

GSRS 3.1.2 Release Notes (Draft)

May 2025

Table Contents

[Overview 2](#_Toc461277694)

[Highlighted New Features and Improvements 3](#_Toc1390012555)

[Spring Boot upgrade 3](#_Toc1727645858)

[HOCON config changes 3](#_Toc908688406)

[Cross entity search 4](#_Toc686735147)

[Structure search improvements 4](#_Toc1623892128)

[Nitrosamines flow chart 5](#_Toc1504825007)

[Product module Improvements 5](#_Toc670656752)

[SSG1 Improvements 5](#_Toc224979280)

[Improvement: 6](#_Toc1399073116)

[Other selected Improvements and bug fixes: 7](#_Toc2113738505)

[Improvement: A new Exporter: SQL Exporter 8](#_Toc55837281)

[Improvement: ScheduledExportTask 9](#_Toc998989441)

[Improvement: Two InChIKeys for racemates and epimers 11](#_Toc1125968784)

[Improvement: Show changes between two adjacent versions a substance 12](#_Toc11771479)

[Improvement: PDF download option 13](#_Toc237414455)

[Improvement: 14](#_Toc746015794)

[Improvement: 14](#_Toc1651643668)

[Improvement: 14](#_Toc1107288087)

[New Feature: 14](#_Toc1377515261)

[New Feature: 14](#_Toc894301784)

[Bug fix: 15](#_Toc685070747)

[Bug fix: 16](#_Toc1536032578)

# Overview

GSRS 3.1.2 software release is a minor release.

**Highlighted new features include:**

* Spring Boot upgrade
* HOCON config changes
* Cross entity search
* Structure search improvements
* Nitrosamines flow chart
* Product module
* SSG1

**Highlighted improvements and bug fixes include:**



# Highlighted New Features and Improvements

## Spring Boot upgrade

**Purpose and Motivation:**

**Notes for the future:**

## HOCON config changes

**Purpose and Motivation:**

**How it works:**

**Notes for the Future:**

## Cross entity search

**Purpose and Motivation:**

**How it works:**

**Notes for the future:**

## Structure search improvements

We have added a new type of search: ‘Exact Plus.’

**Purpose and Motivation:**

This is a salt-stripped version of the existing Exact structure search.

**How it works:**

An Exact Plus search takes the input query molecule and removes any known salt or solvate fragment. (The list of salt and solvate fragments is configurable so you may add your own counterions or solvate structures.)

We then perform an exact search using the salt-stripped query molecule.

Note: if the salt-stripping process results in an empty structure, we revert to the input query structure. (An empty structure results from, for example, sodium tartrate, both of whose fragments are on the default salt list.)

**Notes for the Future:**

We are open to your feedback on the behavior of our structure searching options!

## Nitrosamines flow chart

**Purpose and Motivation:**

**How it works:**

**Notes for the Future:**

## Product module Improvements

**Purpose and Motivation:**

**How it works:**

**Notes for the future:**

## SSG1 Improvements

**Purpose and Motivation:**

**How it works:**

**Notes for the Future:**

## Improvement:

**Purpose and Motivation:**

**How it works:**

**Notes for the Future:**

# Other selected Improvements and bug fixes:

## Improvement: A new Exporter: SQL Exporter

**Purpose and Motivation:**

This exporter provides the ability to use a SQL statement to drive a data export.

You can add multiple instances of this exporter, each configured with a different SQL statement. When the individual instance is selected, the SQL is run to fill a data file that is then available for download.

**How it works:**

Most exporters create a data file based on the current results set (generally used after a search). This exporter runs an SQL query over the *entire* database. It may be confusing to users that a recently-performed search has no bearing on the data extracted by the SQL Exporter!

Example configuration:

ix.ginas.export.exporterfactories.substances.list.SQLExporterFactory1 = {  
 "exporterFactoryClass": "gsrs.module.substance.exporters.SQLExporterFactory",  
 "parameters": {  
 "format": {  
 "extension": "dnames.xlsx",  
 "displayName": "Display Names (xlsx) File"  
 },  
 files: [  
 {  
 "name":"Names",  
 "format":"PostgreSQLCsv",  
 "header":"UNII,NAME",  
 "sql":"SELECT S.APPROVAL\_ID AS UNII, COALESCE(N.FULL\_NAME, N.NAME) AS NAME FROM IX\_GINAS\_SUBSTANCES S LEFT JOIN IX\_GINAS\_NAME N ON S.UUID = N.OWNER\_UUID AND N.DISPLAY\_NAME = '1'"  
 },  
 {  
 "name":"CAS",  
 "format":"PostgreSQLCsv",  
 "header":"UNII,CAS",  
 "sql":"SELECT S.APPROVAL\_ID AS UNII, C.CODE AS CAS FROM IX\_GINAS\_SUBSTANCES S LEFT JOIN IX\_GINAS\_CODE C ON S.UUID = C.OWNER\_UUID AND C.CODE\_SYSTEM = 'CAS' AND C.TYPE = 'PRIMARY'"  
 }  
 ]  
 }  
}

Note: that the PostgresQLCsv format can be used with other RDBMS!

