Gen3 Data Exploration

Thursday, August 8, 2019
1:00 PM - 2:00 PM (CST)

Data Exploration in Gen3

Showcase of Gen3 by Exploring the Cancer Cell Line Encyclopedia Data in the DCF Sandbox Commons

Chris Meyer, Ph.D.

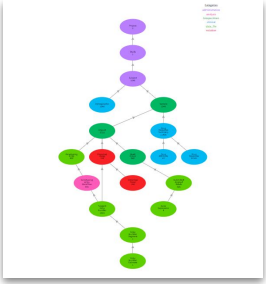
Center for Translational Data Science,
University of Chicago

August 8, 2019

Outline of Webinar Topics

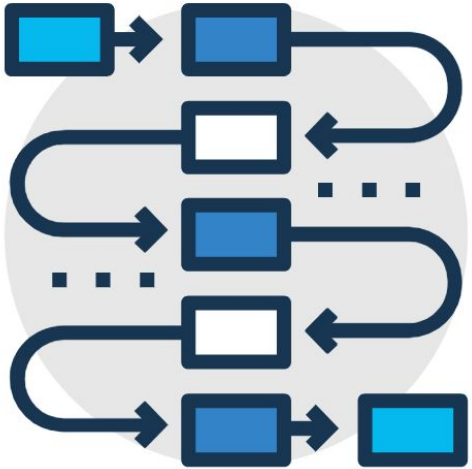
The topics covered during this webinar will include:

1. Exploration of the CCLE Project's Structured Data via:
   - The Graph Model Viewer
   - The Data Explorer UI
   - The GraphQL Open API Endpoints
2. Importing CCLE Data into a Gen3 Workspace
3. Performing Exploratory Data Analysis in JupyterHub
4. Announcing Future Gen3 Developments
5. Live Q&A Session with Gen3's Product Manager



Explore the CCLE Graph Model

Overview of the Project's Structured Data

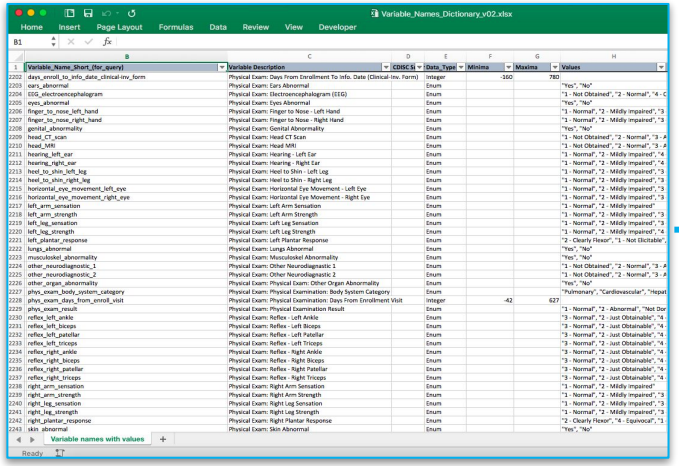


General Concept of the Project Graph Model

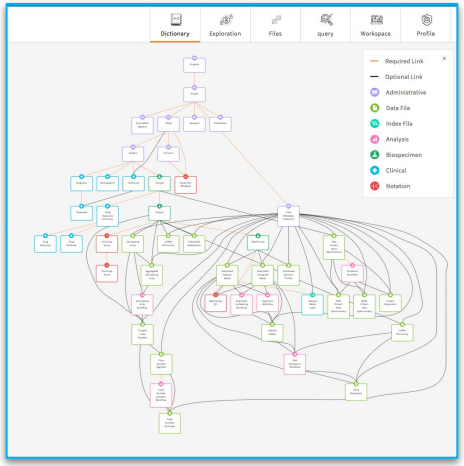
GEN3 Data Commons

- The graphical model of a project is based on the ***data model***, which organizes terms in a data dictionary and defines how they relate to one another.
- The ***data dictionary*** defines how datasets are represented in the database and harmonizes term definitions from different data sources.
- ***Data harmonization*** is foundational to the *data commons* concept of sharing data for cross-project analyses.

