PyNIDM Documentation

Release 4.1.0.post45+geb0fa56

dbkeator@uci.edu, camille.maumet@inria.fr, satrajit.ghosh@gmai

Feb 09, 2024

CONTENTS:

A Python library to manipulate the Neuroimaging Data Model.

CHAPTER

ONE

DEPENDENCIES

- Git-annex
- Graphviz (native package):
  - Fedora: *dnf install graphviz*
  - OS-X: *brew install graphviz*

CHAPTER

TWO

INSTALLATION

```
$ pip install pynidm
```

CHAPTER

THREE

CONTRIBUTING TO THE SOFTWARE

This software is open source and community developed. As such, we encourage anyone and everyone interested in semantic web and neuroimaging to contribute. To begin contributing code to the repository, please fork the main repo into your user space and use the pull request GitHub feature to submit code for review. Please provide a reasonably detailed description of what was changed and why in the pull request.

CHAPTER

FOUR

REPORTING ISSUES OR PROBLEMS

If you encounter a bug, you can directly report it in the issues section. Please describe how to reproduce the issue and include as much information as possible that can be helpful for fixing it. If you would like to suggest a fix, please open a new pull request or include your suggested fix in the issue.

CHAPTER

FIVE

SUPPORT AND FEEDBACK

We would love to hear your thoughts on our Python toolbox. Feedback, questions, or feature requests can also be submitted as issues. Note, we are a small band of researchers who mostly volunteer our time to this project. We will respond as quickly as possible.

CHAPTER

SIX

NIDM-EXPERIMENT TOOLS

6.1 BIDS MRI Conversion to NIDM

This program will convert a BIDS MRI dataset to a NIDM-Experiment RDF document. It will parse phenotype information and simply store variables/values and link to the associated json data dictionary file. To use this tool please set your INTERLEX_API_KEY environment variable to your unique API key. To get an Interlex API key you visit SciCrunch, register for an account, then click on "MyAccount" and "API Keys" to add a new API key for your account.

```
$ bidsmri2nidm -d [ROOT BIDS DIRECT] -bidsignore

usage: bidsmri2nidm [-h] -d DIRECTORY [-jsonld] [-bidsignore] [-no_concepts]
                    [-json_map JSON_MAP] [-log LOGFILE] [-o OUTPUTFILE]

This program will represent a BIDS MRI dataset as a NIDM RDF document and provide user␣
↪with opportunity to annotate
the dataset (i.e. create sidecar files) and associate selected variables with broader␣
↪concepts to make datasets more
FAIR.

Note, you must obtain an API key to Interlex by signing up for an account at scicrunch.
↪org then going to My Account
and API Keys.  Then set the environment variable INTERLEX_API_KEY with your key.

optional arguments:
  -h, --help            show this help message and exit
  -d DIRECTORY          Full path to BIDS dataset directory
  -jsonld, --jsonld     If flag set, output is json-ld not TURTLE
  -bidsignore, --bidsignore
                        If flag set, tool will add NIDM-related files to .bidsignore file
  -no_concepts, --no_concepts
                        If flag set, tool will no do concept mapping
  -log LOGFILE, --log LOGFILE
                        Full path to directory to save log file. Log file name is␣
↪bidsmri2nidm_[basename(args.directory)].log
  -o OUTPUTFILE         Outputs turtle file called nidm.ttl in BIDS directory by default.
↪.or whatever path/filename is set here

map variables to terms arguments:
  -json_map JSON_MAP, --json_map JSON_MAP
                        Optional full path to user-suppled JSON file containing data␣
↪element definitions.
```

