

Tips, Tricks, and Time-savers

WORKING ENVIRONMENT	2
ENLARGE FIELDS.....	2
DATE FORMAT	2
HOW TO FIND THE REST API NAME OF A TYPE OR ITS PROPERTY	2
WHAT ROLE IS NEEDED TO RUN REST API COMMANDS?	3
DEVELOPMENT GLOSSARY RIDS.....	4
OPERATORS AND THE DATA TYPES THEY USE	4
QUERIES – GENERAL COMMENTS	4
QUERIES – LARGE OUTPUT.....	5
ASSET CHANGE HISTORY	5
WHAT PROPERTIES ARE QUERY-ONLY?.....	5
FORMULATING QUERIES	6
BUNDLE-DEFINED PROPERTIES.....	6
TRANSLATION OF ASSET BUNDLES.....	7
TRANSLATION OF CUSTOM ATTRIBUTES.....	7
HOW TO DISPLAY NON-ENGLISH CHARACTERS FROM THE URL IN THE REQUEST URL AREA	9
HTTP STATUS CODES	10
JSON TIPS AND SYNTAX	11
REFERENCES FOR REST API.....	11

Tips, Tricks, and Time-savers

Working environment

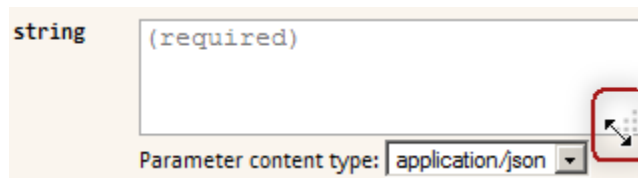
Open two windows in your browser and keep them open while you work:

- IBM InfoSphere Information Governance Catalog
- InfoSphere Information Governance Catalog REST API, and then expand the **GET /TYPES/** area.
Click **Try it out!** The REST name of a type or its property is found in the Response Body area.

Open a third window for InfoSphere Information Catalog REST API. Use this window to do the actual API work.

Enlarge fields

To get a clearer display of some fields, drag the corner of the field to enlarge it.



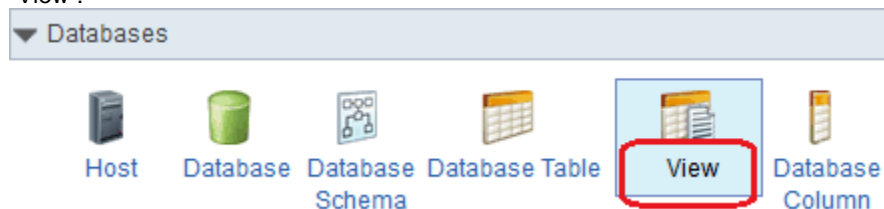
Date format

The date format is yyyy:mm:dd 00:00:00. For example, 2015-10-13T18:23:52. To compare dates, use an external tool to convert the date to Epoch Unix time format.

How to find the REST API name of a type or its property

Find the InfoSphere Information Governance Catalog name within the REST API **GET /types/** section. Do not use the property name that is in Details page of the asset type in InfoSphere Information Governance Catalog.

For example, suppose that the name of the asset type in InfoSphere Information Governance Catalog is "View":

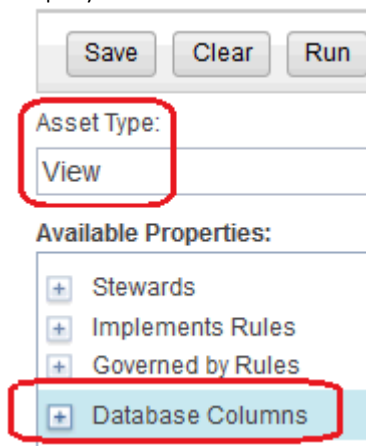


To find the name of this asset type in the REST API, use the **GET /types** command. In the Response Body of the **GET /types** command, find the line "`_name`": View". The string in the "`_id`" field is the name in the REST API.

Tips, Tricks, and Time-savers

```
{
  "_name": "View" This string is the name of the asset type in InfoSphere Information Governance Catalog
  "_plural": "Views",
  "_id": "view", This string is the name of the asset type in REST API
  "_class": "ASCLModel.DatabaseView",
  "_url": "https://localhost:9443/ibm/iis/igc-rest/v1/types/view",
  "_group": "Databases",
}
```

Similarly, to find the REST API name for the database column property in the View asset type, use **GET /types/view** command and set **showViewProperties** to true. In the Response Body, search for "displayName": "<what it says in UI>". In this example, in the Response Body, search for "displayName": "Database Columns".