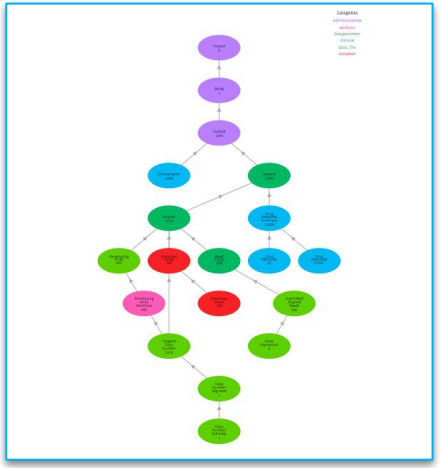
Data Dictionary



Data Model



Specific Project Graph



Structured Data Records in the Graphical Model

GEN3 Data Commons

- A project's graphical model displays interconnected nodes that contain structured data records, which are collections of properties and their values.
- A structured data record in a node stores data as key-value pairs, which can be queried.

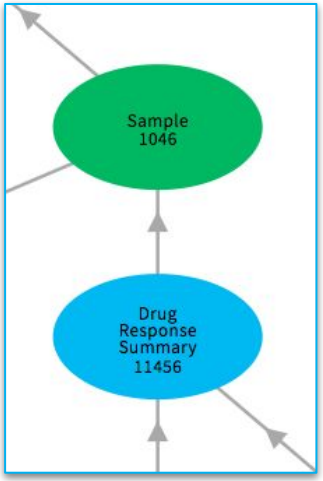
HCC1187_BREAST_AEW541_sumdrug


```
{
  "fit_type": "Constant",
  "IC50": 8,
  "maximum_activity": -6.347360611,
  "submitter_id": "HCC1187_BREAST_AEW541_sumdrug",
  "samples": [
    {
      "id": "1327e573-01bc-4daf-a79c-f8fb1cdcab84",
      "submitter_id": "HCC1187_BREAST"
    }
  ],
  "compound": "AEW541",
  "actarea": 0.2086,
  "project_id": "DCF-CCLE",
  "type": "summary_drug_response",
  "id": "8b4e9d2d-2e7b-43a9-8987-a76d370b52ea"
}
```


Close

Example of one record in the *summary_drug_response* node in JSON format.

- In the CCLE Project, for example, there are 1,046 records in the sample node, which represent samples of individual cell lines.
- Those samples are linked to 11,456 *Drug Response Summary* records containing information about the cell line's response to certain drugs.



Two nodes in the graphical model representation of the CCLE project.

