MolGears Docs

version 1

Adrian Jasinski

March 19, 2015

Contents

MolGears's

The Molgears is a database tool for storing chemical compounds, calculating descriptors, tracking synthesis, storing analytical and bioactivity data. Bioactivity is reported in IC50 results. Data can be filtered, analyzed and downloaded to various file formats.

See *An overview of the MolGears* to learn more.

Molgears is accessed via a web interface. See *Getting started* to look at interface.

Documentation contents

MolGears's

An overview of the MolGears

What is it?

This is a opensorce framework/database/toolkit based on a TurboGears web framework and RDKit open source toolkit for cheminformatics:

- programed in Python2.7 language
- BSD license
- source code on Github (https://github.com/admed/molgears)

Software Dependencies/Third-party software component

- RDKiT (http://www.rdkit.org)
- TurboGears (http://turbogears.org/)
- Postgresql Database (http://www.postgresql.org/)
- RDKit database cartridge for postgresql (http://www.rdkit.org/docs/Cartridge.html)
- Genshi (http://genshi.edgewall.org/)
- JSME editor (http://peter-ertl.com/jsme/)
- **Python based libraries:**
    - SQLAlchemy (http://www.sqlalchemy.org/)
    - razi (http://razi.readthedocs.org/en/latest/index.html)
    - Numpy (http://www.numpy.org/)
    - Scipy (http://www.scipy.org/)
    - Matplotlib (http://matplotlib.org/)
    - Xlwt & Xlrd (http://www.python-excel.org/)
    - xhtml2pdf (http://www.xhtml2pdf.com/)
    - pillow (https://github.com/python-pillow/Pillow)

Design goal

- Project management tool
- Efficient data storage
- Sorting, analysis, aggregation and reporting of data
- Data visualization
- Improved data access
- Automation of procedures
- Facilitate communication

Features

- Multi-projects
- Adding molecules by drawing or pasting SMILES code
- Reading molecules from file (csv, smi, sdf, mol, txt)
- Data presenting in sortable columns
- Exporting data to file (file formats: xls, pdf, csv, sdf, txt, png)
- PAINS (Pan Assay Interference Compounds) filtering (DOI: 10.1021/jm901137j)
- History of changes
- Compound filtering (by similarity, structure, identity, compound name, creator, adding date etc.)
- Stars rating
- IC50 determination based on least squares method,
- Automatic data processing
- Graphs generation
- Access managing
- Tags

License

Project Workflow Overview

Describe the workflow general schemes associated with a project.

Multiproject

- MolGears allows you divide your work into projects. The solution is designed to handle project management for small compounds.
- Molgears allows you to manage access and grant authorized users the right to use the service and access permision for each project independly.
- **In addition each project has basic workflow divided into tasks:**
    - designing process,

- ordering synthesis,
- tracking sythesis process,
- storage compound in library,
- result tool for compounds activity.

Project basic workflow

- A repeatable process that brings similar stages each time
- A clear division of responsibility between different people
- A better basis to estimate task length
- A simple method to communicate process and data to all or selected teem members/employees/colabollators/
- Each task has history records so you can see who added, approved and edited each record.

Basic Workflow Schema

For each project in menu bar you have access to 5 main tables connected to the project workflow.

- Compounds - root table. Storage for molecules structures; both: ideas and existing compounds. Unique records for each structure.
- Requests - put ideas into real things. Table for synthesis requests.
- Synthesis - table for tracking synthesis progress, priority managing, and analytical data storage.
- Library - table only for existing compounds. The library of compounds with ability for tracking location and the amount of compound.
- Results - compound activity data storage connected to library instance. It's allowed to add for one library record many results.

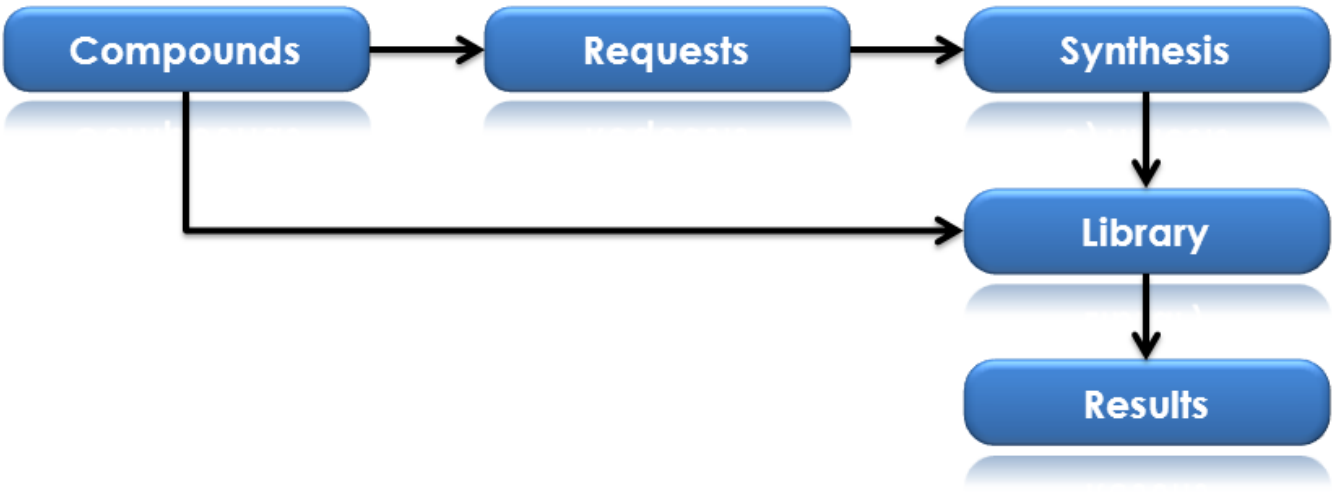
Workflow schema for the tables:

I. Add structures to Compounds table.

II. You can choose two path:

1. Internal synthesis:
    a. Create synthesis request
    b. Accept synthesis by chemist and proceed synthesis phases
    c. Add finished compound from synthesis to library
2. External synthesis (e.g. you're buying ready product):
    a. Accept compound directly to library