The screenshot shows a REST client interface. At the top, there are three buttons: 'Save', 'Clear', and 'Run'. Below them is a section labeled 'Asset Type:' with a dropdown menu currently showing 'View'. Underneath is a section titled 'Available Properties:' containing a list of properties with expand/collapse icons: 'Stewards', 'Implements Rules', 'Governed by Rules', and 'Database Columns'. The 'Database Columns' property is highlighted with a red box.

```
"name": "database_columns", This string is the name of the asset type in REST API
  "type": {
    "name": "database_column",
    "url": "https://localhost:9443/ibm/iis/igc-rest/v1/types/database_column"
  },
  "displayName": "Database Columns" This string is the name of the asset type in InfoSphere Information Governance Catalog
```

As another example, the name of an asset type in InfoSphere Information Governance Catalog is "BI Report". To find the REST name of this asset type, go to the window with REST API **GET /TYPES/**. Scroll down to where `_name` is "BI Report". The REST name for the type is the value that is in the `_id` field.

What role is needed to run REST API commands?

Log in to the REST API with the role that is required for the service that you want to run. See the topic [Tasks that each security role can do](#) for what roles are needed.

For example, to run the **GET/search** command, log in as a user with the Information Governance Catalog User role or higher.

Tips, Tricks, and Time-savers

Development glossary RIDs

In InfoSphere Information Governance Catalog, if you are editing glossary assets (categories, terms, information governance policies, information governance rules) and workflow is **on**, any edit calls that you make in the REST API that require a Resource Identifier (RID) must use the RID that is in the development glossary.

To get the development glossary RID, do either of these steps:

- In InfoSphere Information Governance Catalog, go to the **Details** page of the asset in the development glossary, and then copy the RID that is found in the URL. For example, the RID in this URL is in [blue highlight](https://localhost:9443/ibm/iis/igc/#dossierView/6662c0f2.e1b1ec6c.6uhl1j0ja.avfkk4e.jr3iee.c2srh5dnpn12tagecunjkl?bg_req_context={%22perspective%22%3A%22Glossary%22}):
https://localhost:9443/ibm/iis/igc/#dossierView/6662c0f2.e1b1ec6c.6uhl1j0ja.avfkk4e.jr3iee.c2srh5dnpn12tagecunjkl?bg_req_context={%22perspective%22%3A%22Glossary%22}
- In the REST API **GET/search/** command, type the string draft in the **workflowMode** field.

Operators and the data types they use

Operator	Value	Property example
like {0}	String	"labels.name"
like {0}%	String	"name"
like %{0}	String	"assigned_to_terms.name"
like %{0}%	String	"custom_securitylevel"
isNull	no value (N/A)	" assigned_to_terms"
in	Number, Date/Time, or String	"_id"
=	Number, Date/Time, Boolean, or String	"parent_category._id"
containsWords	String	"short_description",
between	Number, Date/Time, or String (Minimum and maximum)	"modified_on"
<, <=, >, >=	Number, Date/Time, or String	"custom_number"
and, or	no value (N/A)	

A condition can be negated by adding "negated" : true

Note that the value true must not be in quotes.

Queries – General comments

Different asset types can have different property names for representation.

Most asset types have 'name' (without the leading underscore), but some have other representation, for example 'full_name'.

Tips, Tricks, and Time-savers

'_name' (with a leading underscore) is a generic representation property of an asset, regardless of its type. It is returned in search and query commands, but if you put conditions on '_name', then undefined results can occur. Instead of using the generic representation, use the type-specific representation property. For example, make the condition on 'name' or 'full_name', not '_name'.

Queries – Large output

If you expect a large response from a query, set the page size to a large number. If you're not drilling down into relationships a lot, you can set the page size in your queries to 1000 or even 5000, to start with. But, requesting all results at once might not scale well in big scenarios.

Instead, you would retrieve query results page by page. In the Result Body area, you can see what the total result number is, and keep re-issuing the same query, each time with an incremented "begin" parameter. Increment "begin" by your page size, until you do not get any more results.

Objects could be added or deleted while you are re-issuing the query. As a result, there might be 'page drifting':

- When objects get deleted while you are paging, you might not get all of the remaining results.
 - When objects get added while you are paging, you might not get the new objects, and you might get some of the objects multiple times.
-

Asset change history

When you want to retrieve details for a specific asset by using **GET/assets/{id}** command, you can decide whether to include asset history in the output or not. By default, history is not included to improve performance. If you are interested in such details, set the includeHistory parameter to true.

