administrators, identify all private filters?

JIRA database changes

What is the database-level difference between a private filter, and the same filter shared with a group? Let's take a database diff, and make a sample filter visible to the `ac` group:



Here's the database diff:

```

root@coastserver [jira]~ # diff /tmp/{pre,post}.sql
52476c52476
< INSERT INTO propertytext VALUES (12996, 'project in (TESTAC, AC) AND resolution is EMPTY AND Reminders =
"1 week before" AND due >= 7d AND due < 8d');
---
> INSERT INTO propertytext VALUES (12996, NULL);
53568d53567
< INSERT INTO searchrequest VALUES (10600, 'due_in_oneweek', 'jturner', NULL, 'jturner', NULL, NULL,
'project in (TESTAC, AC) AND resolution is empty and Reminders = "1 week before" AND due >= 7d AND due <
8d', 1, 'due_in_oneweek');
53607a53607
> INSERT INTO searchrequest VALUES (10600, 'due_in_oneweek', 'jturner', '', 'jturner', NULL, NULL, 'project
in (TESTAC, AC) AND resolution is empty and Reminders = "1 week before" AND due >= 7d AND due < 8d', 1,
'due_in_oneweek');
53664d53663
< INSERT INTO sequence_value_item VALUES ('SharePermissions', 11100);
53712a53712
> INSERT INTO sequence_value_item VALUES ('SharePermissions', 11200);
53763a53764
> INSERT INTO sharepermissions VALUES (11100, 10600, 'SearchRequest', 'group', 'ac', NULL);
root@coastserver [jira]~ #

```

What does this tell us? The propertytext and searchrequest changes are minor and irrelevant. **What's important is the sharepermissions table**, which has gained a new value. The table looks like:

```

jira=> \d sharepermissions
      Table "public.sharepermissions"
   Column      |          Type          | Modifiers
-----+-----+-----
 id            | numeric(18,0)          | not null
 entityid      | numeric(18,0)          |
 entitytype    | character varying(60)  |
 sharetype     | character varying(10)  |
 param1        | character varying(255) |
 param2        | character varying(60)  |
Indexes:
    "pk_sharepermissions" PRIMARY KEY, btree (id)
    "share_index" btree (entityid, entitytype)

```

The second value, 10600, refers to the entityid, or ID of the searchrequest table entry.

So to find all unshared filters, simply search for all searchrequest rows without a sharepermission row:

```

jira=> SELECT username, id, filtername FROM searchrequest WHERE NOT EXISTS (SELECT * FROM sharepermissions
WHERE entityid=searchrequest.id AND entitytype='SearchRequest');
   username   | id  | filtername
-----+-----+-----
 jturner      | 10101 | All Keys
 jsmith       | 10500 | Stuff due in next 6 months
 jturner      | 10601 | due_in_onemonth
 jturner      | 10602 | due_in_threedays
 jturner      | 10603 | due_in_oneday
 jturner      | 10604 | due_today

```

Then, just to close off the example, we could find Confluence usages of these private filters by searching the Confluence database for the jira macro's XHTML:

```

<ac:structured-macro ac:macro-id="7714d651-dc9f-4168-af65-1ed994c6b630" ac:name="jira" &
  <ac:parameter ac:name="server">JIRA</ac:parameter>
  <ac:parameter ac:name="columns">key, summary, type, created, updated, due, assignee, reporter
  <ac:parameter ac:name="maximumIssues">20</ac:parameter>
  <ac:parameter ac:name="jqlQuery">filter=11202 </ac:parameter>
  <ac:parameter ac:name="serverId">e97c080d-536e-3787-9354-38d8c2a9b0e6</ac:parameter>
</ac:structured-macro>

```

with some bash-plus-SQL scripting:

```
echo "SELECT string_agg(''||id, '||') FROM searchrequest WHERE NOT EXISTS (SELECT * FROM sharepermissions
WHERE entityid=searchrequest.id AND entitytype='SearchRequest');" \
| psql -tAq jira \
| while read ids; do \
    echo "SELECT distinct content.contentid, content.title \
        FROM bodycontent JOIN content ON bodycontent.contentid=content.contentid \
        WHERE content.prevver IS NULL AND bodycontent.body ~ 'jQuery\>filter=({ids})' \
        GROUP BY 1"; done \
| psql -tAq confluence \
| while read id title; do echo "https://confluence.example.com/pages/viewpage.action?pageId=$id
$title"; done

https://confluence.example.com/pages/viewpage.action?pageId=21004542|AC Meeting 2016-1 Agenda
https://confluence.example.com/pages/viewpage.action?pageId=17793222|AC Meeting 2015-4 Agenda
```