6.2 CSV File to NIDM Conversion

This program will load in a CSV file and iterate over the header variable names performing an elastic search of https://scicrunch.org/nidm-terms for NIDM-ReproNim tagged terms that fuzzy match the variable names. The user will then interactively pick a term to associate with the variable name. The resulting annotated CSV data will then be written to a NIDM data file. To use this tool please set your INTERLEX_API_KEY environment variable to your unique API key. To get an Interlex API key you visit SciCrunch, register for an account, then click on "MyAccount" and "API Keys" to add a new API key for your account.

```
usage: csv2nidm [-h] -csv CSV_FILE [-json_map JSON_MAP | -redcap REDCAP]
                [-nidm NIDM_FILE] [-no_concepts] [-log LOGFILE] -out
                OUTPUT_FILE

This program will load in a CSV file and iterate over the header variable
names performing an elastic search of https://scicrunch.org/ for NIDM-ReproNim
tagged terms that fuzzy match the variable names. The user will then
interactively pick a term to associate with the variable name. The resulting
annotated CSV data will then be written to a NIDM data file. Note, you must
obtain an API key to Interlex by signing up for an account at scicrunch.org
then going to My Account and API Keys. Then set the environment variable
INTERLEX_API_KEY with your key.  The tool supports import of RedCap data
dictionaries and will convert relevant information into a json-formatted
annotation file used to annotate the data elements in the resulting NIDM file.

optional arguments:
  -h, --help            show this help message and exit
  -csv CSV_FILE         Full path to CSV file to convert
  -json_map JSON_MAP    Full path to user-suppled JSON file containing
                        variable-term mappings.
  -redcap REDCAP        Full path to a user-supplied RedCap formatted data
                        dictionary for csv file.
  -nidm NIDM_FILE       Optional full path of NIDM file to add CSV->NIDM
                        converted graph to
  -no_concepts          If this flag is set then no concept associations will
                        beasked of the user. This is useful if you already
                        have a -json_map specified without concepts and want
                        tosimply run this program to get a NIDM file with user
                        interaction to associate concepts.
  -log LOGFILE, --log LOGFILE
                        full path to directory to save log file. Log file name
                        is csv2nidm_[arg.csv_file].log
  -out OUTPUT_FILE      Full path with filename to save NIDM file
```

6.3 convert

This function will convert NIDM files to various RDF-supported formats and name then / put them in the same place as the input file.

```
Usage: pynidm convert [OPTIONS]

Options:
  -nl, --nidm_file_list TEXT      A comma separated list of NIDM files with
                                  full path  [required]
  -t, --type [turtle|jsonld|xml-rdf|n3|trig]
                                  If parameter set then NIDM file will be
                                  exported as JSONLD  [required]
  --help                          Show this message and exit.
```

6.4 concatenate

This function will concatenate NIDM files. Warning, no merging will be done so you may end up with multiple prov:agents with the same subject id if you're concatenating NIDM files from multiple visits of the same study. If you want to merge NIDM files on subject ID see pynidm merge

```
Usage: pynidm concat [OPTIONS]

Options:
  -nl, --nidm_file_list TEXT  A comma separated list of NIDM files with full
                              path  [required]
  -o, --out_file TEXT         File to write concatenated NIDM files
                              [required]
  --help                      Show this message and exit.
```

6.5 visualize

This command will produce a visualization(pdf) of the supplied NIDM files named the same as the input files and stored in the same directories.

```
Usage: pynidm visualize [OPTIONS]

Options:
  -nl, --nidm_file_list TEXT  A comma separated list of NIDM files with full
                              path  [required]
  --help                      Show this message and exit.
```

6.6 merge

This function will merge NIDM files. See command line parameters for supported merge operations.

```
Usage: pynidm merge [OPTIONS]

Options:
  -nl, --nidm_file_list TEXT  A comma separated list of NIDM files with full
                              path  [required]
  -s, --s                     If parameter set then files will be merged by
                              ndar:src_subjec_id of prov:agents
      -o, --out_file TEXT         File to write concatenated NIDM files
                              [required]
      --help                      Show this message and exit.
```

6.7 Query

This function provides query support for NIDM graphs.

```
Usage: pynidm query [OPTIONS]

Options:
  -nl, --nidm_file_list TEXT      A comma separated list of NIDM files with
                                  full path  [required]
  -nc, --cde_file_list TEXT       A comma separated list of NIDM CDE files
                                  with full path. Can also be set in the
                                  CDE_DIR environment variable
  -q, --query_file FILENAME       Text file containing a SPARQL query to
                                  execute
  -p, --get_participants          Parameter, if set, query will return
                                  participant IDs and prov:agent entity IDs
  -i, --get_instruments           Parameter, if set, query will return list of
                                  onli:assessment-instrument:
  -iv, --get_instrument_vars      Parameter, if set, query will return list of
                                  onli:assessment-instrument: variables
  -de, --get_dataelements         Parameter, if set, will return all
                                  DataElements in NIDM file
  -debv, --get_dataelements_brainvols
                                  Parameter, if set, will return all brain
                                  volume DataElements in NIDM file along with
                                  details
  -bv, --get_brainvols            Parameter, if set, will return all brain
                                  volume data elements and values along with
                                  participant IDs in NIDM file
  -o, --output_file TEXT          Optional output file (CSV) to store results
                                  of query
  -u, --uri TEXT                  A REST API URI query
  -j / -no_j                      Return result of a uri query as JSON
  -v, --verbosity TEXT            Verbosity level 0-5, 0 is default
  --help                          Show this message and exit.
```

