MatInf API

Application Programming Interface (API) enables external software to interact with MatInf.

Several APIs are planned to be developed, so the information here will be updated as soon as some new functionality becomes available.

Read-only API

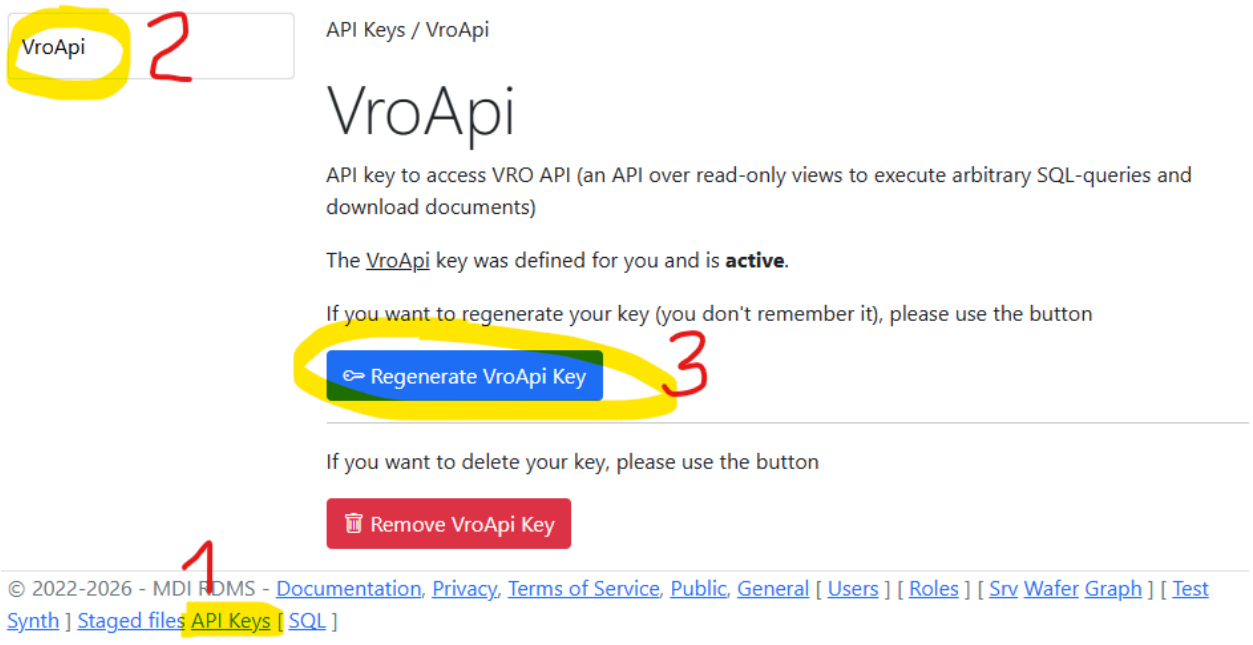
Also known as VRO (Views Read Only) API.

This REST API is designed to read the data flexibly. Arbitrary SQL queries are supported to traverse the database in a desired fashion with document downloads. Prerequisites to use the API are:

- **Tenant_URL** - the URL of the tenant.
- **API_key** - a secure string that must be present in the "*VroApi*" header of every call to the API method (essentially, there should be a user in a database with a claim named "*VroApi*", containing the API key as its value).

Knowing the Tenant URL and the API key, anyone could access the MatInf via any REST HTTP client, e.g. Postman, Telerik Fiddler, CURL, or your own code!

IMPORTANT: If you need to get an API Key, you should use the web interface (API Keys link in the page footer or use relative link /adminsecrets) to generate it. Please do not share your API Key, since it is associated with your account and you are responsible for using it.



Obsolete note: Since administrators can grant access to the API by assigning a claim named "*VroApi*", it's recommended to construct its value (the API key itself) by a template

<e-mail>_<GUID>, for example, “username@domain.org_00000000-0000-0000-0000-000000000000”. GUID generators can be found online (for example, https://www.guidgenerator.com/).

