Neo4j Database Creation

1.0

Generated by Doxygen 1.9.8

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 Author.Author Class Reference

Public Member Functions

- remove_duplicate_authors_and_collabs ()
- upload_current_gt_authors (url, username, password, database)
- upload_collabs_parallel (batch_number, url, username, password, database, batch_size=8000)
- get_max_batch_number (batch_size=8000)
- test_no_duplicates_in_author_and_collab ()
- check_no_lost_collabs ()
- run_method (arguments)

2.1.1 Detailed Description

```
A class to hold various methods relating to authors.

Imports
-------
Uses the File and Database classes from this project
json
sys

Methods
-------
remove_duplicate_authors_and_collabs():
    Removes any duplicated authors and collaborators from the author and collab json files
    Merges authors which appear in both files so that only one node will be created
upload_current_gt_authors(url, username, password, database):
    Uploads authors within the author json file
upload_collabs_parallel(batch_number, url, username, password, database, batch_size=8000):
    Uploads collabs within the collab json file, set up to make use of a slurm array job
get_max_batch_number(batch_size=8000):
    Prints the length of the array used when uploading the collabs in parallel
    Use to determing correct size of array for an array job
    Prints the result to output file
test_no_duplicates_in_author_and_collab():
    A test to determine if there are any authors which appear both as an author and a collab,
    meaning they were not propery merged
    Prints the result to output file
check_no_lost_collabs():
    A test to determine if every collab in the original collab json is also in the edited version
    Prints the result to output file
run_method(arguments):
    Used to run any method in the class to easily access methods in an sbatch file
```

2.1.2 Member Function Documentation

2.1.2.1 check_no_lost_collabs()

```
Author.Author.check_no_lost_collabs ( )
```

A test to determine if every collab in the original collab json is also in the edited version
Prints the result to output file

Requirements
---
openalex_collab_dump.json
openalex_collab_dump_edit.json

Parameters

None

Returns

None

2.1.2.2 get_max_batch_number()

```
Author.Author.get_max_batch_number (
              batch_size = 8000 )
```

Prints the length of the array used when uploading the collabs in parallel
Use to determine correct size of array for an array job
Prints the result to output file

Requirements
---
openalex_collab_dump_edit.json

Parameters
---
batch_size: int (optional)
    Represents the size of each batch, the number of nodes to be uploaded in each sub-job
    If the size is too large, Neo4j will run out of memory and the upload will not complete

Returns

None

2.1.2.3 remove_duplicate_authors_and_collabs()

```
Author.Author.remove_duplicate_authors_and_collabs ( )
```

Removes any duplicated authors and collaborators from the author and collab json files
Merges authors which appear in both files so that only one node will be created
When merging nodes, combines the past institution lists to capture all institutions an author worked at
Does not include GT in a current GT author's past institutions, as this will be represented through a WORKING_AT relationship

Requirements
---
openalex_author_dump.json

```
openalex_collab_dump.json
openalex_work_dump.json
openalex_institution_dump.json

Parameters
----------
None

Returns
-------
None
```

2.1.2.4 run_method()

```
Author.Author.run_method (
              arguments )


Used to run any method in the class to easily access methods in an sbatch file

Requirements
------------
None

Parameters
----------
arguments: list
    List of methods to execute
    ex. ['Author.check_no_lost_collabs()']

Returns
-------
None
```

2.1.2.5 test_no_duplicates_in_author_and_collab()

```
Author.Author.test_no_duplicates_in_author_and_collab ( )


A test to determine if there are any authors which appear both as an author and a collab,
meaning they were not propery merged
Prints the result to output file

Requirements
------------
openalex_author_dump_edit.json
openalex_collab_dump_edit.json

Parameters
----------
None

Returns
-------
None
```