What properties are query-only?

A query-only property is a property that can be queried, but is not returned in **GET /types/{id}** even if all View, Edit, or Create fields are turned on.

For example, one query-only property is 'category_path'. This property contains the root category and all subcategories that lead up to the term or category of a query.

For getting terms under a certain category, including under any of its subcategories, use this condition:

```
{
  "property" : "category_path._id",
  "operator" : "=",
  "value" : "My category RID",
}
```

In a big catalog, be sure to put other, more restrictive conditions first. This placement will improve the query performance.

Tips, Tricks, and Time-savers

Formulating Queries

If you want the functionality of queries rather than a ranked search, specify only one type to search on.

When a query is on type 'T' and there is a condition and/or selected property 'P', do not state the property as 'T.P', but rather as 'P'.

For example, use 'assigned_to_terms', and not 'database_table.assigned_to_terms'. To drill deeper, separate each step by using a dot. For example, assigned_to_terms.name.

To find the property type names for REST API, use the **GET /types/** command.

Bundle-defined properties

For properties that are defined by using the bundle API, do not prefix the property name with the type or bundle ID. For example:

```
{
  "types": ["$AllDataTypes2-Project"],
  "properties": ["name", "$steps", "$decimal"],
  "where": {
    "conditions": [{
      "property": "name",
      "operator": "=",
      "value": "TestProj"},
      {
        "property": "$steps",
        "value": 333,
        "operator": "="
      }
    ],
    "operator": "and"
  }
}
```

Note how bundle-defined properties have a leading dollar sign (“\$steps”), but the built-in properties do not (“name”).

The Response will be:

```
{
  "items": [
    {
      "$decimal": 444.3,
      "_type": "$AllDataTypes2-Project",
      "_id": "5b818a0c.187019e4.6vml1s7mq.atq8hb1.hjurvg.jo399oscie2b8pjl4nj",
      "_context": [],
      "_url": "https://localhost:9443/ibm/iis/igc-
```

Tips, Tricks, and Time-savers

```
rest/v1/assets/5b818a0c.187019e4.6vml1s7mq.atq8hb1.hjurvg.jo399oscie2b8pjl4nj
k",
  "_name": "TestProj",
  "$steps": [
    333
  ]
},
"paging": {
  "numTotal": 1,
  "pageSize": 10,
  "end": 0,
  "begin": 0
}
```

Translation of asset bundles

The labels in 'labels.properties', if they exist, override whatever you state in the asset_type_descriptor.xml file.

For example, inDefaultLocale is **A** for label key **tree1**:

```
<tree position="3" rootClassRefs="Project">
  <label key="tree1" inDefaultLocale="A" />
```

The inDefaultLocale string is overridden by the language support in label.properties:

tree1=B

If your asset bundles do not need translation, then delete everything in the asset bundle's i18n directory.

The locale names are a Java standard: <http://www.oracle.com/technetwork/java/javase/javase7locales-334809.html>.

Translation of custom attributes

You can provide translations for custom attributes and their values. Complete the following steps:

1. Find the RID of the custom attribute for which you want to provide translations. Use the GET /administration/attributes/ call.
2. Get the details of the custom attribute. Use the GET /administration/attributes/{id} call.
3. Copy the details of your custom attribute and edit them by adding localeLabels property. The examples of usage are provided after this procedure.
4. Upload new definitions of custom attributes. Use the PUT /administration/attributes/{id} call.

Custom attributes of type text:

```
{
  "id": "721f6090.6acd460b.4gnm9vmde.kkcmlt2.n746m2.q1p1priuncbgfp2krd3c4",
  "name": "text_ca",
```

Tips, Tricks, and Time-savers

```
"description": "text_ca description",
"appliesTo": [
  "term"
],
"localeLabels": [
  {
    "locale": "de",
    "name": "text_ca de",
    "description": "text_ca description de"
  }
]
}
```