Data Files in the Graph Model

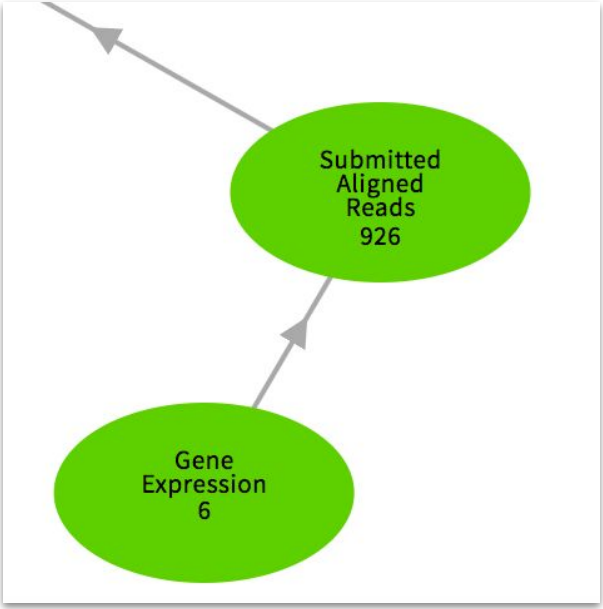
GEN3 Data Commons

- Certain types of data are stored as files as opposed to structured data.
- Records in a data file node describe files with properties like the file's name, size, and GUID, which is a unique identifier pointing to the storage location of the file.


```
{
  "read_groups": [
    {
      "id": "9e054b8f-d088-4973-84b5-284fabaef300",
      "submitter_id": "EB1_HAEMATOPOIETIC_AND_LYMPHOID_TISSUE_rna_read_group"
    }
  ],
  "data_type": "Aligned Reads",
  "file_name": "G41717.EB1.5.bam",
  "md5sum": "64cde2467331dfe3d4c953839986506d",
  "core_metadata_collections": [],
  "data_format": "BAM",
  "object_id": "f3249ae2-af71-45f4-9bd6-cf84c005f7c6",
  "submitter_id": "G41717.EB1.5",
  "data_category": "Sequencing Data",
  "file_size": 16422627814,
  "project_id": "DCF-CCLE",
  "type": "submitted_aligned_reads",
  "id": "ada3b8e6-b0d0-4a3a-8bf1-9c895d2c8b89",
  "experimental_strategy": "RNA-Seq"
}
```


Example data file record in the *aligned_reads* node.

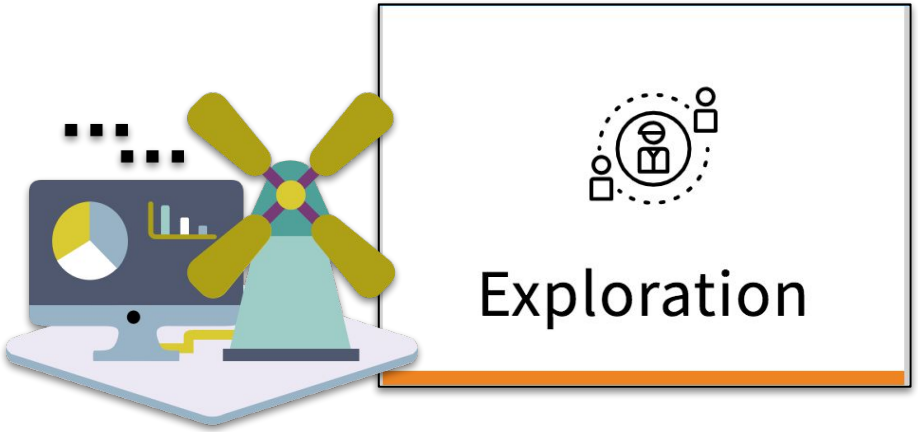
- Here we see that there are 926 files in the *submitted_aligned_reads* node and 6 *gene_expression* files that are linked to those aligned reads files.



Snapshot of two data file nodes in the CCLE project's graphical model.