API methods description

Read-only API has version 1 and is available by address <Tenant_URL>/vroapi/v1/<method_name>. It exposes only two methods, which are briefly described here:

- **execute** - **POST** method with a single string parameter ***sql*** - execute arbitrary SQL query on read-only views in the ***vro*** database schema and returns table in JSON format (easily convertible to any other type, e.g. Pandas DataFrame).
- **download** - **GET** method with a single integer parameter ***id*** - download a file/document, associated with a specified *ObjectId* (original file/document name can be found in a ***content-disposition*** header).

Tip: To issue SQL queries please refer to the database structure in the manual, available in your tenant which describes tables and views structure (but since the internal database structure is the same you can use any tenant), e.g. https://mdi.matinf.pro/home/doc.

API client for Python

To facilitate client development here you can find the client code in Python that helps you start using the API (don’t forget to adjust ***tenant_url*** and ***api_key*** with your values).

The current version of the client source code is also available at https://gitlab.ruhr-uni-bochum.de/vic/infproject/-/blob/master/SCRIPTS/VroApi.py?ref_type=heads

Here you can find the auxiliary MatInfWebApiClient class (sorry for formatting issues, please use the link above to have it nicely formatted), which is essentially a client for the REST service, and a short code demonstrating both methods (**execute** and **download**)

==========

```
import requests
import re
import pandas


# The simple REST API client for MatInf VRO API
class MatInfWebApiClient:
    def __init__(self, service_url, api_key):
        self.service_url = service_url + "/vroapi/v1/"
        self.api_key = api_key
        self.file_name=""
```

```
    self.dataframe=None

def getFilename_fromCd(self, cd):
    if not cd:
        return None
    fname = re.findall('filename=(.+)(?:;.+)', cd)
    if len(fname) == 0:
        return None
    self.file_name=fname[0]
    return fname[0]

def get_headers(self):
    return { 'VroApi': self.api_key }

def execute(self, sql):
    headers = self.get_headers()
    data = { 'sql': sql }
    try:
        response = requests.post(self.service_url+"execute", headers=headers, data=data)
        response.raise_for_status()  # Raise exception for non-2xx status codes
        js = response.json()
        self.dataframe = pandas.DataFrame.from_dict(js)
        return js
    except requests.exceptions.RequestException as e:
        print(f"Error: {e}")
        return None

def download(self, id, file_name=None):
    headers = self.get_headers()
    data = { 'id': id }
    try:
        response = requests.get(self.service_url+"download", params=data, headers=headers)
```

```
            response.raise_for_status()  # Raise exception for non-2xx status codes

            self.file_name=file_name
            if not file_name:
                file_name = self.getFilename_fromCd(response.headers.get('content-disposition'))
                #print('extracted file_name: ' + self.file_name)
            open(file_name, 'wb').write(response.content)
            return response

        except requests.exceptions.RequestException as e:
            print(f"Error: {e}")
            return None


# Example usage:
if __name__ == "__main__":
    # initialise service_url (tenant main url)
    tenant_url = "https://..." # tenant_url here

    # initialise api_key (corresponding VroApi claim must be associated with a user in database)
    api_key = "email_key" # your_api_key here

    client = MatInfWebApiClient(tenant_url, api_key)

    result = client.execute("select * from vroTypeInfo")
    if result:
        print("API Response(execute):", result)
        print("API Response(execute): ", client.dataframe)

    result = client.download(43511) # ObjectId
    if result:
        print(f"API Response(download, {client.file_name}):", result)
```

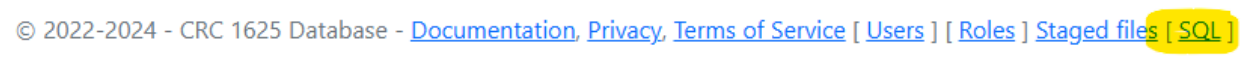
==========

Source Code

For further questions, please refer to the source code first, which is available at https://gitlab.ruhr-uni-bochum.de/vic/infproject. Scripts to use API are available in the **Scripts** folder of the repository (including CURL calls): https://gitlab.ruhr-uni-bochum.de/vic/infproject/-/tree/master/SCRIPTS?ref_type=heads