**Notes for the Future:**

The team may look for an alternative way to run SQL Exporter and similar exporters that do not use the current result set.

## Improvement: ScheduledExportTask

**Purpose and Motivation:**

This scheduled task provides a way to run a report on a regular basis. The report uses a configured GSRS API query to find records in the database, write a pre-configured list of fields for these records to a file and then send the data to various destinations:

* A file on disk
* Email recipients
  + Data can be sent as an attachment or in the body of the email message
* Ftp servers

**How it works:**  
You add one or more instances of this task to your configuration (for the substance service) and the task becomes available to admins in the Scheduled Jobs tab of the admin panel.

Sample configuration:

gsrs.scheduled-tasks.list.ScheduledExportTask1 = {

"scheduledTaskClass" : "gsrs.module.substance.tasks.ScheduledExportTask",

"parameters" : {

"cron": "0 10 1 \* \* ?",

"autorun": false,

"description": "Scheduled export of substances edited over the past month",

"extension": "csv",

"query": "root\_codes\_codeSystem:\"^FDA UNII$\" AND root\_lastEdited:[P1M TO 10E50]",

"filenameTemplate": "auto-export-$DATE$",

"publicOnly": false,

"username": "ADMIN",

"parameters": {},

"destinations": [

{

"uri": "file:///home/srs/exports/export.csv"

},

{

"uri":"sftp://target\_server/inbox/last\_month\_edited.csv",

"user":"username",

"password":"PASSWORD",

"userDirIsRoot":"false",

"strictHostKeyChecking":"no",

"sessionTimeoutMillis":"10000"

},

{

"uri":"ftp://target\_server\_2/gsrs\_exports/",

"user":"username",

"password":"PASSWORD",

"userDirIsRoot":"true",

"pasiveMode":"true"

},

{

"uri": "smtp://mail.server.org:25",

"from": "gsrs@server.org",

"to": "some.user@server.org",

"subject": "Substances edited last month",

"body": "<h1>Substances</h1></br><table><tr><th>Approval ID</th><th>Display Name</th></tr>",

"recordTemplate": "<tr><td><a href='https://gsrs.ncats.nih.gov/ginas/app/ui/substances/{0}'>{1}</a></td><td>{2}</td></tr>",

"footer": "</table>",

"charset": "utf-8",

"maxSize": "10M"

},

{

"uri": "smtp://mail.server.org:25/last\_month\_edited.csv",

"from": "gsrs@server.org",

"to": "some.user@server.org",

"subject": "Substances edited last month",

"body": "Substances edited last month",

"maxSize": "10M"

}

]