Data Exploration Tool

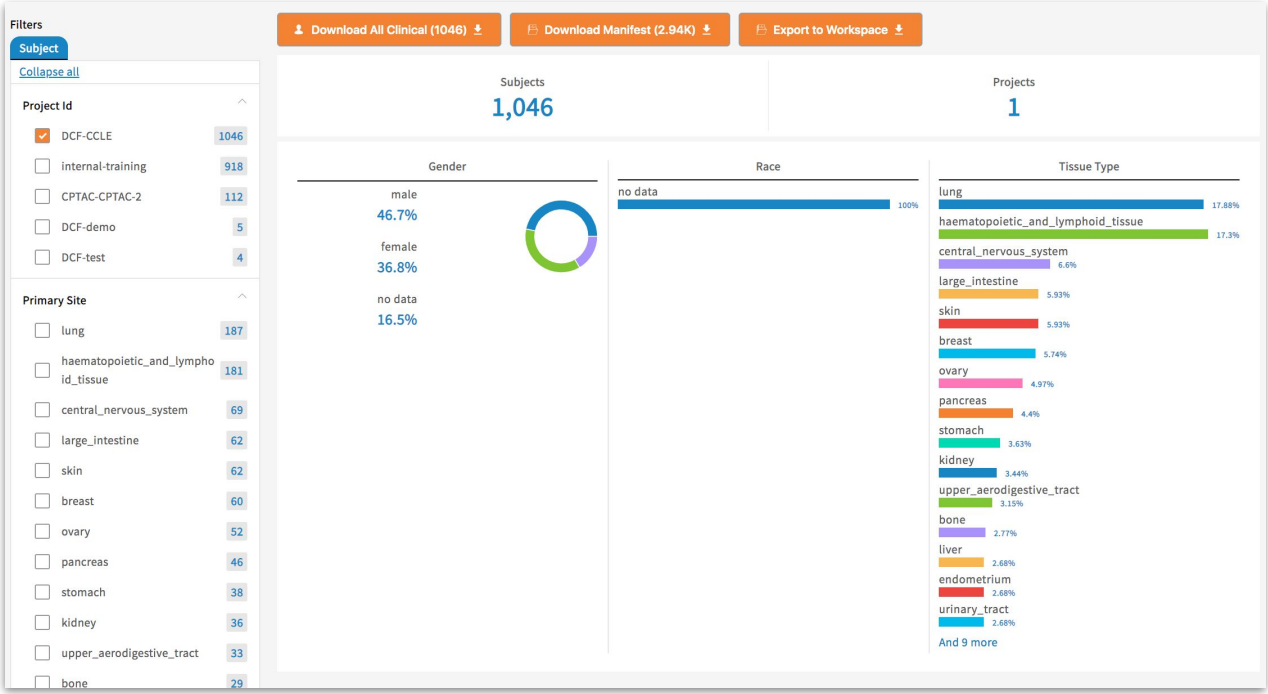
a graphical user interface for cohort selection



Explorer GUI Makes Cohort Selection Easy

GEN3 Data Commons

- Structured data in a project can also be explored using the Exploration GUI.
- Data records are filtered based on the selection of subject properties.
- The records for a selected cohort can then be downloaded or exported to the Workspace.
- A file download manifest can also be generated for downloading all files related to the selected cohort.
- The facets available for filtering and summarizing data are configurable.