2.1.2.6 upload_collabs_parallel()

```
Author.Author.upload_collabs_parallel (
              batch_number,
              url,
              username,
              password,
              database,
              batch_size = 8000 )
```

```
Uploads collabs within the collab json file, set up to make use of a slurm array job
Adds the WORKED_AT relationship with past institutions

Requirements
------------
openalex_collab_dump_edit.json

Parameters
----------
batch_number: int
    Represents which batch to upload, based on batch_size
    with the default number for batch_size (8000):
    A batch number of 0 uploads the first 8000, 1 the next 8000, etc
url: str
    URL of the database to connect and upload the nodes to
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to upload to
batch_size: int (optional)
    Represents the size of each batch, the number of nodes to be uploaded in each sub-job
    If the size is too large, Neo4j will run out of memory and the upload will not complete
    Default value: 8000

Returns
-------
None
```

2.1.2.7 upload_current_gt_authors()

```
Author.Author.upload_current_gt_authors (
              url,
              username,
              password,
              database )
```

```
Uploads authors within the author json file
Gives all current gt authors both the Author and GT label
Creates a WORKING_AT relationship with GT

Requirements
------------
openalex_author_dump_edit.json

Parameters
----------
url: str
    URL of the database to connect and upload the nodes to
username: str
    Username of Neo4j user
password: str
```

```
    Password of Neo4j user
database: str
    Specific database to upload to

Returns
-------
None
```

The documentation for this class was generated from the following file:

- Author.py

2.2 DataBase.DataBase Class Reference

Public Member Functions

- __init__ (self, url, username, password, database)
- run_command (self, command, additional_data=None)

Public Attributes

- **url**
- **username**
- **password**
- **database**
- **driver**

2.2.1 Detailed Description

```
A class which represents a connection to a Neo4j database

Imports
-------
GraphDatabase from neo4j

Attributes
----------
url: str
    URL of the database to establish a connection with
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to upload to
driver: GraphDatabase.driver
    Established connection to Neo4j based on above attributes

Methods
-------
run_command(command, additional_data=None):
    Runs a given cypher command on the database
```

2.2.2 Constructor & Destructor Documentation

2.2.2.1 __init__()

```
DataBase.DataBase.__init__ (
              self,
              url,
              username,
              password,
              database )
```

```
Constructs attributes for a DataBase object and establishes a driver

Parameters
----------
url: str
    URL of the database to connect to
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to connect to
```

2.2.3 Member Function Documentation

2.2.3.1 run_command()

```
DataBase.DataBase.run_command (
              self,
              command,
              additional_data = None )
```

```
Runs a given cypher command on the database

Requirements
------------
None

Parameters
----------
command: str
    Cypher command to run on the Neo4j database
additional_data: list, optional
    Additional data to give to Neo4j
    Often a list of nodes to add
    Default value: None

Returns
-------
None
```

The documentation for this class was generated from the following file:

- DataBase.py

2.3 Download.Download Class Reference

Public Member Functions

- __init__ (self)
- download_based_on_last_known (self)
- download_based_on_works (self)
- add_author (self, id)
- add_work (self, id)
- create_files (self)
- close_files (self)

Public Attributes

- **added_institutes**
- **added_works**
- **missed_authors**
- **missed_works**

2.3.1 Detailed Description

```
A class for an openalex download

Imports
-------
json
requests
os
sys

Methods
-------
download_based_on_las_known():
    Download all authors into json files based on last known institution as Georgia Tech
download_based_on_works():
    Filters openalex works based on GT and collects necessary information
add_author(id):
    Adds a single author and their relevant information
add_work(id):
    Adds a single work and relevant information
create_files():
    Creates 4 json files to collect information
close_files():
    Closes the files so that they're in proper json format
```

2.3.2 Constructor & Destructor Documentation

2.3.2.1 __init__()

```
Download.Download.__init__ (
              self )


Constructs attributes for a SlurmJob object

Parameters
----------
None
```

2.3.3 Member Function Documentation

2.3.3.1 add_author()

```
Download.Download.add_author (
              self,
              id )
```

```
    Adds a single author and their relevant information
    Puts authors in openalex_author_dump.json
    Puts their collaborators in openalex_collab_dump.json
    Puts their institutions in openalex_institution_dump.json
    Puts their works in openalex_work_dump.json

    Requirements
    ------------
    None

    Parameters
    ----------
    id: str
        Openalex id of the author to be added

    Returns
    -------
    None
```