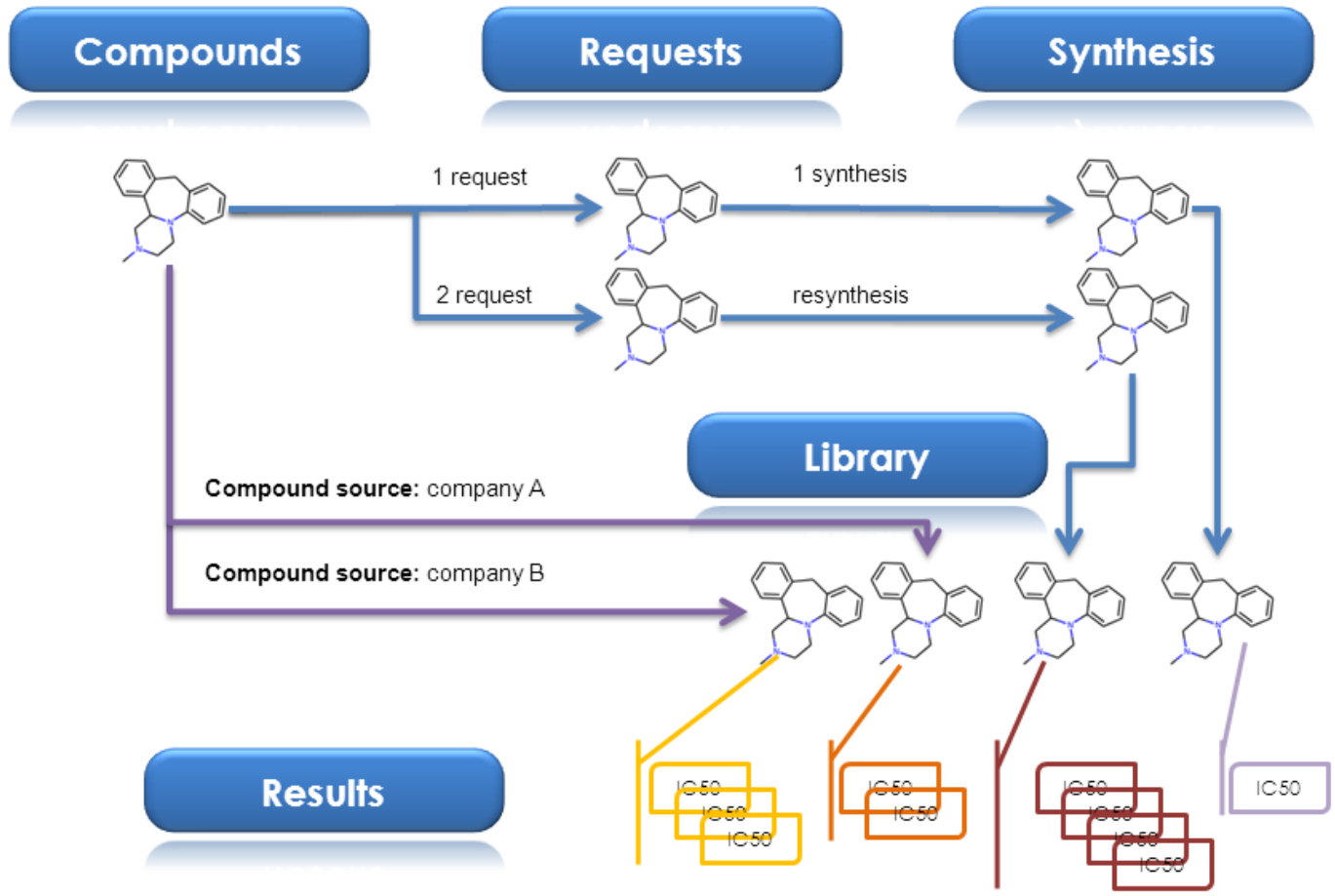
III. Add one or more activity results to library instance.



Example for the various sources of sample

Compounds table is containing unique structures of molecules. This means you can't add the same structure twice to Compounds Table. Only exception for this rule is adding compound as isomer. **[references]**. This rule also don't apply to other tables. You can create many request for one compound (structure) and then accept them to synthesis. One of the example of using this mechanism is performing synthesis of small amount of some compound. Later when we need more we can order the resynthesis by putting the same compound second time into request and follow the workflow once again.

Separate case is when you don't want to carry out the synthesis procedure and want to add a compound directly to the library e.g. when you're buying ready product from *company A*. Then follow the second path in workflow (see *Project basic workflow*). But you can also add this like that many times. If you add this twice and earlier when you performed 2 synthesis procedure as a result you will have 4 instances in library table referenced for the same structure. Than you can add one or more activity results independently for each of library instance of your compound. The shema illustrating this is presented below:



Database model

Database model was built based on Project Workflow (see: *Project Workflow Overview*).

Simple Model

Compounds and **Projects** tables are connected by many to many relationship with allowing to adding one compound to many projects and many compounds to one project.

Compounds Table is containing basic information about molecule struture like:

- SMILES and InChi code,
- name
- fingerprints
- molecular weight
- logP
- numer of atoms with and without Hydrogens
- number of rings
- Hbond acceptors
- Hbond donors
- Teorethical PSA

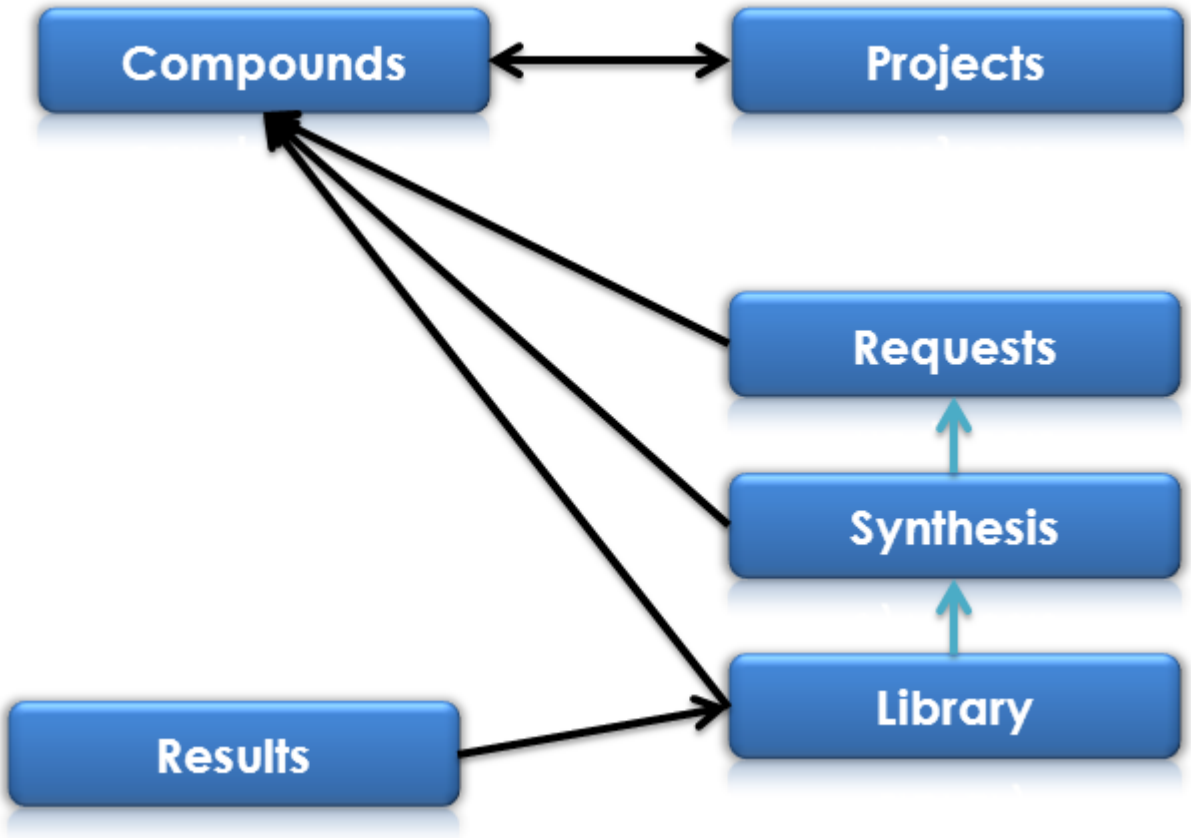
Requests, **Synthesis** and **Library** tables are refering to this data by relationship to the Compounds. This mean that editing and changing structure for Compound will make changes for all branches in all connected tables.