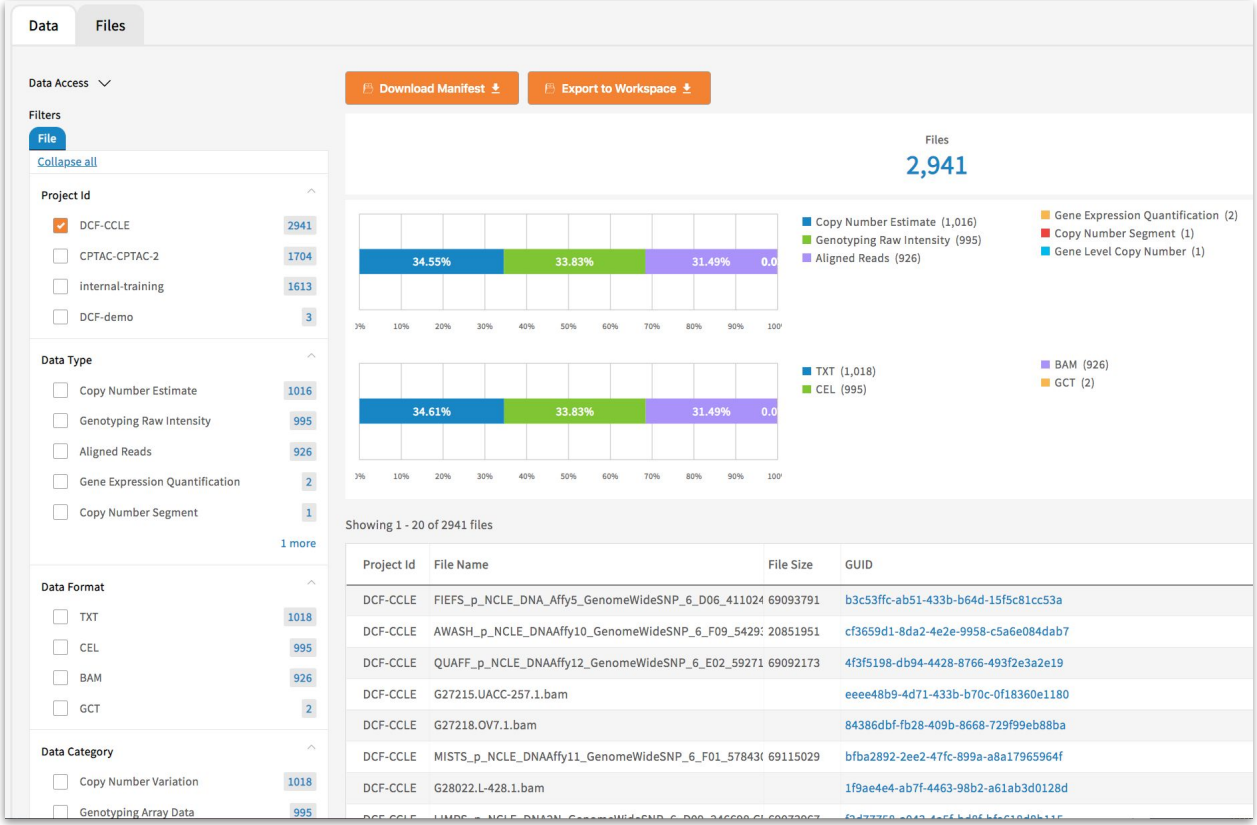


In this example snapshot, the DCF-CCLE project has been selected, and the bar charts are updated automatically, summarizing the number of cell lines of each tissue type in the project.

File Explorer for Downloading Data Files

GEN3 Data Commons

- Data Files in a project can also be explored using the *File* tab of the Exploration GUI.
- Data File records are filtered based on properties like file type, format and category.
- A file download manifest can be generated here as well for easily downloading the batch of files using the gen3-client.
- Files can also be easily exported to the analysis Workspace with a click of the "Export to Workspace" button.



In the above example, the DCF-CCLE project is selected, filtering file records to show only the 2,941 files available in the project, and the bar charts summarize the number of data files of each type, format, and category.

Queries and Programmatic Data Exploration

accessing structured data with queries of Gen3 open APIs

GraphiQL Query Builder Helps Explore CCLE

GEN3 Data Commons

The interactive GraphiQL interface makes building database queries easier.

To access it, click on “Query” in the top navigation bar.

- It features built-in documentation and history.
- Autocomplete for objects, fields, and arguments.
- The ability to pass variables.
- The ability to switch between ElasticSearch and Postgres databases.

This example query searches the *summary_drug_response* node for the compound name, it’s IC50, and the area under the activity curve.

The query also requests the *primary_site* in the subject node, which is the property that stores the tissue type of the cell line.

Query graph

Switch to Flat Model

Switch to Graph Model

GraphiQL ▶ Prettify Merge History

```
1  {
2    summary_drug_response {
3      IC50
4      actarea
5      compound
6      samples {
7        subjects {
8          primary_site
9        }
10     }
11   }
12 }
13
```

QUERY VARIABLES

```
{
  "data": {
    "summary_drug_response": [
      {
        "IC50": 8,
        "actarea": 0.2178,
        "compound": "Sorafenib",
        "samples": [
          {
            "subjects": [
              {
                "primary_site": "pancreas"
              }
            ]
          }
        ]
      },
      {
        "IC50": 8,
        "actarea": 0.1927,
        "compound": "PLX4720",
        "samples": [
          {
            "subjects": [
              {
```

Root summary_drug_response

TYPE

[summary_drug_response]

ARGUMENTS

fit_type: [String]