GUI to execute SQL queries

To access the GUI to perform queries, the user must have the appropriate permission set: the user must belong to one of the groups (User / PowerUser / Administrator) AND (either the user's Claim ***VroUi*** is set to ***1*** or the user belongs to one of the PowerUser / Administrator). If the condition is met, the user will see a ***SQL*** link to the GUI in the footer of the site:



SQL Query Examples

To understand the database structure, please refer to the documentation regarding tables and views description available in any tenant under /home/doc link, e.g., https://mdi.matinf.pro/home/doc

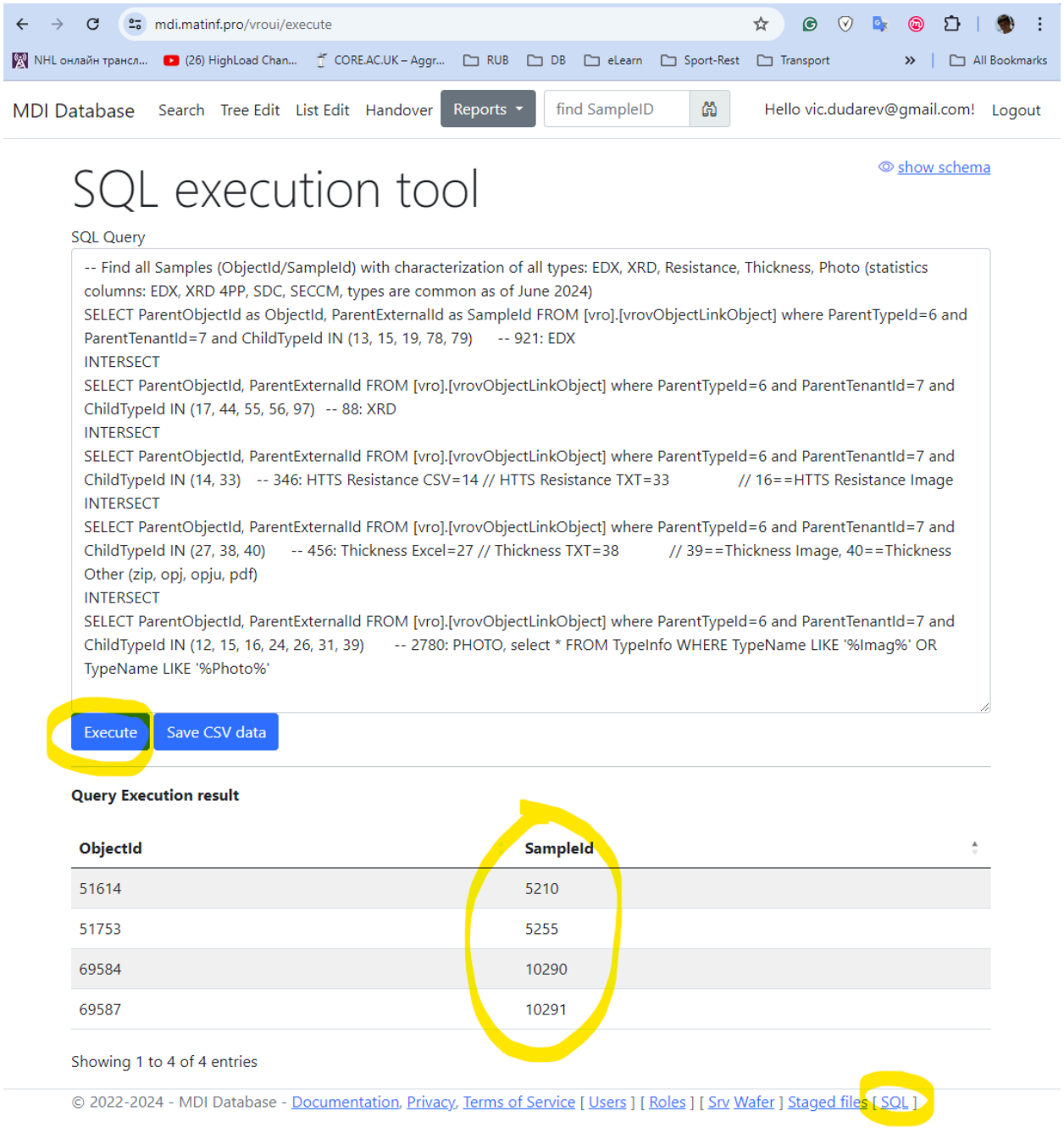
Here are a couple of examples provided to demonstrate tasks that could be solved and corresponding SQL queries.
1) Samples having all the given characterisation