Synthesis table is storing ID of **Request** instance from which it is derived. This making the posibility for tracking the pathway of compound from one to other table and the ability to implement The Project Workflow in a consistent way.

The same connection is between **Synthesis** and **Library** tables.

Results table has many to one relationship to **Library**. In this way it's aviailble to add many results to one library instance.

Basic schema of database model is presented below:



Detailed Model

Entity - Relationship Diagram for MolGears Database model is presented below.

Diagram was generated using DBSchema.

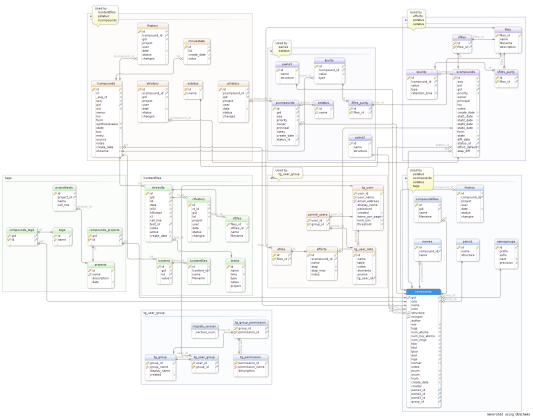
Main 6 Tables are respectively named as:

Model Name	SQL name
Projects	projects
Compounds	compounds
Requests	pcompounds
Synthesis	scompounds
Library	lcompounds
Results	ctoxicity

- Each of the main tables (except the Projects Table) has dedicated history tables for storing changes.
- Request and Synthesis tables have dedicated tables for Status.

- Compounds, Synthesis, Library and Results tables have dedicated tables for storing files.
- There are 3 PAINS tables for storing SMARTS codes.
- The tables for access managing have "tg_" prefix (tables for Users accounts, Users Lists, Groups and Permissions).
- Other tables are auxiliary and servs i.a. for auto-naming, state and purity monitoring, test description, tags storing etc.

ER Diagram (click the image to enlarge):



Installation

Installing procedure is tested only for Ubuntu 12.04+ and other debian-derived systems but general steps should work for other systems

Installing on Debian / Ubuntu

Installation steps one by one.

1. Installing TurboGears

TurboGears is meant to run inside python virtualenv and provides its own private index to avoid messing with your system packages and to provide a reliable set of packages that will correctly work together. First install python & python-virtualenv:

```
$ sudo apt-get install python
$ sudo apt-get install python-virtualenv
```

Running virtualenv with the option *--no-site-packages* will not include the packages that are installed globally. This can be useful for keeping the package list clean in case it needs to be accessed later.:

```
$ virtualenv --no-site-packages tg2env
```

in this tutorial we assume that you are in your home directory (~/ or /home/username) but you can choose your own directory for installation destination.

To begin using the virtual environment, it needs to be activated:

```
$ source tg2env/bin/activate
(tg2env)$ cd tg2env            #now you are ready to work with TurboGears
```

Be sure that you are in virtualenv mode during all installation procedure.

Install turbogears:

```
(tg2env)$ pip install tg.devtools
```

Start the project:

```
(tg2env)$ gearbox quickstart molgears
```

Delete unneeded files created by quickstart. We will replace it later by our project:

```
(tg2env)$ rm -rf ~/tg2env/molgears/molgears
```

Download molgears project from githubhttps://github.com/admed/molgears to ~/tg2env/molgears/

or use git:

```
(tg2env)$ cd ~/tg2env/molgears/
(tg2env)$ git clone https://github.com/admed/molgears
```

2. Build rdkit from source code with INCHI support

Read the rdkit docs Install for more informations.

Installing prerequisites

Install the following packages using apt-get:

```
flex bison build-essential python-numpy cmake python-dev
libboost-dev libboost-python-dev libboost-regex-dev
```

Setting environment variables

RDBASE: the root directory of the RDKit distribution (e.g. ~/RDKit):

```
export RDBASE="/home/username/RDKit"
```

LD_LIBRARY_PATH: make sure it includes $RDBASE/lib and wherever the boost shared libraries were installed:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$RDBASE/lib
```

PYTHONPATH: make sure it includes $RDBASE:

```
export PYTHONPATH=$PYTHONPATH:$RDBASE
```

Building the RDKit

Fetch the source, here as tar.gz but you could use git as well:

```
wget http://downloads.sourceforge.net/project/rdkit/rdkit/QX_20XX/RDKit_20XX_XX_X.tgz
```

for download the latest version (Q3 2014):

```
(tg2env)$ mkdir ~/RDKit
(tg2env)$ cd ~/RDKit
(tg2env)$ wget http://downloads.sourceforge.net/project/rdkit/rdkit/Q3_2014/RDKit_2014_09_1.
```

unpack the archive:

```
(tg2env)$ tar -xvf RDKit_2014_09_1.tgz
```

move files to ~/RDKit directory:

```
(tg2env)$ mv RDKit_2014_09_1/* ~/RDKit
```

Download InChi:

```
(tg2env)$ cd ~/RDKit/External/INCHI-API
(tg2env)$ ./download-inchi.sh
```

Building:

```
(tg2env)$ cd $RDBASE
(tg2env)$ mkdir build
(tg2env)$ cd build
(tg2env)$ cmake -DRDK_BUILD_INCHI_SUPPORT=ON ..
(tg2env)$ make
(tg2env)$ make install
```

Testing the build (optional, but recommended):

```
(tg2env)$ ctest
```

copy rdkit directory to:

```
(tg2env)$ cp ~/RDKit/rdkit ~/tg2env/lib/python2.X/site-packages/
```

3. Download razi

Razi provides extensions to SQLAlchemy to work with chemical databases.

Download razi from GitHub or from my fork

or by git:

```
(tg2env)$ cd ~/RDKit
(tg2env)$ git clone https://github.com/rvianello/razi
```

copy razi/razi to: ~/tg2env/lib/python2.X/site-packages/:

```
(tg2env)$ cp ~/RDKit/razi/razi ~/tg2env/lib/python2.X/site-packages/
```

4. Install postgresql

To install use the command line and type:

```
(tg2env)$ sudo apt-get install postgresql postgresql-contrib
```

Basic Server Setup

In a terminal, type:

```
(tg2env)$ sudo -u postgres psql postgres
```

Set a password for the "postgres" database role using the command:

```
postgres=# \password postgres
```

and give your password when prompted. The password text will be hidden from the console for security purposes.

Type Control+D to exit the posgreSQL prompt.

Since the only user who can connect to a fresh install is the postgres user, here is how to create yourself a database account (which is in this case also a database superuser) with the same name as your login name and then create a password for the user:

```
(tg2env)$ sudo -u postgres createuser --superuser $USER
(tg2env)$ sudo -u postgres psql
```

```
postgres=# \password $USER
```

Type Control+D to exit the posgreSQL prompt.

More installation information.

Creating a database

To create the first database, which we will call "molgears", simply type:

```
(tg2env)$ sudo -u postgres createdb molgears
```

Configuration

To improve performance while loading the database and building the index, I changed a couple of postgres configuration settings in postgresql.conf as they recommend in rdkit cartridge docs:

```
fsync = off                             # turns forced synchronization on or off
synchronous_commit = off                # immediate fsync at commit
full_page_writes = off                  # recover from partial page writes
```

And to improve search performance, I allowed postgresql to use more memory than the extremely conservative default settings:

```
shared_buffers = 2048MB                 # min 128kB
work_mem = 128MB                        # min 64kB
```

Change requires restart:

```
(tg2env)$ sudo /etc/init.d/postgresql restart
```

5. Build the cartridge

Go to the cartridge directory:

```
(tg2env)$ cd $RDBASE/Code/PgSQL/rdkit
```

run compilation, installation and testing:

```
(tg2env)$ make && make install && make installcheck
```

Add rdkit cartridge extension to database:

```
(tg2env)$ sudo -u postgres psql -c 'create extension rdkit' molgears
```

More info in rdkit docs and rdkit google code.

6. development.ini

Add information about your database to development.ini file. Edit the file: ~/tg2env/molgears/development.ini and comment the line:

```
# sqlalchemy.url = sqlite:///%(here)s/devdata.db
```

Than uncomment the line:

```
sqlalchemy.url=postgres://username:password@hostname:port/databasename
```

put your data like database USER name, password, hostname (or host IP i.e. 127.0.0.1), port (i.e. 8080) and databasename (in this example "molgears")

example line:

```
sqlalchemy.url=postgres://mike:hiddenpassword@127.0.0.1:8080/molgears
```

7. Install additional libraries by pip

Install by using the command:

```
(tg2env)$ pip install library_name
```

where libraries are::

- tw2.forms
- tw.forms
- Genshi
- zope.sqlalchemy
- webhelpers
- repoe.who
- repoze.who.plugins.sa
- psycopg2
- tgext.admin
- pillow
- paste

- xhtml2pdf
- xlwt
- xlrd
- numpy
- scipy
- matplotlib
- sqlalchemy_migrate

8. RUN your aplication

Go to molgears directory:

```
(tg2env)$ cd ~/tg2env/molgears
```

run setup.py:

```
(tg2env)$ python setup.py develop
(tg2env)$ gearbox setup-app
```

run gearbox:

```
(tg2env)$ gearbox serve
```

than you can open your browser and go to address:

```
http://hostname:port (i.e. http://127.0.0.1:8080).
```

put your data:

```
login: manager
passwor: managepass
```

enjoy :)

Getting started

What is this?

This document is intended to provide an overview of web interface functionality.

For bugs fix and/or suggestions for improvements, please contact me: adrian.jasinski@adamed.com.pl

See first *Installation* for running web interface.

Admin Panel

To enter Admin Panel you should have admin privileges (managers group).

Click **Admin** on top right side of the page (in horizontal menu).

From this panel you can create and manage: Users, Projects and Tags.

Admin Panel screenshot is presented below:

Projects Table

In project table you have listed all created projects. In *actions* column you can edit (by clicking pencil) or delete a project (delete button).

New Project < 1 >

actions	Name	Created	Cell Lines	Description
	new_project	2014-12-18 11:04:46	space_line, all_align, new_line	test
	test_new	2014-12-05 11:32:01	TEST2, TEST1, ACL, NBA	Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
	old_project	2014-10-20 13:31:59	JAR, MDM4	test
	project1	2014-10-20 09:16:02	JAR, SJSA-1, A-549, JEG-3, HCT-116++, MCF-7, LNCaP, A-431, PC-3, HCT-116--, MCF-10a	test1

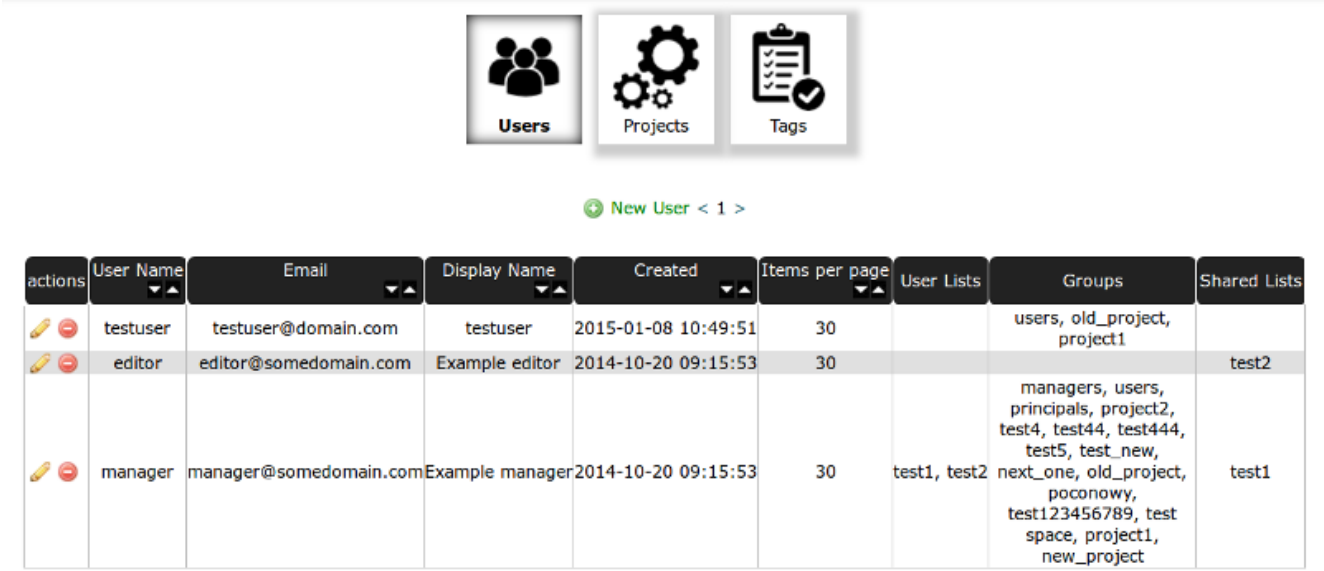
You can create **New Project** by clicking Green Plus button on the top of the table. The Name of the Project and at least one Cell Line are required.

After Creating new project the Group named as The Project with privileges to access the Project is added to database and is available in Users Table. An user to access the Project should be add to Group named as The project.

You should create at least one project to start working with database!

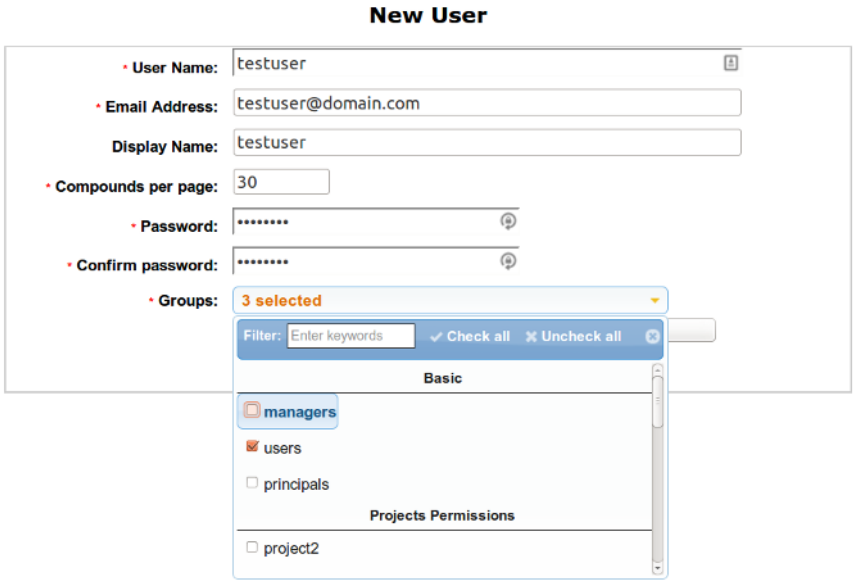
Users Table

In users table you can create/manage users accounts. In *actions* column you can edit (by clicking pencil) or delete a user (delete button).



You can create **New User** by clicking Green Plus button on the top of the table. You should set a password for a user and select Basic Groups and Projects Group.

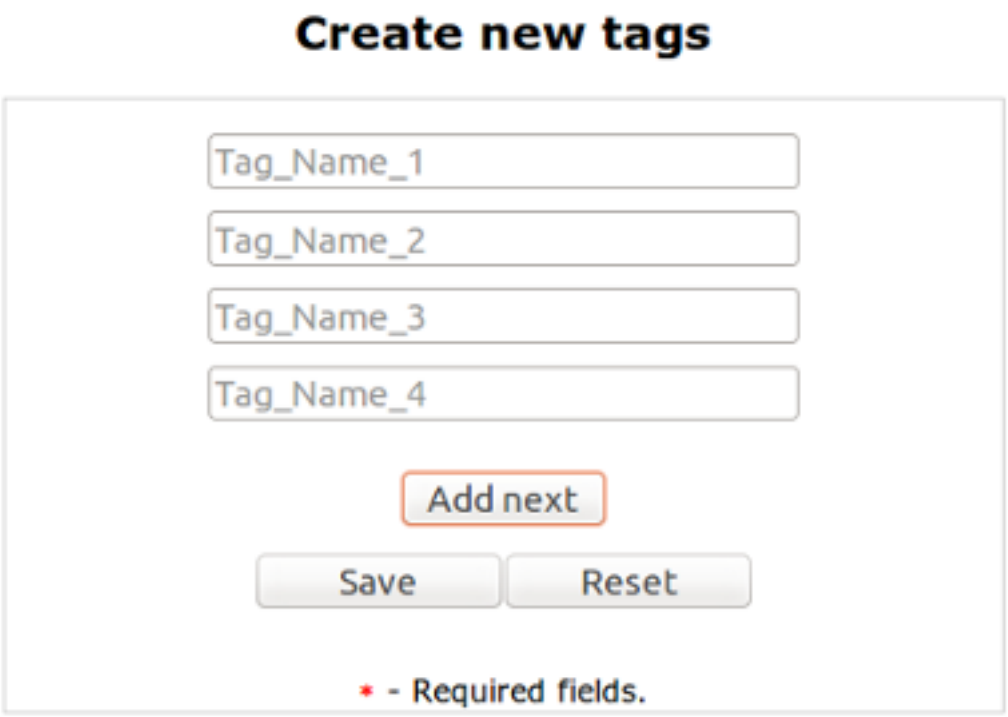
Managers group is for admin privileges, Users group is for basic access and Principals group is for receiving the compounds and notifications. If you want add a user access to any of the projects add it to Group named as the project (Select it from Projects Permissions).



Tags Table

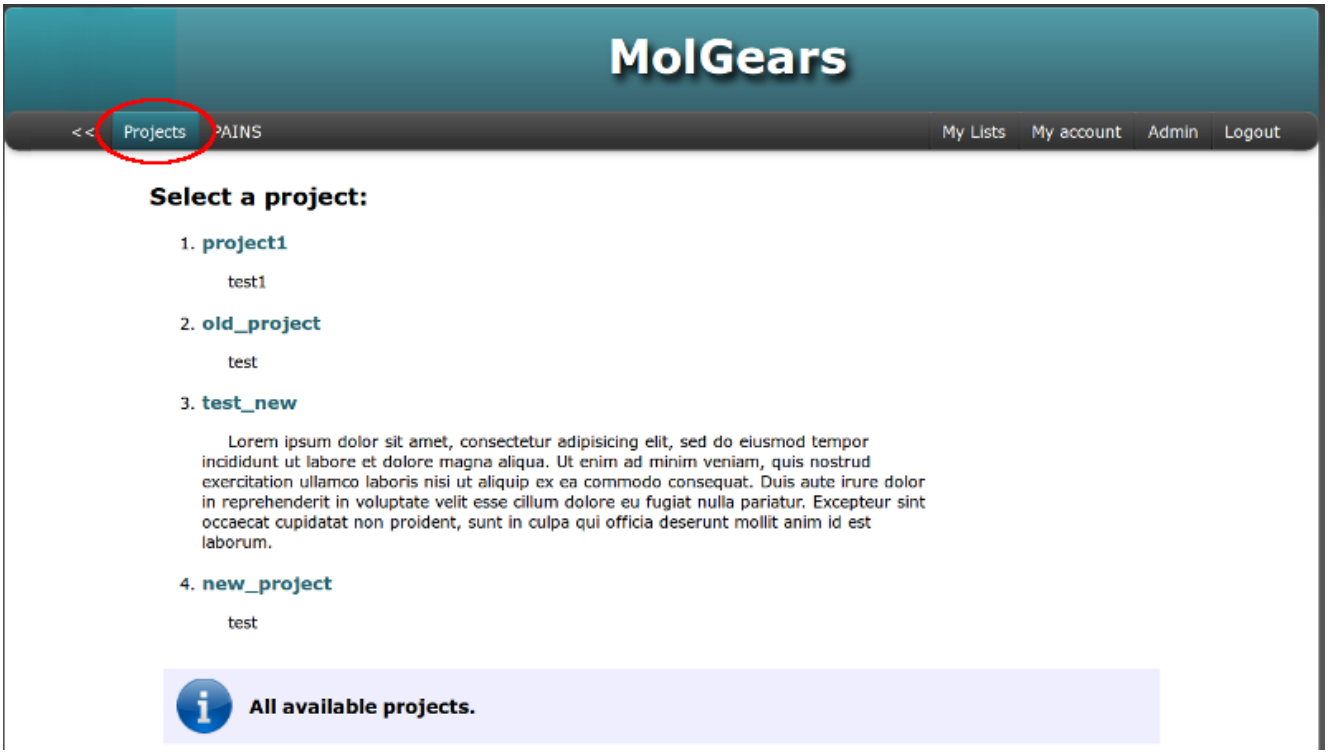
In tags table you can manage/created tags. In *actions* column you can edit (by clicking pencil) or delete a user (delete button).

You can create **New Tag** by clicking Green Plus button on the top of the table. You can add one or more tags at once.



Project View

The default view for user after logging in is a list of available projects. It's also available from the top horizontal menu. User has permission to view and to access to the project after assigning it to a group named as the project. Lets select one of the projects:

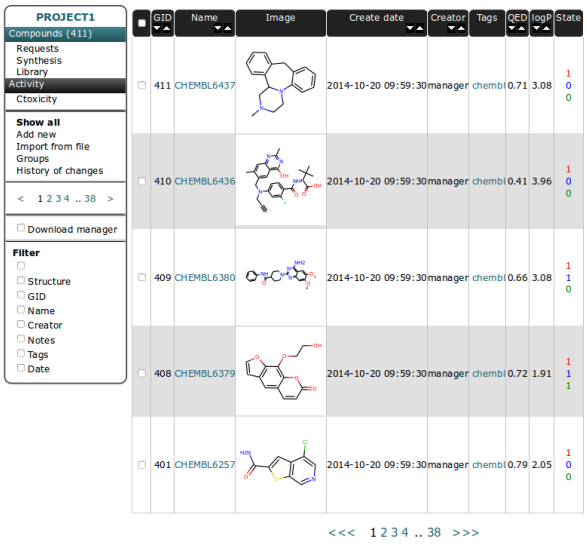


The default view of project is presented below. On the left side of page is placed menu for tables corresponding to the *Project basic workflow*

Compounds Table

The Basic functionality of this table is to store structure information for each unique compound. Other tables have connection to structure by relationship to compounds. See: *Database model*.

Interface of other tables is very similar so we will use this table as an example and describe it in more detail. Interface is composed of two parts. Left vertical menu and table with sortable columns containing information and image of compound.



Left vertical menu