2.3.3.2 add_work()

```
Download.Download.add_work (
              self,
              id )
```

```
    Adds a single work and relevant information
    Puts all collaborators in openalex_collab_dump.json
    Puts their institutions in openalex_institution_dump.json
    Puts their works in openalex_work_dump.json

    Requirements
    ------------
    None

    Parameters
    ----------
    id: str
        Openalex id of the work to be added

    Returns
    -------
    None
```

2.3.3.3 close_files()

```
Download.Download.close_files (
              self )
```

```
Closes the files so that they're in proper json format

Requirements
------------
openalex_author_dump.json
openalex_collab_dump.json
openalex_institution_dump.json
openalex_work_dump.json

Parameters
----------
None

Returns
-------
None
```

2.3.3.4 create_files()

```
Download.Download.create_files (
              self )


    Creates 4 json files to collect information
    openalex_author_dump.json
    openalex_collab_dump.json
    openalex_institution_dump.json
    openalex_work_dump.json

    Requirements
    ------------
    None

    Parameters
    ----------
    None

    Returns
    -------
    None
```

2.3.3.5 download_based_on_last_known()

```
Download.Download.download_based_on_last_known (
              self )


    Download all authors into json files based on last known institution as Georgia Tech
    Puts authors in openalex_author_dump.json
    Puts their collaborators in openalex_collab_dump.json
    Puts their institutions in openalex_institution_dump.json
    Puts their works in openalex_work_dump.json

    Requirements
    ------------
    None

    Parameters
    ----------
    None

    Returns
    -------
    None
```

2.3.3.6 download_based_on_works()

```
Download.Download.download_based_on_works (
              self )
```

```
    Filters openalex works based on GT and collects necessary information
    Puts all collaborators in openalex_collab_dump.json
    Puts their institutions in openalex_institution_dump.json
    Puts their works in openalex_work_dump.json

    Requirements
    ------------
    None

    Parameters
    ----------
    None

    Returns
    -------
    None
```

The documentation for this class was generated from the following file:

- Download.py

2.4 File.File Class Reference

Public Member Functions

- close_file (file_name)
- chunk_json (self, file_name, chunk_size)
- open_file (file_name)
- write_data_to_file (file_name, data)
- create_json_file (file_name, data)
- check_num_in_json (file_name)
- compare_to_edit (original_file, edit_file)
- check_for_duplicates (edit_file)
- get_max_batch_number_works (self, batch_size=8000)
- get_max_batch_number_collabs (self, batch_size=8000)
- check_files (original=True, edit=True)
- run_method (arguments)

2.4.1 Detailed Description

```
A class to hold various methods relating to files

Imports
-------
os
json
sys

Methods
-------
close_file(file_name):
```

```
    Adds the necessary ending to a given json file once all data has been imported
chunk_json(filename, chunk_size):
    Separates the json file into smaller chunks and returns them in the form of a list
open_file(file_name):
    Adds the necessary beginning to a new json file before adding the data
write_data_to_file(file_name, data):
    Writes each item in a given list to the given json file
create_json_file(file_name, data):
    Combines methods to create a new json file with the given data in the correct format.
check_num_in_json(file_name):
    Prints the number of items inside a given json file
compare_to_edit(original_file, edit_file):
    Ensures all entries in the original json are also in the edited json
check_for_duplicates(edit_file):
    Checks for duplicates in the given file
get_max_batch_number_works(batch_size=8000):
    Prints length of chunked list for works json
get_max_batch_number_collabs(batch_size=8000):
    Prints length of chunked list for the collab json
check_files(original=True, edit=True):
    Checks to see if certain files exist within the current directory
run_method(arguments):
    Used to run any method in the class to easily access methods in an sbatch file
```

2.4.2 Member Function Documentation

2.4.2.1 check_files()

```
File.File.check_files (
              original = True,
              edit = True )
```

```
Checks to see if certain files exist within the current directory

Requirements
------------
None

Parameters
----------
original: boolean, optional
    If true, then the method will make sure all four original json files are in the current
    directory
    Default value: True
edit: boolean, optional
    If true, then the method will make sure all four edited json files are in the current
    directory
    Default value: True

Returns
-------
boolean representing if the specific files were located or not
```