}

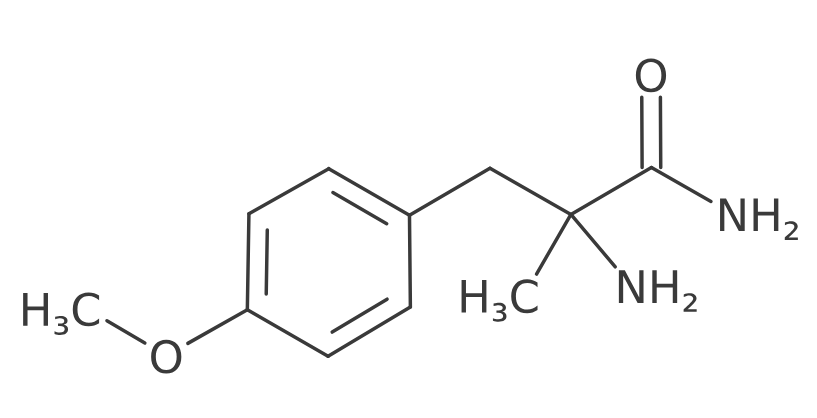
}

## Improvement: Two InChIKeys for racemates and epimers

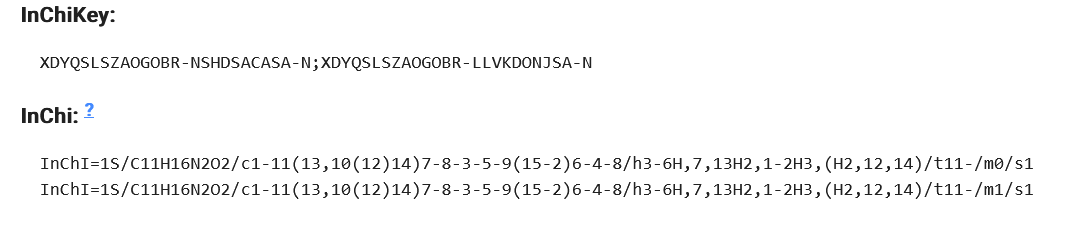
**Purpose and Motivation:**

Some chemical species can we represented by more than one structure and therefore have more than one InChI and InChIKey. in this improvement, we provide a pair of InChIs and InChIKeys for those species designated as *racemic* or *epimeric*.