6.8 linear_regression

This function provides linear regression support for NIDM graphs.

```
Usage: pynidm linear-regression [OPTIONS]

Options:
  -nl, --nidm_file_list TEXT      A comma-separated list of NIDM files with
                                  full path  [required]
  -r, --regularization TEXT       Parameter, if set, will return the results of
                                  the linear regression with L1 or L2 regularization
                                  depending on the type specified, and the weight
                                  with the maximum likelihood solution. This will
                                  prevent overfitting. (Ex: -r L1)
  -model, --ml TEXT                   An equation representing the linear
                                  regression. The dependent variable comes
                                  first, followed by "=" or "~", followed by
                                  the independent variables separated by "+"
                                  (Ex: -model "fs_003343 = age*sex + sex +
                                  age + group + age*group + bmi") [required]
  -contstant, --ctr TEXT              Parameter, if set, will return differences in
                                  variable relationships by group. One or
                                  multiple parameters can be used (multiple
                                  parameters should be separated by a comma-
                                  separated list) (Ex: -contrast group,age)
  -o, --output_file TEXT          Optional output file (TXT) to store results
                                  of query
  --help                          Show this message and exit.
```

To use the linear regression algorithm successfully, structure, syntax, and querying is important. Here is how to maximize the usefulness of the tool:

First, use pynidm query to discover the variables to use. PyNIDM allows for the use of either data elements (PIQ_tca9ck), specific URLs (http://uri.interlex.org/ilx_0100400), or source variables (DX_GROUP).

An example of a potential query is:

```
pynidm query -nl /simple2_NIDM_examples/datasets.datalad.org/abide/RawDataBIDS/CMU_a/
↪nidm.ttl,/simple2_NIDM_examples/datasets.datalad.org/abide/RawDataBIDS/CMU_b/nidm.ttl -
↪u /projects?fields=fs_000008,DX_GROUP,PIQ_tca9ck,http://uri.interlex.org/ilx_0100400
```

You can also do:

```
pynidm query -nl /simple2_NIDM_examples/datasets.datalad.org/abide/RawDataBIDS/CMU_a/
↪nidm.ttl,/Users/Ashu/Downloads/simple2_NIDM_examples/datasets.datalad.org/abide/
↪RawDataBIDS/CMU_b/nidm.ttl -gf fs_000008,DX_GROUP,PIQ_tca9ck,http://uri.interlex.org/
↪ilx_0100400
```

The query looks in the two files specified in the -nl parameter for the variables specified. In this case, we use fs_000008 and DX_GROUP (source variables), a URL (http://uri.interlex.org/ilx_0100400), and a data element (PIQ_tca9ck). The output of the file is slightly different depending on whether you use -gf or -u. With -gf, it will return the variables from both files separately, while -u combines them.

Now that we have selected the variables, we can perform a linear regression. In this example, we will look at the effect of DX_GROUP, age at scan, and PIQ on supratentorial brain volume.

The command to use for this particular data is:

```
pynidm linear-regression -nl /simple2_NIDM_examples/datasets.datalad.org/abide/
↪RawDataBIDS/CMU_a/nidm.ttl,/simple2_NIDM_examples/datasets.datalad.org/abide/
↪RawDataBIDS/CMU_b/nidm.ttl -model "fs_000008 = DX_GROUP + PIQ_tca9ck + http://uri.
↪interlex.org/ilx_0100400" -contrast "DX_GROUP" -r L1
```

-nl specifies the file(s) to pull data from, while -model is the model to perform a linear regression model on. In this case, the variables are fs_000008 (the dependent variable, supratentorial brain volume), DX_GROUP (diagnostic group), PIQ_tca9ck (PIQ), and http://uri.interlex.org/ilx_0100400 (age at scan). The -contrast parameter says to contrast the data using DX_GROUP, and then do a L1 regularization to prevent overfitting.