==========

```
-- Find all Samples (ObjectId/SampleId) with characterisation of all types: EDX, XRD, Resistance, Thickness, Photo (statistics columns: EDX, XRD 4PP, SDC, SECCM, types are common as of June 2024)

SELECT ParentObjectId as ObjectId, ParentExternalId as SampleId FROM [vro].[vrovObjectLinkObject] where ParentTypeId=6 and ParentTenantId=7 and ChildTypeId IN (13, 15, 19, 78, 79) -- 921: EDX

INTERSECT

SELECT ParentObjectId, ParentExternalId FROM [vro].[vrovObjectLinkObject] where ParentTypeId=6 and ParentTenantId=7 and ChildTypeId IN (17, 44, 55, 56, 97)      -- 88: XRD

INTERSECT
```

```
SELECT ParentObjectId, ParentExternalId FROM [vro].[vrovObjectLinkObject] where ParentTypeId=6 and
ParentTenantId=7 and ChildTypeId IN (14, 33)          -- 346: HTTS Resistance CSV=14 // HTTS Resistance
TXT=33                // 16==HTTS Resistance Image

INTERSECT

SELECT ParentObjectId, ParentExternalId FROM [vro].[vrovObjectLinkObject] where ParentTypeId=6 and
ParentTenantId=7 and ChildTypeId IN (27, 38, 40)    -- 456: Thickness Excel=27 // Thickness TXT=38
// 39==Thickness Image, 40==Thickness Other (zip, opj, opju, pdf)

INTERSECT

SELECT ParentObjectId, ParentExternalId FROM [vro].[vrovObjectLinkObject] where ParentTypeId=6 and
ParentTenantId=7 and ChildTypeId IN (12, 15, 16, 24, 26, 31, 39)          -- 2780: PHOTO, select * FROM
vroTypeInfo WHERE TypeName LIKE '%Imag%' OR TypeName LIKE '%Photo%'
===============
```

The results could be the following



2) Samples containing Au-Pd-Rh-Pt with their characterization statistics

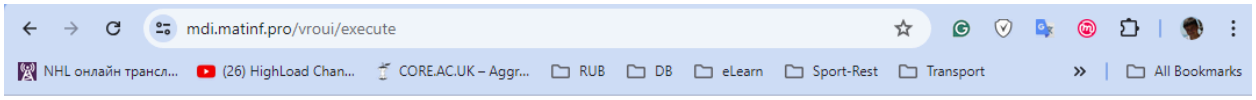
==========

-- Find all Samples containing Au-Pd-Rh-Pt (so 4 elements of higher) with their characterization statistics (statistics columns: EDX, XRD, 4PP, SDC, SECCM, types are common as of June 2024)

```
DECLARE @dt as TABLE([Value] [varchar](2) NOT NULL);

INSERT INTO @dt VALUES('Au'),('Pd'),('Rh'),('Pt');
```

```
select ExternalId as SampleId, ObjectName, ElemNumber, RTRIM(LTRIM([Elements], '-'), '-') as [Elements],

        (       select TOP 1 OI.ObjectName from vro.vroObjectLinkObject as OLO

        INNER JOIN vro.vroObjectInfo as OI ON OLO.LinkedObjectId=OI.ObjectId AND OI.TenantId=S.TenantId

        WHERE OLO.ObjectId=S.ObjectId AND OI.TypeId=5/*Substrate*/) as [SubstrateMaterial]

        , (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE
ParentObjectId=S.ObjectId AND ChildTypeId IN (13, 15, 19, 53, 78, 79)) as EDX

        , (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE
ParentObjectId=S.ObjectId AND ChildTypeId IN (17, 31, 44, 55, 56, 97)) as XRD

        , (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE
ParentObjectId=S.ObjectId AND ChildTypeId IN (14, 16, 33)) as [4PP]

        , (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE
ParentObjectId=S.ObjectId AND ChildTypeId IN (57, 58, 85)) as [SDC]

        , (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE
ParentObjectId=S.ObjectId AND ChildTypeId IN (59, 60, 86, 87)) as [SECCM]

from vro.vrovSample as S

where TypeId=6/*Sample*/ and NOT EXISTS (select top 1 [Value] from @dt WHERE CHARINDEX ('-'+[Value]+'-',
S.[Elements])=0)
===============
```