Custom attributes of type predefined values:

```
{
  "id": "721f6090.6acd460b.4gnm9vr9j.1sdt5cb.bk3b15.kmb8mn8og8mktre8tj548",
  "name": "predefined_values_ca",
  "description": "Predefined values ca description",
  "appliesTo": [
    "term"
  ],
  "validValues": [
    {
      "name": "One",
      "description": "One Description"
    },
    {
      "name": "Three",
      "description": "Three Description"
    },
    {
      "name": "Two",
      "description": "Two Description"
    }
  ],
  "localeLabels": [ {
    "locale": "de",
    "name": "term number de",
    "description": "term number description de",
    "validValues": [
      {
        "id": "One",
        "name": "Ein",
        "description": "Eine Beschreibung"
      },
      {
        "id": "Three",
        "name": "Drei",
        "description": "Drei Beschreibung"
      },
      {
        "id": "Two",
        "name": "Zwei",
        "description": "Zwei Beschreibung"
      }
    ]
  } ]
}
```

Custom attribute of type relationship:

```
{
  "id": "721f6090.6acd460b.4gnm9vraq.ahmpj0i.3itcqs.cpg7h19qb0tfekja70236",
  "name": "term to category ca",

```


Tips, Tricks, and Time-savers

```
"description": "term to category ca description",
"appliesTo": [
  "term"
],
"inverseName": "category to term",
"targetReferences": [
  "category"
],
"localeLabels": [ {
  "locale": "de",
  "name": "term to category ca de",
  "description": "term to category ca description de",
  "inverseName": "category to term de"
}
]
}
```

How to display non-English characters from the URL in the Request URL area

The REST API **GET /TYPES/** command returns the proper characters in the **Response Body area**, but those characters are garbled when you use the URL that is in the **Request URL area**.

You must specify the character-set parameter to return proper characters for your browser because the browser has a default character encoding. In the URL, add the character-set parameter and specify the language to the URL string.

For example, add the text in blue to display Polish characters properly: <https://localhost:9443/ibm/iis/igc-rest/v1/search/term/?Accept-Charset=utf-8;Accept-Language=pl-PL>

Tips, Tricks, and Time-savers

HTTP Status Codes

Status Code	Text	Description
200	OK	The API command completed successfully.
304	Not Modified	There was no new data to return.
400	Bad Request	The request was invalid or cannot be otherwise served. In API v1.1, requests without authentication are not considered valid and give this response.
401	Unauthorized	Authentication credentials were missing or incorrect.
403	Forbidden	The request was refused or access was denied.
404	Not Found	The URI requested is not valid or the resource requested, for example a user, does not exist.
406	Not Acceptable	Returned by the Search API when a format that is not valid is specified in the request.
429	Too Many Requests	The request cannot be served because the limit for the resource is exceeded.
500	Internal Server Error	The requested format is not supported by the requested method, or a runtime error has occurred. The log files SystemOut.log, SystemError.log, and igc.log contain more information.
502	Bad Gateway	
503	Service Unavailable	The server is up, but overloaded with requests. Try again later.

Tips, Tricks, and Time-savers

JSON tips and syntax

Tips:

Watch your commas, brackets, and all punctuation!

Syntax:

{ } designates a map in the format of "property" : "val"
[] designates an array or list of maps

```
# Begin "body" with the following lines:
#
{
  "pageSize": 999,
  "includeAssetId": true,          ### or false
  "workflowMode": "draft",       ### or blank ("",) or don't include this line
#
# Continue the "body" with the following lines:
#
  "properties": [ "name", "label.name" ], ### list desired properties to display in the result set
  "types": [ "term" ],             ### specify the asset type (or types) that is being queried
  "where": {                       ### "{" indicates the beginning of the "where" clause (search constraint)
    "conditions": [ {              ### "[" indicates multiple conditions; "{" indicates a set of objects required
      ### for a single condition
      "property": "label.name",    ### specify a property (or indirect property) of the asset type listed
      ### in "types" above
      "operator": "=",            ### like {0}; like %{0}%, <=, =>
      "value": "BDW"              ### property value (of the "property" below) that are either
      ### numbers or characters that are delimited by quotation marks
    },                             ### end of first condition
    {                               ### beginning of second condition
      "property": "label",
      "operator": "isNull",       ### boolean operator
      "negated": false            ### when "negated" is false, it is not needed; true value would indicate
      ### "is NOT null"
    },                             ### "]" end of second condition; "]" end of all conditions
    "operator": "or"              ### combination of conditions are either "and" or "or"; if there is only one
      ### condition, use either "operator"
  }
}
}
```

References for REST API

Headers and parameters: <http://www.soapui.org/testing-dojoo/best-practices/understanding-rest-headers-and-parameters.html>

Formatting JSON code: <http://www.jsoneditoronline.org/>.