For example, consider 2-Amino-3-(4-methoxyphenyl)-2-methylpropanamide

The above structure has no stereochemical markings at the chiral carbon and is marked as a mixture of two enantiomers.

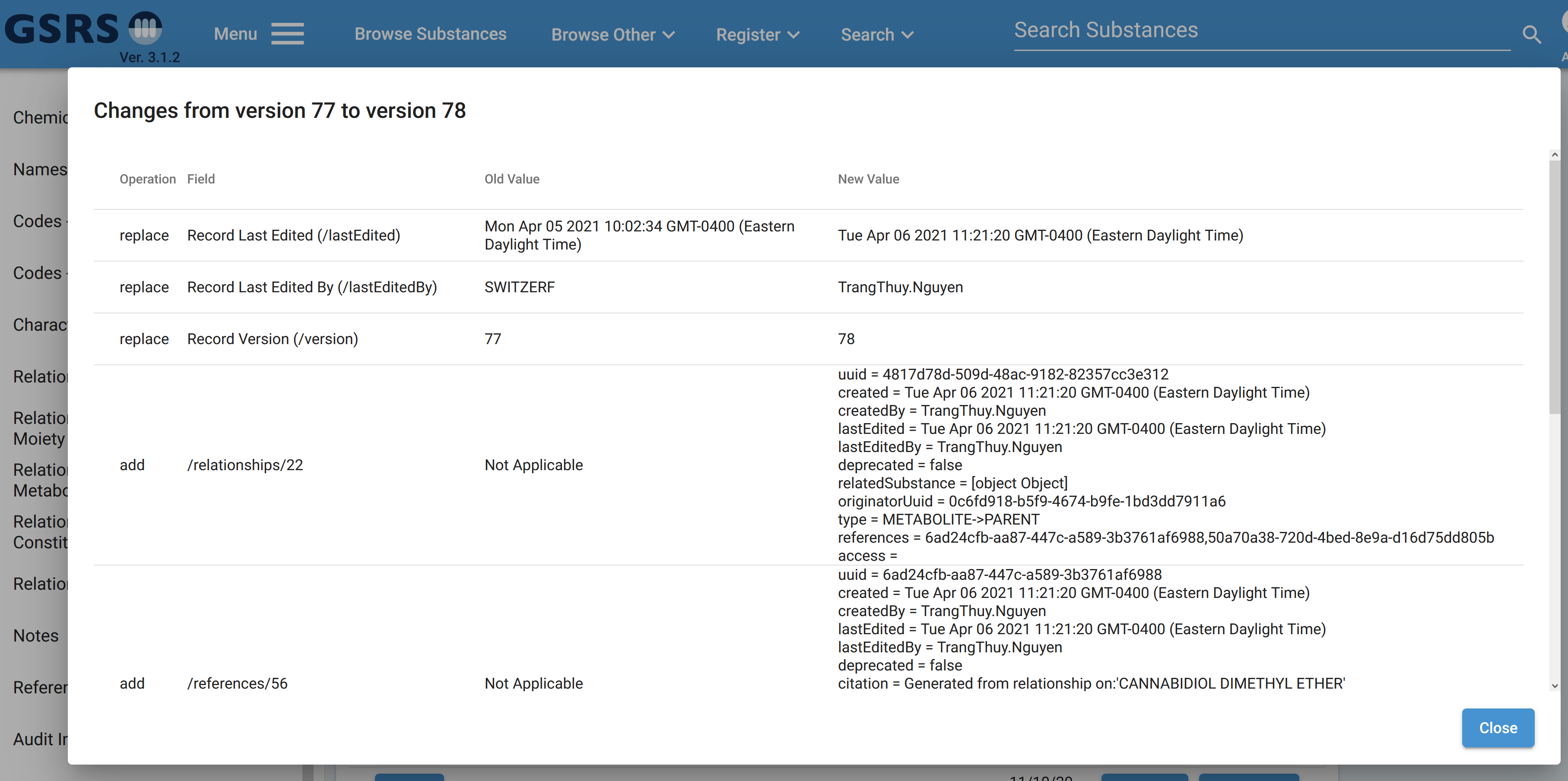
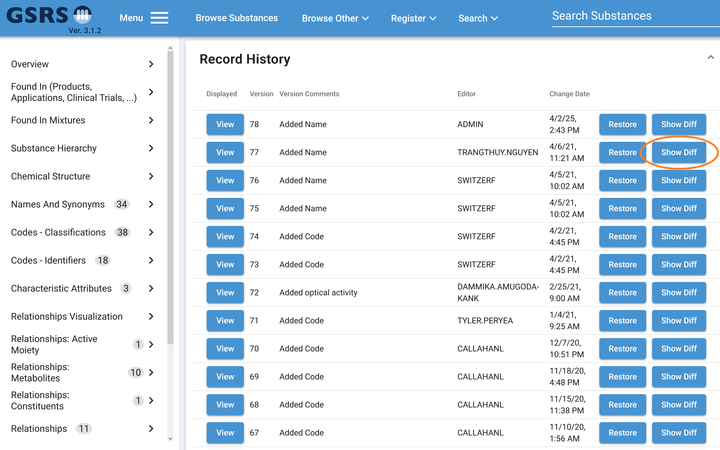
This version of GSRS displays 2 InChIs and 2 InChIKeys:



**How it works:**   
Behind the scenes. GSRS generates both enantiomers and calculates the InChI and InChIKey of each.

## Improvement: Show changes between two adjacent versions a substance

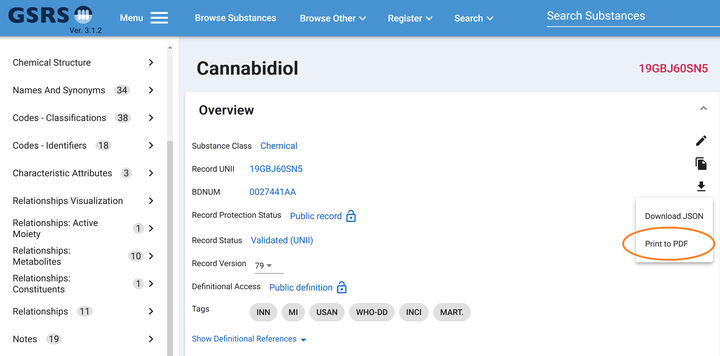
**Purpose and Motivation:**   
  
**How it works:**



## Improvement: PDF download option

**Purpose and Motivation:**

**How it works:**



## Improvement:

**Purpose and Motivation:**

**How it works:**

Improvement:

**Purpose and Motivation:**

**How it works:**

## Improvement:

## New Feature:

**Purpose and Motivation:**

**How it works:**

**Notes for the Future:**

## New Feature:

**Purpose and Motivation:**

**How it works:**

**Notes for the Future:**

## Bug fix:

**Pasting certain SMILES into a GSRS structure box resulted in an improbably structure**

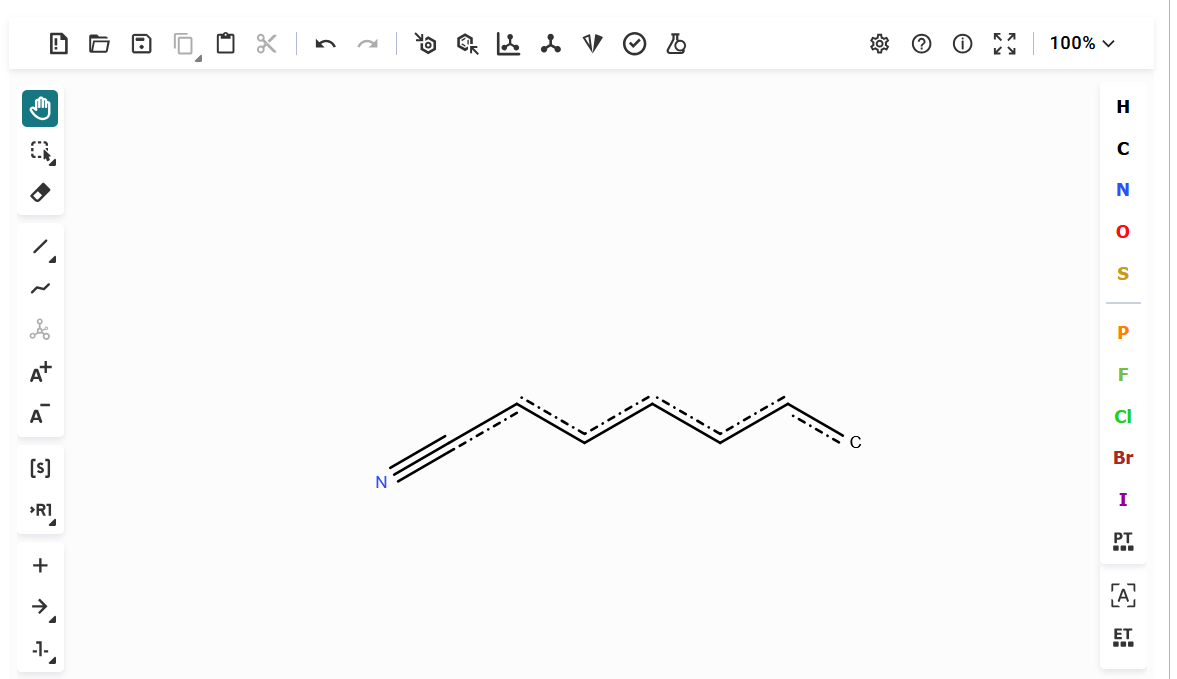
**Purpose and Motivation:**

When SMILES strings containing a pound sign (‘#,’ symbolic of a triple bond) were pasted into a GSRS structure search or registration box, a structure was generated with most bonds given the ‘single or aromatic’ type, reserved for query. This was probably different from the user’s intention.

Example:

SMILES: CCCCCCC#N (heptanenitrile)

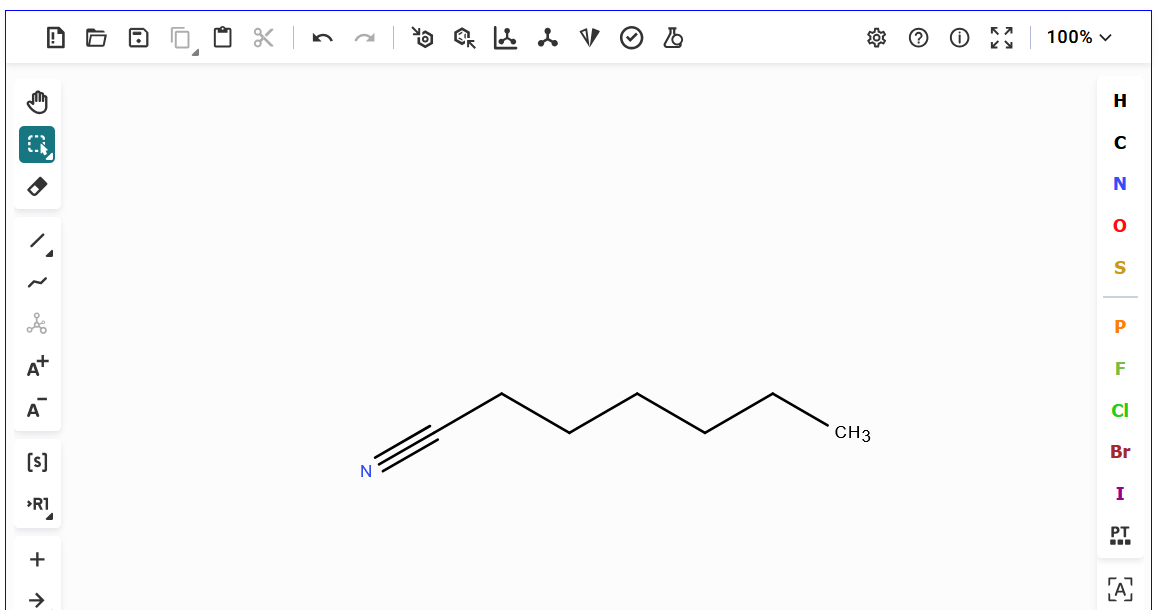
Result:



**How it works:**

The issue was caused by molwitch-cdk, one of our libraries, interpreting ‘#’ as indicating that input string was SMARTS.

With this fix, we now perceive bond types correctly.



## Bug fix:

**Purpose and Motivation:**

**How it works:**