Substance Registration Tasks in the GSRS REST API

|  |  |  |
| --- | --- | --- |
| Version | Date | Comments |
|  |  |  |
| 1.0 | 9 October 2024 | Officially released |
| 1.1 | 29 May 2025 | Added new endpoint |

[Purpose of this document 1](#_Toc1959672971)

[Note on notation in this document 1](#_Toc45338518)

[API Information 1](#_Toc1371108240)

[Substance References 6](#_Toc681815441)

Purpose of this document

To give the software developers information on which API calls they need to make in order to create new substances and to make changes to existing substances. This document focuses on single record operations. A separate document will explain how to import a file of multiple substances.

# Note on notation in this document

The general format of API URLs is:

[http://localhost:8080/api/v1/vocabularies/search?q=root\_domain:%22^DOCUMENT\_TYPE$%22](http://localhost:8080/api/v1/vocabularies/search?q=root_domain:%22%5eDOCUMENT_TYPE$%22)

[https://gsrs.ncats.nih.gov/api/v1/vocabularies/search?q=root\_domain:%22^DOCUMENT\_TYPE$%22](https://gsrs.ncats.nih.gov/api/v1/vocabularies/search?q=root_domain:%22%5eDOCUMENT_TYPE$%22)

* http may be replaced by https on your system.
* ‘localhost’ will likely be replaced by the name of your server.
* ‘8080’ will be replaced by the port number of your substance service. When the port number is 80 or 443 (for https), it will probably be omitted.)
* ‘api/v1/’ will probably not change from one system to another.

# API Information

1. Creating a new substance
   1. JSON
      1. The key to the creation of a new substance is the JSON document you are submitting.
      2. Reference the data model documentation
         1. <https://gsrs.ncats.nih.gov/downloads/GSRS_data_dictionary_2021_12_08.xlsx>
         2. <https://gsrs.ncats.nih.gov/downloads/GSRS_schema_2_0_0.json>
      3. A JSON object for substance creation must contain:
         1. A name object (at least one; additional names are allowed)
         2. A reference object (at least one; additional references are allowed)
         3. Definitional section
            1. This varies by type of substance.

|  |  |  |
| --- | --- | --- |
| **Substance Type** | **Defining JSON section** | **Key fields/collections** |
| Structurally Diverse | "structurallyDiverse" | sourceMaterialClass  sourceMaterialType  organismFamily  organismGenus  organismSpecies  organismAuthor  part (collection)  sourceMaterialClass  sourceMaterialType  part (collection)  fractionName |
| Protein | “protein” | Subunits – collection. Key part of each collection member is a sequence  disulfideLinks -collection  glycosylation - collection with 3 subcollections: NGlycosylationSites, OGlycosylationSites and CGlycosylationSites |
| Nucleic Acid | “nucleicAcid” | nucleicAcidType  subunits - collection  Sequence  subunitIndex  linkages - collection  linkage  sites  sugars – collection  sugar  sites  Note this is the most complex of substance types |
| Chemical | “structure”  Also has a top-level “moieties” collection | Molfile  smiles  stereochemistry  opticalActivity |
| Polymer | “polymer”  Also has a top-level “moieties” collection | idealizedStructure  monomers  structuralUnits  structure (molfile)  attachmentMap |
| Concept | [none] | No ‘definitional’ part.  Data must include at least one name and one reference, as true for all substances. |
| Mixture | “mixture” | Components – collection of substance references + a type ["MUST\_BE\_PRESENT",  "MAY\_BE\_PRESENT\_ANY\_OF", or  "MAY\_BE\_PRESENT\_ONE\_OF"]  for each  parentSubstance – optional |
| Specified Substance Group 1 | specifiedSubstance | Constituents - collection of substance references |

The following collections are available but usually not mandatory for all substance classes. In other words, you can add zero, one or many of each of these items to your substance JSON before submitting the JSON to GSRS.

|  |  |  |
| --- | --- | --- |
| Collection | Key fields |  |
| Code | Code  codeSystem  type  comments\*  codeText\* |  |
| Name2 | Name  Type (values: cn|bn|cd|sn..)5  displayName (Boolean)  languages (collection) |  |
| Property3 | Name  propertyType (values: CHEMICAL|PHYSICAL|…)  value (average, high, low, nonNumericValue, units) |  |
| Reference4 | doctype  citation  url |  |
| Modifications | 3 subcollections:  agentModifications  structuralModifications  physicalModifications | Applicable to most substance types but not chemicals or concepts |
| Relationships | Type (see RELATIONSHIP\_TYPE CV)  relatedSubstance (pointer to another substances. Refuuid which has the same value as the other substance’s UUID) |  |

1 Note that in GSRS UI, for historical reasons, code text and comments appear switched.

2 You must have at least one name in each substance. You can add as many additional names as you like.

3 When you create a protein, the system looks for properties with names “MOL\_WEIGHT:NUMBER(CALCULATED)” and “Molecular Formula’ and will create them if they don’t already exist.