IC50: [Float]

order_by_asc: String

updated_before: String

without_path_to: [WithPathToInput]

compound: [String]

submitter_id: [String]

with_links: [String]

not: [NotPropertiesInput_summary_drug_response]

offset: Int

actarea: [Float]

created_before: String

Sending Queries to Gen3 Open APIs

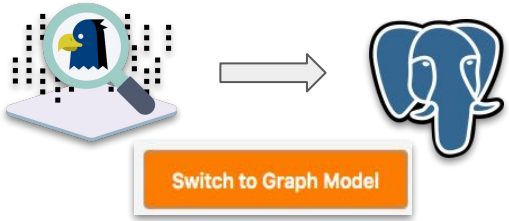
GEN3 Data Commons

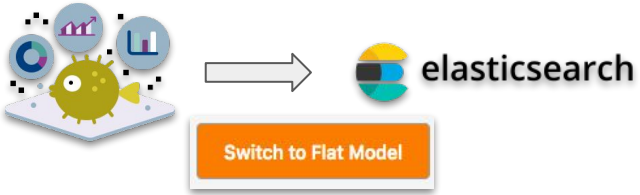
- In addition to the GraphiQL interface, all structured data can be accessed by sending queries (or POST requests) to the following open API endpoints:
  - The PostgreSQL DB, which the graph model is based on and uses the *Peregrine* service:

    <URL>/api/v0/submission/graphql/

  - The ElasticSearch DB, which the Data Explorer GUI is based on and uses our *Guppy* service:

    <URL>/guppy/graphql





- Note that the ElasticSearch database is configured for performant Exploration queries; thus, it typically has only a subset of the structured data available in PostgreSQL, which are the data for cohort selection.

Programmatic Data Access: Peregrine Queries

GEN3 Data Commons

- Queries can be sent to both graphQL API endpoints programmatically.
- This example demonstrates sending a POST request to the PostgreSQL endpoint in a Python Shell.
- The example query requests the compound name, IC50, and area under the activity curve for each compound tested on a particular cell line.

```
In [17]: # Query:
    ...: project_id = 'DCF-CCLE'
    ...: node = 'subject'
    ...:
    ...: props = ['submitter_id','primary_site','samples{summary_drug_responses{IC50,actarea,compound}}']
    ...: properties = ' '.join(map(str,props))
    ...:
    ...: query_txt = """query Test { %s (project_id: "%s") {%s}} """ % (node, project_id, properties)
    ...: query = {'query': query_txt}
    ...:
    ...: graphql_endpoint = api + 'api/v0/submission/graphql/'
    ...: resp = requests.post(graphql_endpoint, json=query, auth=auth).text # Get id from submitter_id
    ...: data = json.loads(resp)
    ...:
    ...: data
    ...:
    ...:
Out[17]:
{'data': {'subject': [{'primary_site': 'breast',
    'samples': [{'summary_drug_responses': [{'IC50': 8.0,
        'actarea': 0.02117,
        'compound': 'PLX4720'},
       {'IC50': 0.17081114, 'actarea': 0.7984, 'compound': 'Sorafenib'},
       {'IC50': 8.0, 'actarea': 0.04209, 'compound': 'PF2341066'},
       {'IC50': 8.0, 'actarea': 0.3953, 'compound': 'Nilotinib'},
       {'IC50': 0.028183434, 'actarea': 2.1929, 'compound': 'Paclitaxel'},
       {'IC50': 1.559987545, 'actarea': 1.6931, 'compound': 'Irinotecan'},
       {'IC50': 0.029439772, 'actarea': 4.2563, 'compound': 'Panobinostat'},
       {'IC50': 8.0, 'actarea': 0.3207, 'compound': 'TAE684'},
       {'IC50': 8.0, 'actarea': 0.03998, 'compound': 'Erlotinib'},
       {'IC50': 8.0, 'actarea': 0.1118, 'compound': 'AEW541'}]}],
    'submitter_id': 'ZR7530_BREAST_subject'},
```