The results could be the following



MDI Database Search Tree Edit List Edit Handover Reports find SampleID Hello vic.dudarev@gmail.com! Logout

SQL execution tool

show schema

SQL Query

```
-- Find all Samples containing Au-Pd-Rh-Pt (so 4 elements of higher) with their characterization statistics (statistics columns: EDX, XRD 4PP, SDC, SECCM, types are common as of June 2024)
DECLARE @dt as TABLE([Value] [varchar](2) NOT NULL);
INSERT INTO @dt VALUES('Au'),('Pd'),('Rh'),('Pt');
select ExternalId as SampleId, ObjectName, ElemNumber, RTRIM(LTRIM([Elements], '-'), '-') as [Elements],
	(	select TOP 1 OI.ObjectName from vro.vroObjectLinkObject as OLO
		INNER JOIN vro.vroObjectInfo as OI ON OLO.LinkedObjectId=OI.ObjectId AND OI.TenantId=S.TenantId
	WHERE OLO.ObjectId=S.ObjectId AND OI.TypeId=5) as [SubstrateMaterial]
	, (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE ParentObjectId=S.ObjectId AND ChildTypeId IN (13, 15, 19, 78, 79)) as EDX
	, (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE ParentObjectId=S.ObjectId AND ChildTypeId IN (17, 31, 44, 55, 56, 97)) as XRD
	, (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE ParentObjectId=S.ObjectId AND ChildTypeId IN (14, 16, 33)) as [4PP]
	, (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE ParentObjectId=S.ObjectId AND ChildTypeId IN (57, 58, 85)) as [SDC]
	, (select count(ObjectLinkObjectId) FROM [vro].[vrovObjectLinkObject] WHERE ParentObjectId=S.ObjectId AND ChildTypeId IN (69, 60, 86, 87)) as [SECCM]
from vro.vrovSample as S
where TypeId=6/*Sample*/ and NOT EXISTS (select top 1 [Value] from @dt WHERE CHARINDEX ('-'+[Value]+'-', S.[Elements])=0)
```

Execute Save CSV data

Query Execution result

SampleId	ObjectName	ElemNumber	Elements	SubstrateMaterial	EDX	XRD	4PP	SDC	SECCM
10269	10269 Ag-Au-Pd-Pt-Rh on 15nm Ta Library 1	5	Ag-Au-Pd-Pt-Rh	Sapphire	3	0	1	0	0
10275	10275 Ag-Au-Pd-Pt-Rh on 15nm Ta Library 2	5	Ag-Au-Pd-Pt-Rh	Sapphire	3	0	1	0	0
10304	10304 Au-Pd-Pt-Rh on 15nm Ta Library 1	4	Au-Pd-Pt-Rh	Sapphire	3	0	1	0	0
10311	10311 Au-Pd-Pt-Rh-Ru on 15nm Ta Library 1	5	Au-Pd-Pt-Rh-Ru	Sapphire	3	0	1	0	0

Showing 1 to 4 of 4 entries

3) Projects having no representatives (no members)