Details on the REST API URI format and usage can be found below.

CHAPTER

SEVEN

PYNIDM: REST API AND COMMAND LINE USAGE

7.1 Introduction

There are two main ways to interact with NIDM data using the PyNIDM REST API. First, the pynidm query command line utility will accept queries formatted as REST API URIs. Second, the rest-server.py script can be used to run a HTTP server to accept and process requests. This script can either be run directly or using a docker container defined in the docker directory of the project.

Example usage:

```
$ pynidm query -nl "cmu_a.ttl,cmu_b.ttl" -u /projects

dc1bf9be-10a3-11ea-8779-003ee1ce9545
ebe112da-10a3-11ea-af83-003ee1ce9545
```

7.2 Installation

To use the REST API query syntax on the command line, follow the PyNIDM installation instructions.

The simplest way to deploy a HTTP REST API server would be with the provided docker container. You can find instructions for that process in the README.md file in the docker directory of the Github repository.

7.3 URI formats

Here is a list of the current operations.

```
- /projects
- /projects/{project_id}
- /projects/{project_id}/subjects
- /projects/{project_id}/subjects
- /projects/{project_id}/subjects/{subject_id}
- /projects/{project_id}/subjects/{subject_id}/instruments
- /projects/{project_id}/subjects/{subject_id}/instruments/{instrument_id}
- /projects/{project_id}/subjects/{subject_id}/derivatives/
- /projects/{project_id}/subjects/{subject_id}/derivatives/{derivative_id}
- /subjects
- /subjects/{subject_id}
- /statistics/projects/{project_id}
```

(continues on next page)

(continued from previous page)

```
- /dataelements
- /dataelements/{dataelement_id}
```

You can append the following query parameters to many of the operations:

```
- filter
- field
```

7.3.1 Operations

/projects
Get a list of all project IDs available.

Supported optional query parameters: fields

/projects/{project_id}
See some details for a project. This will include project summary information (acquisition modality, contrast type, image usage, etc) as well as a list of subject IDs and data elements used in the project.

When a fields parameters are provided, all instrument/derivative data in the project matching the field list will be returned as a table.

When a filter parameter is provided, the list of subjects returned will only include subjects that have data passing the filter

Supported optional query parameters: filter, fields

/projects/{project_id}/subjects
Get the list of subjects in a project

When a filter parameter is provided only subjects matching the filter will be returned.

Supported optional query parameters: filter

/projects/{project_id}/subjects/{subject_id}
Get the details for a particular subject. This will include the results of any instrumnts or derivatives associated with the subject, as well a list of the related activities.

Supported query parameters: none

/projects/{project_id}/subjects/{subject_id}/instruments
Get a list of all instruments associated with that subject.

Supported query parameters: none

/projects/{project_id}/subjects/{subject_id}/instruments/{instrument_id}
Get the values for a particular instrument

Supported query parameters: none

/projects/{project_id}/subjects/{subject_id}/derivatives
Get a list of all instruments associated with that subject.

Supported query parameters: none

/projects/{project_id}/subjects/{subject_id}/derivatives/{derivative_id}
Get the values for a particular derivative

Supported query parameters: none

`/subjects`
: Returns the UUID and Source Subject ID for all subjects available.

  If the fields parameter is provided, the result will also include a table of subjects along with the values for the supplied fields in any instrument or derivative

  Supported query parameters: fields

`/subjects/{subject_id}`
: Get the details for a particular subject. This will include the results of any instrumnts or derivatives associated with the subject, as well a a list of the related activities.

  Supported query parameters: none

`/statistics/projects/{project_id}`
: See project statistics. You can also use this operation to get statsitcs on a particular instrument or derivative entry by use a *field* query option.