2.4.2.2 check_for_duplicates()

```
File.File.check_for_duplicates (
              edit_file )
```

```
Checks for duplicates in the given file
Prints result to output file

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use

Returns
-------
None
```

2.4.2.3 check_num_in_json()

```
File.File.check_num_in_json (
              file_name )


Prints the number of items inside a given json file
Prints result to output file

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use

Returns
-------
None
```

2.4.2.4 chunk_json()

```
File.File.chunk_json (
              self,
              file_name,
              chunk_size )


Separates the json file into smaller chunks and returns them in the form of a list
Used to run methods in parallel

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use
chunk_size: int
    Nummber of items in each chunk

Returns
-------
list of items in the original file separated into chunks of the given size
```

2.4.2.5 close_file()

```
File.File.close_file (
              file_name )

Adds the necessary ending to a given json file once all data has been imported

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use

Returns
-------
None
```

2.4.2.6 compare_to_edit()

```
File.File.compare_to_edit (
              original_file,
              edit_file )

Ensures all entries in the original json are also in the edited json
Prints result to output file

Requirements
------------
None

Parameters
----------
original_file: str
    Name of original file in current directory to use
edit_file: str
    Name of edited file in current directory to use

Returns
-------
None
```

2.4.2.7 create_json_file()

```
File.File.create_json_file (
              file_name,
              data )

Combines methods to create a new json file with the given data in the correct format.

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use
data:
    List of data to be written to json file

Returns
-------
None
```

2.4.2.8 get_max_batch_number_collabs()

```
File.File.get_max_batch_number_collabs (
              self,
              batch_size = 8000 )

Prints length of chunked list for collabs json
Use to determine correct size of array for an array job

Requirements
------------
openalex_collab_dump_edit.json

Parameters
----------
batch_size: int, optional
    Number of entries to be in each chunk
    Default value: 8000

Returns
-------
Length of the chunked json
```

2.4.2.9 get_max_batch_number_works()

```
File.File.get_max_batch_number_works (
              self,
              batch_size = 8000 )

Prints length of chunked list for works json
Use to determine correct size of array for an array job

Requirements
------------
openalex_work_dump_edit.json

Parameters
----------
batch_size: int, optional
    Number of entries to be in each chunk
    Default value: 8000

Returns
-------
Length of the chunked json
```

2.4.2.10 open_file()

```
File.File.open_file (
              file_name )

Adds the necessary beginning to a new json file before adding the data

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use

Returns
-------
None
```

2.4.2.11 run_method()

```
File.File.run_method (
              arguments )
```

```
Used to run any method in the class to easily access methods in an sbatch file

Requirements
------------
None

Parameters
----------
arguments: list
    List of methods to execute

Returns
-------
None
```

2.4.2.12 write_data_to_file()

```
File.File.write_data_to_file (
              file_name,
              data )
```

```
Writes each item in a given list to the given json file

Requirements
------------
None

Parameters
----------
file_name: str
    Name of file in the current directory to use
data:
    List of data to be written to json file

Returns
-------
None
```

The documentation for this class was generated from the following file:

- File.py

2.5 Institution.Institution Class Reference

Public Member Functions

- remove_duplicate_institutions ()
- upload_institutions (url, username, password, database)
- run_method (arguments)

2.5.1 Detailed Description

```
A class to hold various methods relating to institutions

Imports
-------
Uses the File and Database classes from this project
json
sys

Methods
-------
remove_duplicate_institutions():
    Removes duplicate institutions and creates a new json file without the duplicates
upload_institutions(url, username, password, database):
    Uploads institutions within the json file
run_medthod(arguments):
    Used to run any method in the class to easily access methods in an sbatch file
```

2.5.2 Member Function Documentation

2.5.2.1 remove_duplicate_institutions()

```
Institution.Institution.remove_duplicate_institutions ( )


Removes duplicate institutions and creates a new json file without the duplicates

Requirements
------------
openalex_institution_dump.json

Parameters
----------
None

Returns
-------
None
```

2.5.2.2 run_method()

```
Institution.Institution.run_method (
              arguments )


Used to run any method in the class to easily access methods in an sbatch file

Requirements
------------
None

Parameters
----------
arguments: list
    List of methods to execute

Returns
-------
None
```