4 You must have at least one reference in each substance. You can add as many additional references as you like.

5 types of names:

cn – common name

bn – brand name

cd – code

sn – systematic name

on – official name

The list of fields in the above table is not meant to be complete but rather to give you an idea of the key items. Similarly, the values given are thought to be the most common, not the only ones allowed.

* + 1. You’ll need to refer to controlled vocabularies to get the allowed values for several of the fields in the data model.
       1. For example, when creating a reference (required as part of all substances), the docType field corresponds to the DOCUMENT\_TYPE vocabulary, which can be retrieved using a URL such as

http://localhost:8080/api/v1/vocabularies/search?q=root\_domain:%22^DOCUMENT\_TYPE$%22

* + - 1. The Data dictionary documentation (in Excel format, as of this writing) lists the ‘CV Domain’ associated with fields in the data model.
      2. The general format of vocabulary-retrieval URLs is

http://<server-name>:<port>/api/v1/vocabularies/search?q=root\_domain:%22^<vocab name>$%22

<server-name> is the name of your GSRS server running the substance service

<port> port on which the substance service is listening.

<vocab name> the specific category of controlled vocabulary you are looking for.

* 1. POST to GSRS
     1. The URL has this form

http://<server>:<substance service port>/api/v1/substances/

* + - 1. Include credentials (within headers)

|  |  |
| --- | --- |
| Header name | Explanation of value |
| auth-username | GSRS username |
| auth-password | Password (use this only in isolated, development systems) |
| auth-key | API key for current user (use this instead of auth-password) |

* 1. Validation
     1. This is also a POST but does not change the contents of the database
     2. Validation checks that a given substance record obeys the configured business rules. (Substance business rules are specific to substance type and business rules can be configured differently on any induvial server. Because some validators search the contents of the database – for example, to confirm that substance names are unique—you can see different results for a given JSON record across multiple GSRS installations.)
     3. The URL for validation has this form:

http://<server>:<substance service port>/api/v1/substances/@validate

* + - 1. Include credentials (within headers as above)

The endpoint

<http://localhost:8081/gsrs/app/api/v1/substances/interpretStructure?mode=&standardize=>

can create SRUs for a polymer as well as moieties for a chemical

The endpoint accepts POST requests where the body is a molfile.

The standardize parameter can take a single value or comma-delimited set of these values or be null

* REMOVE\_HYDROGENS
* ADD\_HYDROGENS
* STEREO\_FLATTEN
* CLEAN

1. Editing an existing substance
   1. Take an existing substance’s JSON, make the changes you need to make (for example, add a name, remove a code, change the citation of a reference, etc.) and PUT the JSON to http://localhost:8080/api/v1/substances
   2. **Do not increment the version field**; the GSRS API will do that for you.
   3. In certain cases, you may want to update an existing substance without performing a validation of the updated data. Note that updating without validation is something to use very sparingly, when you are certain that the update obeys all business rules! In these limited cases, you can PUT the JSON to <http://localhost:8080/api/v1/substances/novalid>
      1. This endpoint is restricted to users with the ‘SuperUpdate’ role.
      2. Note that if you use this endpoint with an updated structure, the structure properties may be incorrect.
2. Deleting a substance.
   1. Note: it is not possible to delete a substance in the GSRS application and, although the API allows deletion, deletion of an existing substance should be thought through before proceeding. Make sure to follow your organization’s data handling policies.
   2. An alternative to deletion is marking a substance as deprecated – retained in the database but hidden in the user interface.

## Substance References

The GSRS data model makes use of Substance References in several places:

* Components in a mixture
* Parent substance for Structurally Diverse, polymer, mixture, etc.
* Structural Modifications

Within substance JSON, a substance reference looks like this:

"substance": {

"deprecated": false,

"uuid": "cc9fa39b-f985-4202-9383-4c60ac66d23b",

"refPname": "3-Fluorocyclobutanamine, trans-",

"refuuid": "a89e67df-17d3-4f9c-9a2e-aa728767a79d",

"substanceClass": "reference",

"approvalID": "5X99FNC2UU",

"name": "3-Fluorocyclobutanamine, trans-",

"linkingID": "5X99FNC2UU",

"references": [],

"access": []

},