  Supported query parameters: filter, field

`/statistics/projects/{project_id}/subjects/{subject_id}`
: See some details for a project. This will include the list of subject IDs and data elements used in the project

  Supported query parameters: none

`/dataelements/{identifier}`
: Returns a table of details on the dataelement that has any synonym matching the provided identifier. The system will attempt to match the data element label, isAbout URI, or data element URI. The return result will also provide a list of projects where the data element is in use.

  Supported query parameters: none

7.3.2 Query Parameters

`filter`
: The filter query parameter is used when you want to receive data only on subjects that match some criteria. The format for the filter value should be of the form:

```
identifier op value [ and identifier op value and ... ]
```

  Identifiers should be formatted as either a simple field, such as "age", or if you want to restrict the match to just instruments or derivatives format it ia "derivatives.ID" or "derivatives.Subcortical gray matter volume (mm^3)"

  You can use any value for identifier that is shown in the data_elements section of the project details. For a derivative ID, you can use the last component of a derivative field URI (ex. for the URI http://purl.org/nidash/fsl#fsl_000007, the ID would be "fsl_000007") or the exact label shown when viewing derivative data (ex. "Left-Caudate (mm^3)").

  The op can be one of "eq", "gt", "lt".

  Example filters:
  `?filter=instruments.AGE_AT_SCAN gt 30  ?filter=instrument.AGE_AT_SCAN eq 21 and derivative.fsl_000007 lt 3500`

`fields`
: The fields query parameter is used to specify what fields should be detailed. The matching rules are similar to the filter parameter.

  Example field query:
  `http://localhost:5000/statistics/projects/abc123?field=AGE_AT_SCAN,derivatives.fsl_000020`

For identifiers in both the fields and filters, when PyNIDM is trying to match your provided value with data in the file a list of synonyms will be created to facilitate the match. This allows you to use the exact identifier, URI, data element label, or an "is about" concept URI if available.

7.4 Return Formatting

By default the HTTP REST API server will return JSON formatted objects or arrays. When using the pynidm query command line utility the default return format is text (when possible) or you can use the -j option to have the output formatted as JSON.

7.4.1 Examples

Get the UUID for all the projects at this location

```
curl http://localhost:5000/projects
```

Example response:

```
[
    "dc1bf9be-10a3-11ea-8779-003ee1ce9545"
]
```

Get the project summary details

```
curl http://localhost:5000/projects/dc1bf9be-10a3-11ea-8779-003ee1ce9545
```

Example response:

```
{
  "AcquisitionModality": [
    "MagneticResonanceImaging"
  ],
  "ImageContrastType": [
    "T1Weighted",
    "FlowWeighted"
  ],
  "ImageUsageType": [
    "Anatomical",
    "Functional"
  ],
  "Task": [
    "rest"
  ],
  "sio:Identifier": "1.0.1",
  "dctypes:title": "ABIDE CMU_a Site",
  "http://www.w3.org/1999/02/22-rdf-syntax-ns#type": "http://www.w3.org/ns/prov#Activity
↪",
  "prov:Location": "file://datasets.datalad.org/abide/RawDataBIDS/CMU_a",
  "subjects": [
```

(continues on next page)

