The Chado schema documentation on this wiki is a mixture of generated content and material directly entered by GMOD users into this wiki. The generated part of the documentation consists of the table definitions that are included on the Chado module pages and on the Chado Tables page, listing all the tables in Chado.

All of the table descriptions on Chado module pages and the Chado Tables page are generated every time there is a new release of Chado. The column and table details, including comments, come from the PostgreSQL data dictionary.

Using Module and Table Documentation

This section describes how to *use* the Chado Schema Documentation when updating/creating content on this wiki.

Showing a Table Description

To show the information about any table, use the *Chado Table Template* for that table:

```
{{ChadoTable_tablename}}
```

For example,

```
{{ChadoTable_cv}}
```

will show the table description for **cv**:

Table: **cv**
Module: **CV**
A controlled vocabulary or ontology. A cv is composed of cvterms (AKA terms, classes, types, universals - relations and properties are also stored in cvterm) and the relationships between them.

cv columns

FK	Name	Type	Description
	cv_id	serial	*PRIMARY KEY* *UNIQUE*
	name	character varying(255)	*NOT NULL* The name of the ontology. This corresponds to the obo-format -namespace-. cv names uniquely identify the cv. In OBO file format, the cv.name is known as the namespace.
	definition	text	A text description of the criteria for membership of this ontology.

Tables referencing **cv** via foreign key constraints:

- **cvterm**
- **cvtermpath**

Linking to Module Documentation

To link to a module page from another wiki page use:

```
{{ChadoModuleLink|Module Name|text to show}}
```

For example:

```
{{ChadoModuleLink|Publication|pub module}}
```

Which is shown as:

pub module

Linking to Table Documentation

To link to a specific table's description on a wiki page, use one of these:

```
{{ChadoModuleTableLink|Module Name|table_name}}
{{ChadoTableLink|table_name}}
```

For example:

```
{{ChadoModuleTableLink|Sequence|feature}}
{{ChadoTableLink|feature}}
```

will result in:

feature which links to the table on the table's module page
feature which links to the table on the Chado Tables page.

Updating Table Documentation Part I

Chado Table Templates are the building blocks of the Chado schema documentation. They are also auto-generated, and cannot be directly updated by editing them on the wiki. Preventing editing of the templates ensures that user updates are not just written over and lost the next time the templates are auto-generated.

However, it is possible to update table documentation. It is a two-step process and you must take the first step. Here's the recipe:

1. Enter your update on the wiki.
    1. Login to the wiki
    2. Go to the module page for the module the table is in.
    3. Click on the [edit] link to the right of the table you want to comment on.
    4. Go to the edit window towards the bottom of the page, and add your comments below the section "Additional Comments"
    5. Save the edits.
2. The next time Chado is released, all the module pages will be reviewed for any comments that have been added since the last update.

1. New comments will be added to the Chado SQL table definitions (and probably dropped from the "Additional Comments" sections).
2. The table documentation will be regenerated and reposted to the wiki as part of the Chado release.

See Updating Table Documentation Part II for more on how step #2 is done.

Updating Module and Table Documentation Part II

Table Templates are themselves a nest of smaller MediaWiki templates. This means it's hard to figure out how the wiki decides what to show. However, all this complexity doesn't really matter to the wiki user or even to the wiki editor. It is all auto-generated, and it is never directly updated using the wiki interface. The upside of this complexity is that it is easy to change the appearance of all Chado tables in the wiki. All you do is modify the appropriate template.

This section describes how to regenerate the Table Templates from a live Chado database as part of creating a new release of Chado. **If you are not creating a Chado release, or not a GMOD web site manager, then you don't care about this section.**

This step is itself a multistep process:

1. Integrate new comments on module pages with comments in the SQL schema.
2. Regenerate the wiki content
3. Push new wiki content to GMOD.org.

Integrate New Comments Into SQL DDL

This step involves walking through all the Chado Module pages, looking at any "Additional Comments" that have been added since the last Chado release, and then integrating them with the comments in the SQL DDL definitions of the tables. Integrated comments should then be removed from the module pages.