Left vertical menu is divided into parts by horizontal lines.

1. The first part is the same as in the Project View except that it has been market the current table with the number of elements in brackets.
2. The second part is for an actions performed on the table.

    - The default is to show all elements.
    - **Add new** is a form for adding new compound to database. SMILES, Name and Tags (see: *Admin Panel* for tags creation) are required.

      SMILES code should be placed in the appropriate field manually or can be drawn using the JSME editor , which will run when you press the "Draw Structure" button.

      Naming can be autonumerated (see more below in Groups) by selecting next number from selection form near the Name field.

      Option *Isomer* is setting flag "Isomer" on Compound and it's allowing to add the same structure to database once more. By default Isomer is set to "No" and only unique structures are allowed.

      Option PAINS is set for filtering "frequent hitter" compounds and for setting a flag near a compound matched the filters.

    -  **Import from file** option if for reading one or more compounds from file. Allowed file formats are sdf, mol, and smi (text file containing in each row SMILES code and the name of the compound separated by a space - remember to remove empty lines).

    -  **Groups** - is for autonaming of compounds. It's allow to create group of names autonumerated from database. The number is placed between prefix and sufix field with defined precision.

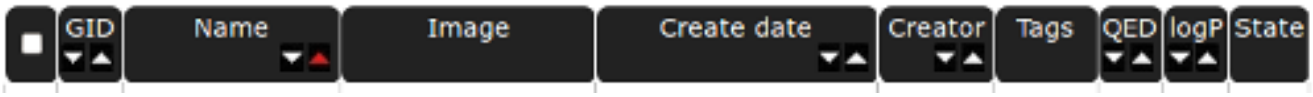
 To create *New Group* click green plus button (*Add new*) and fill the form:

3. Next part is containing navigation bar for switching the page. The same navigation bar is on the bottom of the table.
4. Checking the checkbox "Download manager" will show div with downloading options on the top of the table.
    - You can choose the compounds for download
        - total - all compounds from the table,
        - selected - select compound by checking the checkbox near chosen molecules in the table - it will change the row color,
        - or choose a range from ... to ...
    - Select the file format (PDF, MS Excel, SDF, CSV - data separated by comma, TXT - data separated by space)
    - Choose Information, Attributes or Data to download. In Data Column you can choose the image size in pixels.

5. Choose one of filtering option by checking the appropriate box in Filter section.
    - For all options at once choose blank checkbox. Filter options will show on the top of the table.
    - Options for structure filtering are: Similarity, Substructure, Exact Structure.
    - For filtering the similarity is shown an additional column containing the percentage value of the similarity.
    - For text fields (e.g. Name, Creator, Notes) you can use wildcard character asterisk (*) in your search criteria.
    - Clicking the image of compound in the table will run similarity search with the chosen structure as the search criterion.
    - Clicking the tag name from Tags column in the table will search all compound containing the chosen tag.

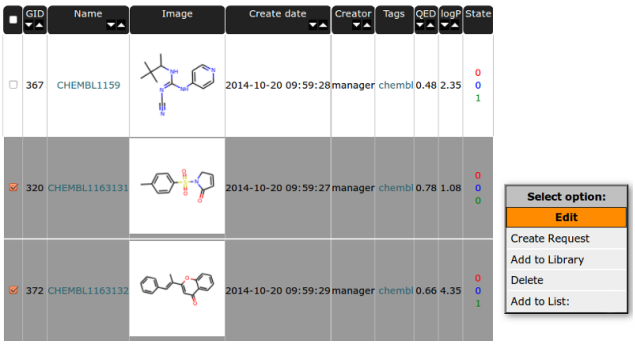
Table View

Some columns in Table header heve arrows for sorting in increasing/decreasing order. The firtst column is for checking all compound from current page. GID - is Global ID - unique number for indexing compounds and it is connected to the compound structure.



Select/unselect compounds independently by checking the checkbox near chosen molecule in the table - it will change the row color.