==========

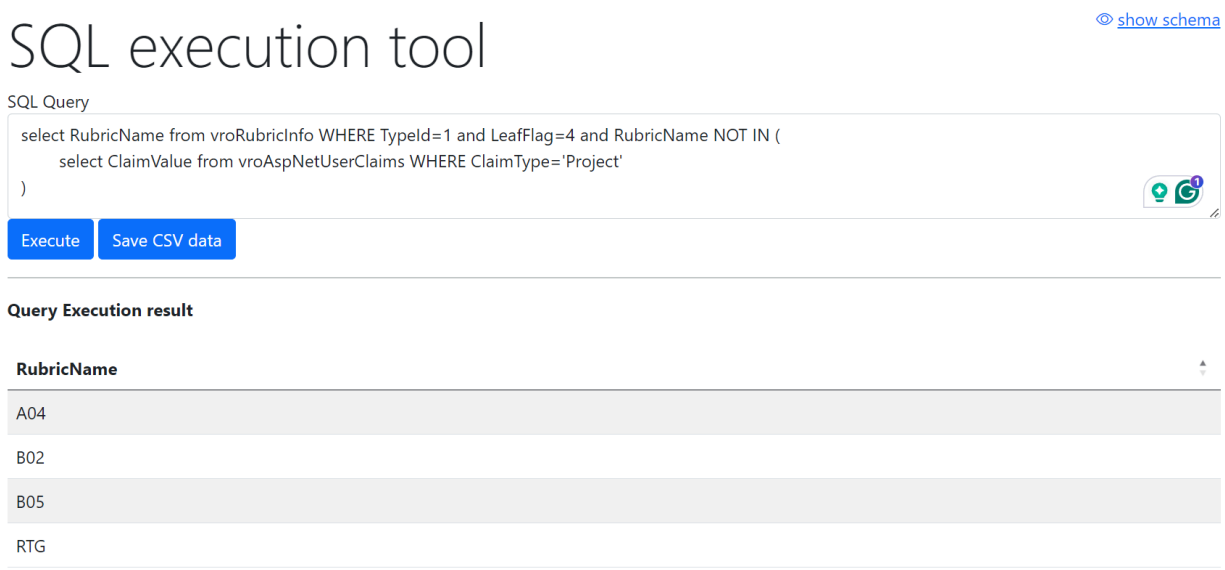
select RubricName from vroRubricInfo WHERE TypeId=1 and LeafFlag=4 and RubricName NOT IN (

select ClaimValue from vroAspNetUserClaims WHERE ClaimType='Project'

)

==========

The results could be the following



4) Wafer types and quantities according to Wager IDs information

* available only in MDI tenant (because of `vro._Wafes` table)

Retrieve types of wafers and their quantity in the `vro._Wafes` table:

==========

select Wafer, count([Index]) as Count

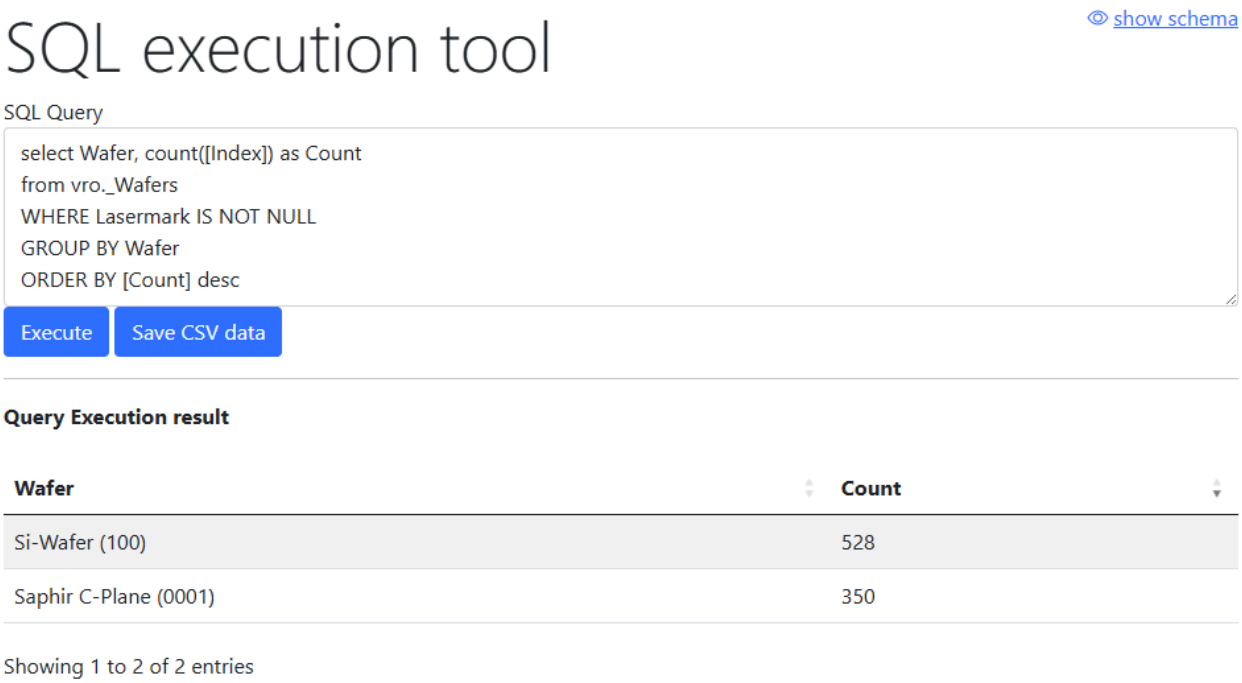
from vro._Wafers

WHERE Lasermark IS NOT NULL

GROUP BY Wafer

ORDER BY [Count] desc

==========



5) Samples/Users/Substrates/WaferIDs

* available only in MDI tenant (because of `vro._Wafers` table)

Retrieve physical samples, their creators, and WafeID data from the _Wafers table starting from the date specified (01.09.2024):

==========

select v.ParentObjectId as ObjectId, v.ParentExternalId as [Sample ID],

'https://mdi.matinf.pro/object/'+v.ParentObjectNameUrl as [URL],

v.Parent_created as _created,

v.ParentObjectName as SampleName, C.ClaimValue as Creator,

SUBSTRING (S.Elements, 2, LEN(S.Elements)-2) as [System], S.ElemNumber as N,

v.ParentObjectDescription as SampleDescription, v.ChildObjectName as [Substrate],

NULLIF(P.[Value],0) as [Wafer ID], W.Wafer

from vro.vrovObjectLinkObject as v

INNER JOIN vro.vroAspNetUsers as U ON v.Parent_createdBy=U.Id

LEFT OUTER JOIN vro.vrovObjectLinkObject as v2 ON v.ParentObjectId=v2.ChildObjectId and v2.ParentTypeId=6 /*parent sample*/

LEFT OUTER JOIN vro.vroAspNetUserClaims as C ON U.Id=C.[UserId] AND C.ClaimType='http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name'

LEFT OUTER JOIN vro.vroSample as s ON v.ParentObjectId=s.SampleId

LEFT OUTER JOIN vro.vroPropertyInt as P ON P.ObjectId=v.ParentObjectId and P.PropertyName='Wafer ID'

LEFT OUTER JOIN vro._Wafers as W ON W.Lasermark=P.Value

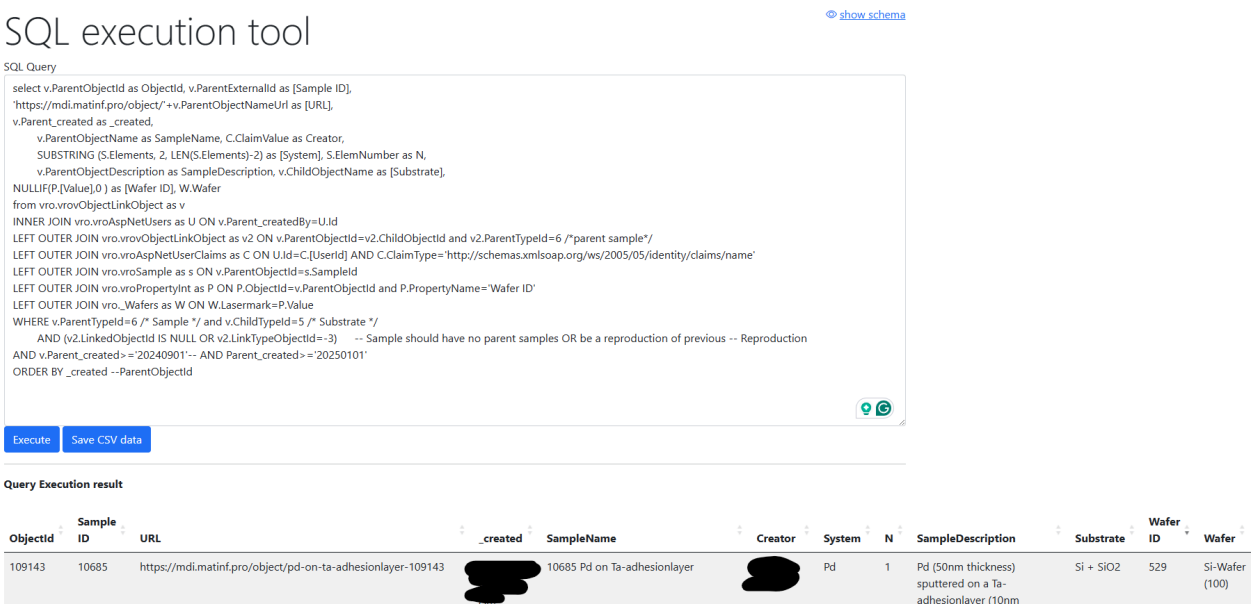
WHERE v.ParentTypeId=6 /* Sample */ and v.ChildTypeId=5 /* Substrate */