Regenerate Wiki Content

Next regenerate the wiki content after you have created a Chado instance using the newly updated SQL. This is done with scripts in the Chado source tree:

```
cd chado/doc/wiki
```

Edit `generateChadoWikiTables.py` and update any of these variables that you need to:

```
# UPDATE THESE 4 BEFORE RUNNING THE PROGRAM.
DB_NAME           = "testdb"
DB_USER           = "gmodhack"
MODULE_TABLE_PATH = "../../modules/module-tables.json"
WIKI_DIR          = "/tmp/ChadoWikiFiles"
```

Before you can run this script, make sure that the `postgresql_autodoc` package is installed. The script won't run without it. Now run:

```
$ ./generateChadoWikiTables.py
Producing testdb.wiki from ./wiki.tmpl
$
```

Note: This script will run for a *looong* time. It takes 18 minutes on my laptop.

This script places generated wiki content in the `WIKI_DIR` directory, which by default is `/tmp/ChadoWikiFiles/`:

`/tmp/ChadoWikiFiles/`	Determined by what `WIKI_DIR` is set to.
`Modules/`	Contains one file per module. These become the "Tables" sections of the Chado module pages
`Tables/`	Contains one file per table; these will become Table Templates.
`allTables.wiki`	List of all tables; will become the module/table list on Chado Tables.

Push Regenerated Wiki Text to GMOD.org

Now we need to update the wiki itself. First tar/compress the directory containing all the generated wiki files. Then copy it to the GMOD web server and unpack it. You'll need shell access to the GMOD web server to do this. Then update the three types of pages:

- Table Templates
- Module Pages
- Chado Tables page

Update Table Templates

On the GMOD web server create this script:

```
#!/usr/bin/python
# ===================
#
# Upload all the table templates to the GMOD wiki.

IMPORT_TEXT_FILE_PATH = "/var/www/htmlhttp://www.gmod.org/w/maintenance/importTextFile.php"
TABLES_DIR = "Tables"
MW_USERNAME = "Your gmod.org MediaWiki Username.  e.g., 'Clements'"
COMMENT = "Table definition  for Chado Version a.b on yyyy/mm/dd"

import glob
import os

for tablePath in glob.glob(TABLES_DIR + "/*.wiki"):
    tableFile = os.path.split(tablePath)[1]
    tableName = tableFile.split(".")[0]
    command = (
        "/usr/bin/php " + IMPORT_TEXT_FILE_PATH +
        " --title 'Template:ChadoTable_" + tableName +
        "' --user '" + MW_USERNAME + "' --comment '" + COMMENT + "' " +
        tablePath)
    print(command)
```

```
os.system(command)
```

Set `MW_USERNAME` and `COMMENT` appropriately. Also set `TABLES_DIR` to be the relative path from the script to directory containing the Table Templates.

This script uses the Mediawiki maintenance script `ImportTextFile.php` to upload the Table Templates.

Update Module Pages

We don't yet have an automated way to synchronize the module pages with the update.

The *first time* this process is done (the process described on this page), you'll probably want to do a wholesale replacement of the "Tables" section of each page. This will replace the current hard-coded table defs, with the Table Templates. It will also update the table list to be current.

On subsequent updates, you will only need to touch the module pages if a table was dropped or added. The Table Templates will take care of the rest.

Update Chado Tables Page

The Chado Tables page lists every table defined in Chado. Replace this wholesale each time.

Why is this complicated?

With any programatically generated wiki doc we've got conflicting goals:

1. Keep the doc close to the source. In this case that means in the SQL table definitions
2. Make that doc available on the wiki, where it is easy to find
3. Keep the wiki doc in sync with the source doc.
4. Allow users to use the wiki to update doc, and not have it be lost every time the wiki doc is regenerated.

We've tackled this for Chado with MediaWiki Templates:

1. There's a template for each table. (e.g., Template:ChadoTable_cv)
2. The templates are protected and can't be edited by regular wiki users (prevents lost updates)
3. The Chado Module pages include the templates and clearly have a place for additional comments to be added. (Encourages updates without requiring SVN update access).
4. Those comments can be incorporated into the SQL on the next Chado update, prior to regenerating the templates.
5. The Chado Table templates are regenerated and reloaded for every Chado release.