You can use the the **right mouse button** to display a menu of tasks or options for selected compounds:

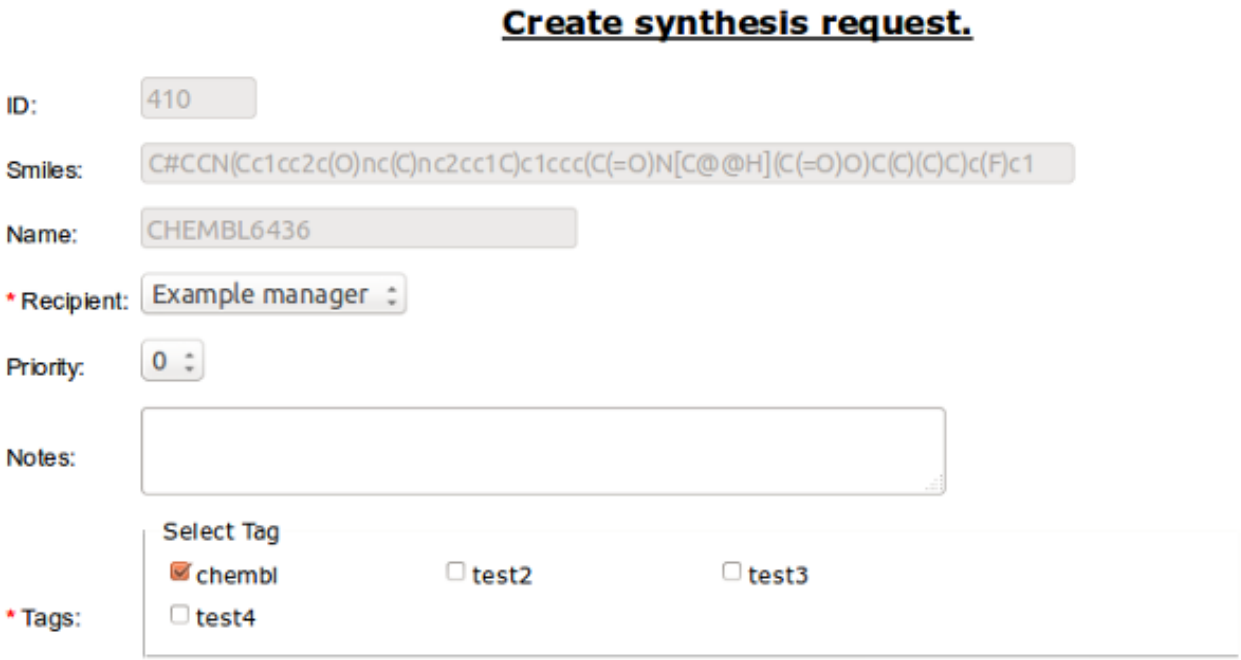


- Edit - is for editing compound data. You can change the structure (it will change the structure for all related instances in requests, synthesis, library and results tables). Since multiple names are allowed for one compound you can add more names to compound here and set the chosen one as a primary.

- Create request - is connected to *Project basic workflow*. It allow to add compound to Requests Table [**ref**].

  Required is selection of the Recipient. This person will be set as a the one to receive email notification about finish of synthesis.

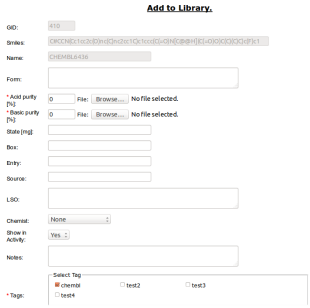
  You can also set the priority of the synthesis (0-5).



- Add to library - is connected to second path in *Project basic workflow*. It allow to add compound to Library Table [**ref**].

  Required is percentage value for one of the purity types (acid or basic in range from 0 to 100). You can optionally attach file with analytical report (e.g. HPLC/MS).

  State is float value for amount (in micrograms) of compound.



Details View

The **Details of Compound** can be viewed after clicking "Name" field in table (in this example CHEMBL6380).

The upper part of the page is the same for all tables and it is divided into two columns:

- The right side shows a compound image (click image to filter compounds by similarity).
- The left indicates the name and GID number in the first row. Than the following information about the descriptors by default colored black. The values are colored red if they exceed

Lipinski rule of five. Toggle show/hide button is for Formula, InChI and SMILES data. The last information are Tags assigned for the compound.

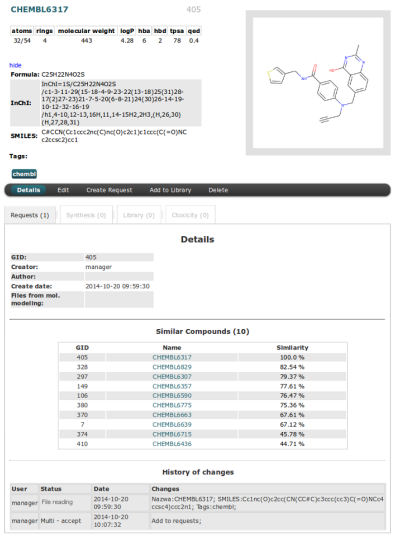
The second part is separated from the first by horizontal menu with actions buttons (actions are the same as in **right mouse button menu**).

This view is different in each table. You can switch between tables by using bookmarks. This place is also good for checking if compound is added to other tables. If not it's bookmark is blocked and font is colored by gray.

Details section has all information about instance from the current table.

Similar Compounds section is only for details in *Compound table*. Mouse Hover on Similarity value will show molecule image. Clicking on the name will redirect to the details page of selected molecule. You can

History of changes section will show all changes and actions recorded for molecule. Data are sorted by date.



Requests Table

This table is a waiting list for compounds to be accepted to synthesis. Each instance in table can have one of three statuses:

- proposed

  This is the default status for molecule in request table after being created. By default only "proposed" molecules are shown in Request Table. To view all instances choose option "*show all*" in left vertical menu bar. Rows in table for this status are colored in yellow.

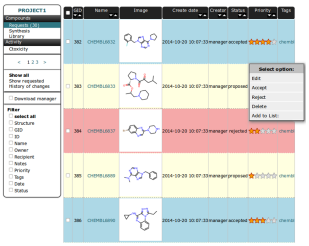
- accepted

  This status is set for molecules accepted to synthesis. See below how to accept a structure to synthesis. Rows in table for this status are colored in cyan.

- reject

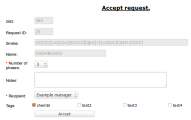
  This status is set for compound is synthesis is canceled for some reason. To reject compound select it from the table and than choose option "Reject" from right mouse botton menu bar.

  Rows in table for this status are colored in magenta.



Synthesis acceptance