(continued from previous page)

```
    "fdb6c8bc-67aa-11ea-ba45-003ee1ce9545",
    "b276ebb6-67aa-11ea-ba45-003ee1ce9545",
    "a38c4e42-67aa-11ea-ba45-003ee1ce9545",
    "a2ff751c-67aa-11ea-ba45-003ee1ce9545",
    "cfce5728-67aa-11ea-ba45-003ee1ce9545",
    "f165e7ae-67aa-11ea-ba45-003ee1ce9545",
    "cf4605ee-67aa-11ea-ba45-003ee1ce9545",
    "a1efa78c-67aa-11ea-ba45-003ee1ce9545",
    "d0de8ebc-67aa-11ea-ba45-003ee1ce9545",
    "a4a999ba-67aa-11ea-ba45-003ee1ce9545",
    "a0555098-67aa-11ea-ba45-003ee1ce9545",
    "b41d75f2-67aa-11ea-ba45-003ee1ce9545",
    "be3fbff0-67aa-11ea-ba45-003ee1ce9545",
    "eec5a0ca-67aa-11ea-ba45-003ee1ce9545"
  ],
  "data_elements": [
    "SCQ_TOTAL", "VIQ", "VINELAND_WRITTEN_V_SCALED", "WISC_IV_VCI", "ADOS_COMM", "FILE_ID
↪", "WISC_IV_BLK_DSN_SCALED",
    "WISC_IV_SYM_SCALED", "ADI_R_SOCIAL_TOTAL_A", "WISC_IV_INFO_SCALED", "ADOS_GOTHAM_
↪SEVERITY",
    "VINELAND_COMMUNICATION_STANDARD", "VINELAND_PERSONAL_V_SCALED", "SUB_ID", "ADOS_
↪GOTHAM_TOTAL",
    "ADI_R_VERBAL_TOTAL_BV", "VINELAND_COPING_V_SCALED", "VINELAND_DOMESTIC_V_SCALED",
↪"SRS_COGNITION",
    "FIQ_TEST_TYPE", "WISC_IV_PSI", "OFF_STIMULANTS_AT_SCAN", "VINELAND_PLAY_V_SCALED",
↪"AGE_AT_MPRAGE",
    "VIQ_TEST_TYPE", "ADI_RRB_TOTAL_C", "WISC_IV_DIGIT_SPAN_SCALED", "FIQ", "DSM_IV_TR",
↪"DX_GROUP",
    "VINELAND_INTERPERSONAL_V_SCALED", "VINELAND_SUM_SCORES", "ADOS_STEREO_BEHAV", "ADI_
↪R_ONSET_TOTAL_D",
    "ADOS_GOTHAM_SOCAFFECT", "ADOS_GOTHAM_RRB", "CURRENT_MED_STATUS", "VINELAND_
↪EXPRESSIVE_V_SCALED",
    "AGE_AT_SCAN", "WISC_IV_PRI", "SEX", "SRS_RAW_TOTAL", "ADOS_RSRCH_RELIABLE", "WISC_
↪IV_SIM_SCALED",
    "WISC_IV_CODING_SCALED", "SRS_MANNERISMS", "AQ_TOTAL", "HANDEDNESS_SCORES",
↪"HANDEDNESS_CATEGORY",
    "SRS_VERSION", "ADI_R_RSRCH_RELIABLE", "EYE_STATUS_AT_SCAN", "MEDICATION_NAME",
↪"ADOS_SOCIAL",
    "ADOS_MODULE", "VINELAND_RECEPTIVE_V_SCALED", "VINELAND_DAILYLVNG_STANDARD",
↪"VINELAND_ABC_STANDARD",
    "PIQ", "VINELAND_SOCIAL_STANDARD", "SITE_ID", "COMORBIDITY", "BMI", "VINELAND_
↪COMMUNITY_V_SCALED",
    "ADOS_TOTAL", "VINELAND_INFORMANT", "WISC_IV_WMI", "WISC_IV_MATRIX_SCALED", "WISC_IV_
↪LET_NUM_SCALED",
    "PIQ_TEST_TYPE", "SRS_COMMUNICATION", "WISC_IV_VOCAB_SCALED", "SRS_AWARENESS", "WISC_
↪IV_PIC_CON_SCALED",
    "SRS_MOTIVATION"
  ]
}
```

