JiraClient/Deskzilla Local Database Schema

This document describes database schema of Deskzilla/JiraClient local cache.

This is an advanced topic. You need to learn it only if you have to access the database with your own scripts and tools.

You may need to do so, for example, if your Bugzilla or Jira server database has been destroyed or corrupted and you have no backup. In this case, you may restore your server database data from the local cache.

> ⚠ This document has no hints on how to restore your server database, it describes only how to retrieve data from the local cache.

The local cache is an SQLite database. The database file is "items.db", it is located in the root of the workspace. Remote files such as attachments are stored in the "download" folder in the root of the workspace.

Items

Item and **attribute** are basic concepts of the data schema.

The local cache stores all data as **items**. An item has an identifier (an integer number) and **attribute** values. Basically, an **item** is an identifiable collection of key-value pairs, where keys are **attributes**.

Attributes correspond to SQL tables. Each attribute table has two columns: "item" and "value". The "item" column contains identifiers of **items** and the "value" column contains value of the **attribute** of the **item**.

Each **attribute** has an identifier. For example, "com.almworks.bugzilla:a:bug.summary" is the identifier of the "Summary" field of a Bugzilla bug.

The "_tables" SQL table contains mapping from **attribute** identifiers to their SQL tables.

The following SQL query display all attributes and their SQL tables:

```
SELECT * FROM _tables;
```

You may start with this query to explorer available attributes.

Special Item Attributes

This section describes some useful special **attributes**.

Connection

The "com.almworks.items.api.sync:a:connection" attribute identifies a connection an item belongs to. In general it is mapped to the "connection" SQL table.

A connection is an **item** which represents a JiraClient or Deskzilla connection.

You may find the connection configuration in the "config.xml" file in the root of the workspace. Connection configurations are "provider" tags under the root "engine" tag.

Each connection as GUID identifier in its "id" tag.

You may map a connection configuration to the connection **item** by this GUID with the following SQL query (replace GUID with your connection's GUID):

```
SELECT * FROM connection_id where value = '068193C9-C4A2-19D8-8000-00005DF6BEED';
```

Assuming that the connection **item** is "2285", the following query selects all **items** which belong to the connection:

```
SELECT item FROM connection where value = 2285;
```

Deleted Items

Deskzilla and JiraClient do not permanently delete **items** from their local caches. Instead they mark **items** as 'existing' with the "existing" SQL table.

In order to test if an **item** exists or if it has been deleted, you may use a query like this:

```
SELECT value FROM existing WHERE item = 2285;
```

The query returns single row with value "1" for existing **items** or empty result set if the **item** has been deleted.

Secondary Items

Secondary **items** are attachments, comments, Jira links and so on. Secondary **items** have an **attribute** which refers to their parent **item** (such as a Bugzilla bug).

For example, Deskzilla stores bug attachments as secondary **items**. Attachment **items** refer to a bug they belong to with the "com.almworks.bugzilla:a:attachment.bug" **attribute** ("attachment_bug" SQL table).

The following query finds all attachments of a Bugzilla bug (bug **item** is "8431"):

```
SELECT * FROM attachment_bug WHERE value = 8431;
```

> ⓘ This query may return deleted attachments as well.

Enum Items

Enum items are projects, components, versions and so on. Primary items (Bugzilla bug, Jira issues) refer to enum items by their corresponding attributes.

For example, the "com.almworks.bugzilla:a:bug.product" ("bug_product" SQL table) attribute of a Bugzilla bug points to a product item.

The following query displays the product item of a Bugzilla bug (bug **item** is "8431"):

```
SELECT value FROM bug_product WHERE item = 8431;
```

To get a display name of the product (item is "4673"), use the "com.almworks.bugzilla:a:product.visualKey" attribute ("product_visualkey" SQL table). The query is:

```
SELECT value FROM product_visualkey WHERE item = 4673;
```

Examples

Let's find an attached file of a Bugzilla bug #1584803.

Search "com.almworks.bugzilla:a:bug.id" ("bug_id" table) values for the bug ID:

```
SELECT item FROM bug_id WHERE value = 1584803;
```

	item
1	8431

Note, the query may return more results: bugs from different connections and deleted bugs.

Now, find all attachments of the bug: find **items** those refer to the bug **item** via "com.almworks.bugzilla:a:attachment.bug" ("attachment_bug" table) **attribut**
:

```
SELECT item FROM attachment_bug WHERE value=8431;
```

	item
1	8472
2	8482
3	8483
4	8484
5	8485
6	8486
7	8487

Some of these attachments may have been deleted.

SELECT item, value FROM attachment_filename WHERE item in (8472, 8482, 8483, 8485, 8486, 8487)
Now, find file names of these attachments: use the "com.almworks.bugzilla:a:attachment.fileName" ("attachment_filename" table) **attribute**:

	item	value
1	8472	1584803-versiondisplay-2531b1.patch
2	8482	1584803-lightning-calendar-253.patch
3	8483	1584803-sm-tempfix-60.patch
4	8485	1584803-sm-tempfix-60.patch
5	8486	1584803-sm-tempfix-60.patch
6	8487	1584803-version-beta1.patch

This query returns only downloaded attachments.

Now we know the file names of the bug's attachments and may find them in the "download" folder

.