To accept selected molecules to synthesis choose option "*Accept*" from **right mouse botton menu bar**.



Required is expected number of synthesis phases/etaps (it can be edited) and Recipient - person who will get notyfication about completion of synthesis and will accept the synthesized compound to library.

> ### *Note*
>
> User to be selected as Recipient should belong to the "*principals*" group. See: *Admin Panel* for editing user groups.

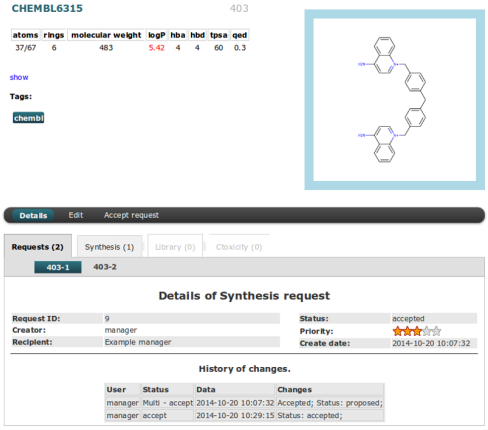
Accepted compound is added to Synthesis table **[ref]**.

> ### *Warning*
>
> Compound will be assigned to the user who accepted it.
>
> For this reason it should be accepted by the person who will be synthesize it.

Details view

The **Details of Requests** can be viewed like in Compounds Table - after clicking "Name" field for chosen molecules in table. Frame around of the compound image is colored in accordance to status type.

If the compound was added to the request more than once, you can switch between them by selecting the next number in the tabs (in this case, 403-1, 403-2 where 403 is a GID number). Current has frame in aqua color.



Synthesis Table

This table is created to store information related to the synthesis process and for tracking synthesis etaps and current status. Priority value is synhronized with request table and can be changed. The phases column present information about current phase vs expected number of phases. By default, only accepted by the user compounds are shown. To switch view to "all" choose option "Show all" from left vertical menu.

Each instance in table can have one of statuses:

- pending

  This is the default status for molecule in synthesis table after being accepted from requests. Rows in table for this status are colored in yellow. For this Status the -1 Phase is set.

- synthesis

  Status set for the compounds in the process of synthesis. Phases for this status are numbered from 0 to max number declared during acceptance from requests. Rows in table for this status are colored in cyan.

- finished

  Status set for the compounds after process of synthesis is finished (current phase number is the same as max number). Analytical data should be added to molecule. Rows in table for this status are colored in orange.

- received

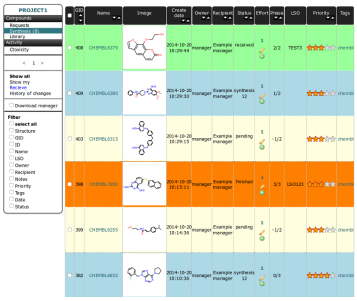
  Status set for the compounds after being recived by Recipient. As a result compound is added to Library table. Rows in table for this status are colored in green.

- rejected

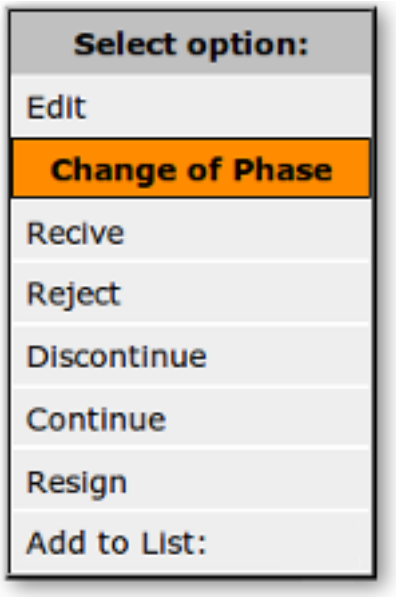
  Status set for the compounds if we decided to not continue synthesis any more. When the molecule is rejected this status is set also for request with it originates. Rows in table for this status are colored in magenta. This change is permanent.

- discontinued

  Status set for the compounds if we temporary decided to not continue synthesis. Previous status can be restored by choosing "continue" option from RMB menu. Rows in table for this status are colored in violet.



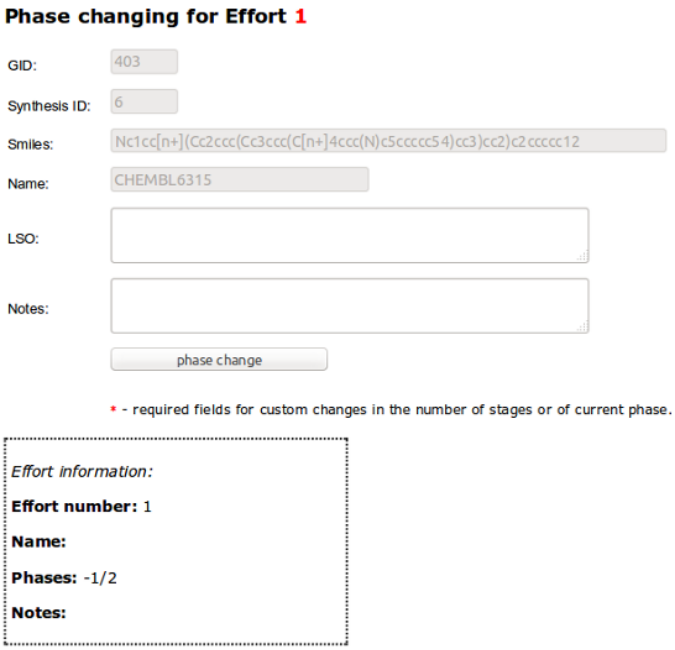
Available actions for synthesis table in RMB menu:



Phase changing

Each molecule can have many synthesis path. Each path is represented by efford record. The number of added efforts is represented in Effort column in Synthesis Table. Click this number to view all efforts list or add new:

To change the Phase for compound choose "Change of phase" in RMB menu and than optionally add LSO number and Notes:



Indices and tables

- *genindex*
- *modindex*
- *search*

Indices and tables

- *genindex*
- *modindex*
- *search*