Get Left-Pallidum volume (fsl_0000012) values for all subjects in a project

```
pynidm query -nl ttl/cmu_a.ttl -u /projects/cc305b3e-67aa-11ea-ba45-003ee1ce9545?
↪fields=fsl_000012
```

```
-------------------------------------------  ---------------------------------------
↪-------------
AcquisitionModality                          ["MagneticResonanceImaging"]
ImageContrastType                            ["FlowWeighted", "T1Weighted"]
ImageUsageType                               ["Functional", "Anatomical"]
Task                                         ["rest"]
sio:Identifier                               "1.0.1"
dctypes:title                                "ABIDE CMU_a Site"
http://www.w3.org/1999/02/22-rdf-syntax-ns#type  "http://www.w3.org/ns/prov#Activity"
prov:Location                                "file://datasets.datalad.org/abide/
↪RawDataBIDS/CMU_a"
-------------------------------------------  ---------------------------------------
↪-------------

subjects
------------------------------------
fdb6c8bc-67aa-11ea-ba45-003ee1ce9545
b276ebb6-67aa-11ea-ba45-003ee1ce9545
a38c4e42-67aa-11ea-ba45-003ee1ce9545
a2ff751c-67aa-11ea-ba45-003ee1ce9545
cfce5728-67aa-11ea-ba45-003ee1ce9545
f165e7ae-67aa-11ea-ba45-003ee1ce9545
cf4605ee-67aa-11ea-ba45-003ee1ce9545
a1efa78c-67aa-11ea-ba45-003ee1ce9545
d0de8ebc-67aa-11ea-ba45-003ee1ce9545
a4a999ba-67aa-11ea-ba45-003ee1ce9545
a0555098-67aa-11ea-ba45-003ee1ce9545
b41d75f2-67aa-11ea-ba45-003ee1ce9545
be3fbff0-67aa-11ea-ba45-003ee1ce9545
eec5a0ca-67aa-11ea-ba45-003ee1ce9545

data_elements
------------------------------
SCQ_TOTAL
VIQ
...
WISC_IV_PIC_CON_SCALED
SRS_MOTIVATION

subject                              field       datumType    label                ␣
↪value  units
-----------------------------------  ----------  -----------  -------------------  ----
↪---  -------
fdb6c8bc-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1630  mm^3
b276ebb6-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪2062  mm^3
a38c4e42-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
```

(continues on next page)

(continued from previous page)

```
↪1699  mm^3
a2ff751c-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1791  mm^3
cfce5728-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪2017  mm^3
f165e7ae-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪2405  mm^3
cf4605ee-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪2062  mm^3
a1efa78c-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1961  mm^3
d0de8ebc-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1568  mm^3
a4a999ba-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1948  mm^3
a0555098-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1764  mm^3
b41d75f2-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪2031  mm^3
be3fbff0-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1935  mm^3
eec5a0ca-67aa-11ea-ba45-003ee1ce9545  fsl_000012  ilx_0738276  Left-Pallidum (mm^3)    ␣
↪1806  mm^3
```

Get the subjects in a project

```
pynidm query -nl "cmu_a.nidm.ttl" -u http://localhost:5000/projects/dc1bf9be-10a3-11ea-
↪8779-003ee1ce9545/subjects
```

Example response:

```
deef8eb2-10a3-11ea-8779-003ee1ce9545
df533e6c-10a3-11ea-8779-003ee1ce9545
ddbfb454-10a3-11ea-8779-003ee1ce9545
df21cada-10a3-11ea-8779-003ee1ce9545
dcfa35b2-10a3-11ea-8779-003ee1ce9545
de89ce4c-10a3-11ea-8779-003ee1ce9545
dd2ce75a-10a3-11ea-8779-003ee1ce9545
ddf21020-10a3-11ea-8779-003ee1ce9545
debc0f74-10a3-11ea-8779-003ee1ce9545
de245134-10a3-11ea-8779-003ee1ce9545
dd5f2f30-10a3-11ea-8779-003ee1ce9545
dd8d4faa-10a3-11ea-8779-003ee1ce9545
df87cbaa-10a3-11ea-8779-003ee1ce9545
de55285e-10a3-11ea-8779-003ee1ce9545
```

Use the command line to get statistics on a project for the AGE_AT_SCAN and a FSL data element

```
pynidm query -nl ttl/cmu_a.nidm.ttl -u /statistics/projects/dc1bf9be-10a3-11ea-8779-
↪003ee1ce9545?fields=instruments.AGE_AT_SCAN,derivatives.fsl_000001
```