AND (v2.LinkedObjectId IS NULL OR v2.LinkTypeObjectId=-3) -- Sample should have no parent samples OR be a reproduction of previous -- Reproduction

AND v.Parent_created>='20240901'

ORDER BY _created

==========



Short explanation/description of the query:

1) it considers links between objects (because a substrate object is connected to the sample via a link) via `vro.vrovObjectLinkObject as v` view, which contains joined parent/child data. The outcome is filtered by parent and child TypeIds accordingly (`v.ParentTypeId=6 /* Sample */ and v.ChildTypeId=5 /* Substrate */`).

2) a join with `vro.vroAspNetUsers as U` view is user to retrieve the user, tht created the sample (parent object, see the join condition: `v.Parent_createdBy=U.Id`)

3) an outer self-join with `vro.vrovObjectLinkObject as v2` with a join condition `v.ParentObjectId=v2.ChildObjectId and v2.ParentTypeId=6 /*parent sample*/` is used to attempt to search parent samples for the current sample (in case of sample splitting or processing). We need to consider only original samples (not the processing steps or the results of splitting samples); this is done via condition in WHERE clause: `AND (v2.LinkedObjectId IS NULL OR v2.LinkTypeObjectId=-3)`, which means that th parent sample either should not exist at all OR it could exist only if the type of link is “Reproduction”, which is marked with `LinkTypeObjectId=-3` in the MDI database.

4) an outer join with `vro.vroAspNetUserClaims as C` with join condition `U.Id=C.[UserId] AND C.ClaimType='http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name'` enables to retrieve the “Name” claimfor the user, who created te sample

5) An outer join with the Sample view (`vro.vroSample as s`) is required to retrieve the chemical system and arity of the sample: join condition: `v.ParentObjectId=s.SampleId`.

6) an outer join with `vro.vroPropertyInt` view is used to retrieve the **Wafer ID** property, as specified in th joi clause: `P.ObjectId=v.ParentObjectId and P.PropertyName='Wafer ID'`

7) an outer join to `vro._Wafers as W` table (which is a copy of Google Spreadsheet, containing wafers lasermarks list, to use it in the query) is used to get the wafer type according to **Wafer ID**

Hints: .P.[Value]>0 could be added to the WHERE clause to output samples with the specified **Wafer ID**.

6) Lonely (disconnected) objects

Retrieve all objects that are disconnected from the object graph (have neither a parent nor a single child):

==========

select * from vroObjectInfo

where ObjectId not IN (select ObjectId from vroObjectLinkObject) AND ObjectId not IN (select LinkedObjectId from vroObjectLinkObject)

==========

SQL execution tool

show schema

SQL Query

```
select * from vroObjectInfo
where ObjectId not IN (select ObjectId from vroObjectLinkObject) AND ObjectId not IN (select LinkedObjectId from vroObjectLinkObject)
```

Execute Save CSV data

Query Execution result

ObjectId	TenantId	_created	_createdBy	_updated	_updatedBy	TypeId	RubricId	SortCode	AccessControl	IsPublished
6648	4	2/8/2023 6:42:32 PM	9	2/8/2023 6:47:59 PM	9	12	139	0	1	True
6649	4	2/8/2023 6:54:36 PM	9	2/8/2023 6:54:41 PM	9	4	136	0	1	True