Programmatic Data Access: Guppy Queries

GEN3 Data Commons

- *Guppy* queries can also be sent to the Flat Model endpoint to retrieve structured data in ElasticSearch.
- Guppy queries have the ability to return aggregations and statistics.
- This example demonstrates sending a POST request to the Flat Model (or ElasticSearch) endpoint in Python.
- The query requests the primary_site data as an aggregation, which returns the counts of cell lines per primary site.
- If we had queried for a numerical property, like drug EC50, we could have requested summary statistics with a Guppy query.

```
In [80]: import requests
    ...: api = 'https://nci-crdc-demo.datacommons.io/'
    ...: flat_endpoint = api + 'guppy/graphql'
    ...: af = open('dcf-credentials.json', 'r')
    ...: keys = json.load(af)
    ...: token_url = api + 'user/credentials/cdis/access_token'
    ...: token = requests.post(token_url, json=keys).json()
    ...: headers = {'Authorization': 'bearer '+ token['access_token']}
    ...: query_txt = """
    ...: {
    ...:   _aggregation {
    ...:     subject {
    ...:       primary_site {
    ...:         histogram {
    ...:           key
    ...:           count
    ...:         }
    ...:       }
    ...:     }
    ...:   }
    ...: }
    ...: """
    ...: query = {'query': query_txt}
    ...: resp = requests.post(flat_endpoint, json=query, headers=headers).text
    ...: data = json.loads(resp)
    ...: data
    ...:
Out[80]:
{'data': {'_aggregation': {'subject': {'primary_site': {'histogram': [{'count': 187,
    'key': 'lung'},
    {'count': 181, 'key': 'haematopoietic_and_lymphoid_tissue'},
    {'count': 112, 'key': 'Breast'},
    {'count': 69, 'key': 'central_nervous_system'},
    {'count': 62, 'key': 'skin'},
    {'count': 62, 'key': 'large_intestine'},
```