Example response:

```
------------------------------------------------  ---------------------------------------
↪-------
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"  http://www.w3.org/ns/prov#Activity
"title"                                            ABIDE CMU_a Site
"Identifier"                                       1.0.1
"prov:Location"                                    /datasets.datalad.org/abide/
↪RawDataBIDS/CMU_a
"NIDM_0000171"                                     14
"age_max"                                          33.0
"age_min"                                          21.0

  gender
--------
       1
       2

handedness
------------
R
L
Ambi

subjects
------------------------------------
de89ce4c-10a3-11ea-8779-003ee1ce9545
deef8eb2-10a3-11ea-8779-003ee1ce9545
dd8d4faa-10a3-11ea-8779-003ee1ce9545
ddbfb454-10a3-11ea-8779-003ee1ce9545
de245134-10a3-11ea-8779-003ee1ce9545
debc0f74-10a3-11ea-8779-003ee1ce9545
dd5f2f30-10a3-11ea-8779-003ee1ce9545
ddf21020-10a3-11ea-8779-003ee1ce9545
dcfa35b2-10a3-11ea-8779-003ee1ce9545
df21cada-10a3-11ea-8779-003ee1ce9545
df533e6c-10a3-11ea-8779-003ee1ce9545
de55285e-10a3-11ea-8779-003ee1ce9545
df87cbaa-10a3-11ea-8779-003ee1ce9545
dd2ce75a-10a3-11ea-8779-003ee1ce9545

-----------  ------------------  --------
AGE_AT_SCAN  max                 33
AGE_AT_SCAN  min                 21
AGE_AT_SCAN  median              26
AGE_AT_SCAN  mean                26.2857
AGE_AT_SCAN  standard_deviation   4.14778
-----------  ------------------  --------
```

(continues on next page)

(continued from previous page)

```
----------  ------------------  -----------
fsl_000001  max                 1.14899e+07
fsl_000001  min                 5.5193e+06
fsl_000001  median              7.66115e+06
fsl_000001  mean                8.97177e+06
fsl_000001  standard_deviation  2.22465e+06
----------  ------------------  -----------
```

Get details on a subject

Use `-j` for a JSON formatted response

```
pynidm query -j -nl "cmu_a.nidm.ttl" -u http://localhost:5000/projects/dc1bf9be-10a3-
↪11ea-8779-003ee1ce9545/subjects/df21cada-10a3-11ea-8779-003ee1ce9545
```

Example response:


```
 {
"uuid": "df21cada-10a3-11ea-8779-003ee1ce9545",
"id": "0050665",
"activity": [
  "e28dc764-10a3-11ea-a7d3-003ee1ce9545",
  "df28e95a-10a3-11ea-8779-003ee1ce9545",
  "df21c76a-10a3-11ea-8779-003ee1ce9545"
],
"instruments": {
  "e28dd218-10a3-11ea-a7d3-003ee1ce9545": {
    "SRS_VERSION": "nan",
    "ADOS_MODULE": "nan",
    "WISC_IV_VCI": "nan",
    "WISC_IV_PSI": "nan",
    "ADOS_GOTHAM_SOCAFFECT": "nan",
    "VINELAND_PLAY_V_SCALED": "nan",
    "null": "http://www.w3.org/ns/prov#Entity",
    "VINELAND_EXPRESSIVE_V_SCALED": "nan",
    "SCQ_TOTAL": "nan",
    "SRS_MOTIVATION": "nan",
    "PIQ": "104.0",
    "FIQ": "109.0",
    "WISC_IV_PRI": "nan",
    "FILE_ID": "CMU_a_0050665",
    "VIQ": "111.0",
    "WISC_IV_VOCAB_SCALED": "nan",
    "VINELAND_DAILYLVNG_STANDARD": "nan",
    "WISC_IV_SIM_SCALED": "nan",
    "WISC_IV_DIGIT_SPAN_SCALED": "nan",
    "AGE_AT_SCAN": "33.0"
    }
 },
"derivatives": {
```


(continues on next page)

(continued from previous page)


```
    "b9fe0398-16cc-11ea-8729-003ee1ce9545": {
       "URI": "http://iri.nidash.org/b9fe0398-16cc-11ea-8729-003ee1ce9545",
       "values": {
         "http://purl.org/nidash/fsl#fsl_000005": {
           "datumType": "ilx_0102597",
           "label": "Left-Amygdala (voxels)",
           "value": "1573",
           "units": "voxel"
         },
         "http://purl.org/nidash/fsl#fsl_000004": {
           "datumType": "ilx_0738276",
           "label": "Left-Accumbens-area (mm^3)",
           "value": "466.0",
           "units": "mm^3"
         },
         "http://purl.org/nidash/fsl#fsl_000003": {
           "datumType": "ilx_0102597",
           "label": "Left-Accumbens-area (voxels)",
           "value": "466",
           "units": "voxel"
         }
       },
       "StatCollectionType": "FSLStatsCollection"
    }
}
```


7.5 Indices and tables

- genindex
- modindex
- search