Internals

Documentation on the internals of how all this is done.

wiki.tmpl

`wiki.tmpl` defines the template used by `postgresql_autodoc` to generate the initial MediaWiki definitions of the tables. Since it generates MediaWiki markup and that markup is sensitive to newlines and blank lines, this template is unreadable. Therefore, a version is included below that is indented and has newlines. Note that this version won't work, but hopefully it will make the version that is used be more readable.

```
<!-- TMPL_LOOP name="schemas" -->
  <!-- TMPL_LOOP name="tables" -->
    <!-- TMPL_UNLESS name="view_definition" -->
      __TABLE_START__

        <noinclude>{{ChadoTableTemplateHeader}}</noinclude>
        {{ChadoTableDesc
            |__MODULE__
            |<!-- TMPL_VAR ESCAPE="HTML" name="table" -->
            |<!-- TMPL_IF name="table_comment" -->
               <!-- TMPL_VAR ESCAPE="HTML" name="table_comment" -->
             <!-- /TMPL_IF name="table_comment" -->
        }}
        {{ChadoColumnsHeader
            |__MODULE__
            |<!-- TMPL_VAR ESCAPE="HTML" name="table" -->
        }}
        <!-- TMPL_LOOP name="columns" -->
          {{ChadoColumnDesc
            |<!-- TMPL_LOOP name="column_constraints" -->
               <!-- TMPL_VAR name="column_constraints" -->
               <!-- TMPL_IF name="column_fk" -->
                 {{ChadoModuleTableLink
                     |__FK_MODULE__
                     |<!-- TMPL_VAR ESCAPE="HTML" name="column_fk_table" -->}}
               <!-- /TMPL_IF name="column_fk" -->
             <!-- /TMPL_LOOP name="column_constraints" -->
            |<!-- TMPL_VAR ESCAPE="HTML" name="column" -->
            |<!-- TMPL_VAR ESCAPE="HTML" name="column_type" -->
            |<!-- TMPL_LOOP name="column_constraints" -->
               <!-- TMPL_IF name="column_primary_key" -->
                 ''PRIMARY KEY''<br />
               <!-- /TMPL_IF name="column_primary_key" -->
               <!-- TMPL_IF name="column_unique" -->
                 ''UNIQUE
                 <!-- TMPL_IF name="column_unique_keygroup" -->
                   #<!-- TMPL_VAR name="column_unique_keygroup" -->
                 <!-- /TMPL_IF name="column_unique_keygroup" -->''<br />
               <!-- /TMPL_IF name="column_unique" -->
             <!-- /TMPL_LOOP name="column_constraints" -->
             <!-- TMPL_IF name="column_constraint_notnull" -->
               ''NOT NULL''<br />
             <!-- /TMPL_IF name="column_constraint_notnull" -->
             <!-- TMPL_IF name="column_default" -->
               ''DEFAULT ''<!-- TMPL_VAR ESCAPE="HTML" name="column_default" --><br />
             <!-- /TMPL_IF name="column_default" -->
             <!-- TMPL_IF name="column_comment" -->
               <!-- TMPL_VAR ESCAPE="HTML" name="column_comment" -->
             <!-- /TMPL_IF name="column_comment" -->
          }}
```

```
        <!-- /TMPL_LOOP name="columns" -->
        {{ChadoColumnsFooter}}

        {{ChadoTablesReferencingHeader
            |<!-- TMPL_VAR ESCAPE="HTML" name="table" -->
        }}
        <!-- TMPL_IF name="fk_schemas" -->
          <!-- TMPL_LOOP name="fk_schemas" -->
            {{ChadoReferencingTable
                |__FK_MODULE__
                |<!-- TMPL_VAR ESCAPE="HTML" name="fk_table" -->
            }}
          <!-- /TMPL_LOOP name="fk_schemas" -->
        <!-- TMPL_ELSE name="fk_schemas" -->
          * None.
        <!-- /TMPL_IF name="fk_schemas" -->

    <!-- /TMPL_UNLESS name="view_definition">
  <!-- /TMPL_LOOP name="tables" -->
<!-- /TMPL_LOOP name="schemas" -->
```

There ya go. Clear as mud.