Programmatic Data Access: the Gen3 SDK

GEN3 Data Commons

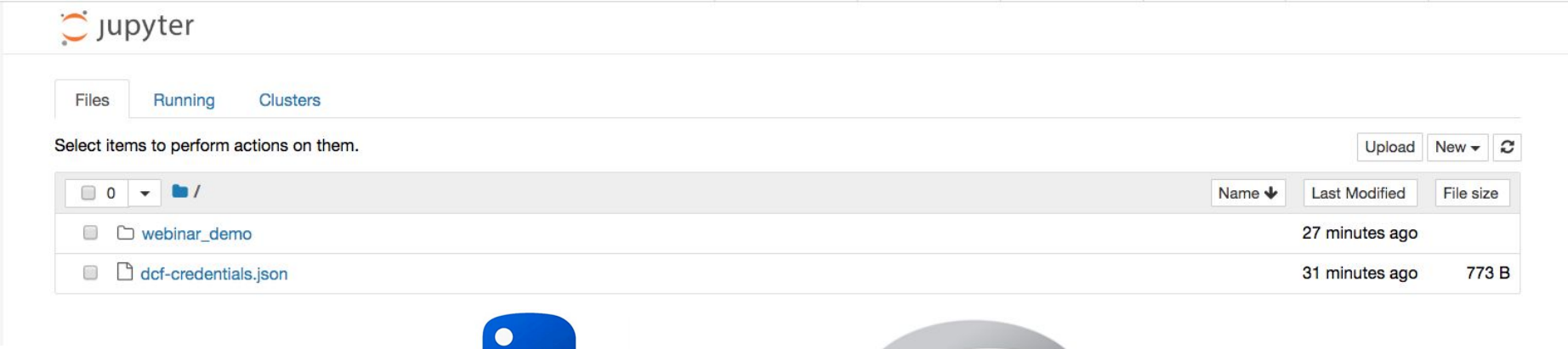
- Instead of requesting specific data with queries, all structured data in a node can be exported using the Gen3 SDK, which is an open-source suite of functions for interacting with Gen3 APIs via the command-line.
- The Gen3 SDK is installed and imported into Python using the commands *install gen3* and *import gen3*.
- The code is also publically available on GitHub: https://github.com/uc-cdis/gen3sdk-python
- In this example, all records in the CCLE project's *summary_drug_response* node are exported as a single TSV.

```
In [39]: ## Export structured data using Gen3 SDK
    ...: import gen3
    ...: from gen3.auth import Gen3Auth
    ...: from gen3.submission import Gen3Submission
    ...:
    ...: api = 'https://nci-crdc-demo.datacommons.io/' # DCF Sandbox Commons
    ...: profile = 'dcf'
    ...: creds = '/Users/christopher/Downloads/dcf-credentials.json'
    ...: auth = Gen3Auth(api, refresh_file=creds)
    ...: sub = Gen3Submission(api, auth)
    ...:
    ...: program = 'DCF'
    ...: project = 'CCLE'
    ...: node_type = 'summary_drug_response'
    ...: fileformat = 'tsv'
    ...: data = sub.export_node(program, project, node_type, fileformat)
    ...: print(os.linesep.join(data.split(os.linesep)[:10]))
    ...:
type    id      project_id      submitter_id    EC50    IC50    actarea compound        fit_type        maximum_activity
summary_drug_response   df24950a-2a79-43ec-9daf-9aac8eb5a91f    DCF-CCLE        SU8686_PANCREAS_Sorafenib_sumdrug
summary_drug_response   f60c5f23-445f-448c-a28e-633f7b429e78    DCF-CCLE        OC316_OVARY_PLX4720_sumdrug             8.0
summary_drug_response   d66b3388-206c-4172-bd09-b5a535b9fd17    DCF-CCLE        PC3_PROSTATE_AZD6244_sumdrug            8.0
summary_drug_response   30c17487-8223-4b41-b054-5063d0036c27    DCF-CCLE        UACC257_SKIN_AZD6244_sumdrug    0.146748021
```

Workspace JupyterHub Demonstration

GEN3 Data Commons

- Now, we will take a look at the CCLE Project in the Gen3 Workspace.
- Data will be accessed, prepared and summarized in a Python notebook, and then the data will be further visualized in an R notebook.



Future Gen3 Developments

a look at some up-and-coming features of the Gen3 platform

Future Gen3 Developments

GEN3 Data Commons

- Users will soon be able to run containerized pipelines using our Workflow Execution Services.
- We will be improving the ability to export clinical data to workspaces.
- Clinical data will be versioned, and users will have the ability to select different versions of harmonized clinical data.
- We’re redesigning the data submission process to make it simpler and more intuitive.

Learn More about the Gen3 Platform

GEN3 Data Commons

You can learn more about the Gen3 software stack for creating data commons through the following resources:

- Our open source code lives on GitHub: github.com/uc-cdis
- Documentation for Users, Developers and Operators is at gen3.org
- Chat with developers and support live in our Gen3 Community on Slack
- Get involved in Q&A at the Gen3 Forum: https://forums.gen3.org/
- Email our support team at: support@datacommons.io
- Learn more about the Center for Translational Data Science (CTDS) at ctds.uchicago.edu

Selected Data Commons Using Gen3

GEN3 Data Commons

Finally, you can view data statistics on several Gen3-powered Data Commons at stats.gen3.org

NIH NATIONAL CANCER INSTITUTE
Cancer Research Data Commons

Gabriella Miller Kids First
PEDIATRIC RESEARCH PROGRAM

DataSTAGE
Powered by Gen3

NIAID DATA HUB

NIH National Heart, Lung, and Blood Institute

BloodPAC
BLOOD PROFILING ATLAS IN CANCER

AnVIL

BRAINCOMMONS

NIH National Human Genome Research Institute

Questions?

GEN3 Data Commons