2.5.2.3 upload_institutions()

```
Institution.Institution.upload_institutions (
              url,
              username,
              password,
              database )


 Uploads institutions within the json file

Requirements
------------
openalex_institution_dump_edit.json

Parameters
----------
url: str
    URL of the database to connect and upload the nodes to
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to upload to

Returns
-------
None
```

The documentation for this class was generated from the following file:

- Institution.py

2.6 SlurmJob.SlurmJob Class Reference

Public Member Functions

- __init__ (self, account, python_env_location, num_nodes=1, cores_per_node=4, mem_per_core='7G', job↩ _duration='3:00:00')
- remove_duplicates (self)
- download_files (self)
- upload_to_neo4j (self, url, username, password, database)
- run_tests_on_cleaning (self)
- print_nums_in_file (self)

Public Attributes

- **account**
- **num_nodes**
- **cores_per_node**
- **mem_per_core**
- **job_duration**
- **python_env_location**

2.6.1 Detailed Description

```
A class representing a slurm job

Imports
-------
Uses the File class from this project
os
Popen, PIPE from subprocess

Methods
-------
remove_duplicates():
    Create and submit a job to remove all duplicates from the downloaded files
download_files():
    Create and submit a job to download all files
upload_to_neo4j():
    Create and submit a job to upload all nodes to Neo4j
run_tests_on_cleaning():
    Create and submit a job to run multiple tests on the cleaning portion
print_nums_in_files()
    Create and submit a job to print the number of nodes in each file
__create_slurm_job(file_name, job_name="neo4j"):
    Creates a job with the given characteristics
__create_parallel_slurm_job(file_name, job_name="neo4j, array_size=1):
    Creates a parallel job with the given characteristics
__execute_slurm_job(self, file_name, dependency=None):
    Submits a given job to the cluster
```

2.6.2 Constructor & Destructor Documentation

2.6.2.1 __init__()

```
SlurmJob.SlurmJob.__init__ (
              self,
              account,
              python_env_location,
              num_nodes = 1,
              cores_per_node = 4,
              mem_per_core = '7G',
              job_duration = '3:00:00' )


Constructs attributes for a SlurmJob object

Parameters
----------
account: str
    Account of the pace user
python_env_location: str
    Location of python virtual environment
    If following the docs on submitting a pace job, the location will be:
    /storage/coda1/p-pace-user/0/<USERNAME>/test_installs/neo4j_venv/bin/activate
num_nodes: str, optional
    Number of nodes to use for job
    Default value: 1
cores_per_node: str, optional
    Number of cores per node to use for job
    Default value: 4
mem_per_core: str, optional
    Memory to use per core
    Default value: 7G
job_duration: str, optional
    Time allotted for job
    In format HH:MM:SS
    Default value: 3:00:00
```

2.6.3 Member Function Documentation

2.6.3.1 download_files()

```
SlurmJob.SlurmJob.download_files (
              self )
```

```
Create and submit a job to download all files
Creates an sbatch file to submit: download_files.sbatch
Uses Download.py

Requirements
------------
None

Parameters
----------
None

Returns
-------
None
```

2.6.3.2 print_nums_in_file()

```
SlurmJob.SlurmJob.print_nums_in_file (
              self )
```

```
Create and submit a job to print the number of nodes in each file
Creates the following sbatch files:
counting.sbatch
Uses the following methods:
File.check_num_in_json()
Check output files for results

Requirements
------------
openalex_author_dump_edit.json
openalex_work_dump_edit.json
openalex_collab_dump_edit.json
openalex_institution_dump_edit.json

Parameters
----------
None

Returns
-------
None
```

2.6.3.3 remove_duplicates()

```
SlurmJob.SlurmJob.remove_duplicates (
              self )
```

```
Create and submit a job to remove all duplicates from the downloaded files
Creates an sbatch file to submit: clean_data.sbatch
Uses the following methods:
Author.remove_duplicate_authors_and_collabs()
Institution.remove_duplicate_institutions()
Work.remove_duplicate_works()

Requirements
------------
openalex_author_dump.json
openalex_work_dump.json
openalex_collab_dump.json
openalex_institution_dump.json

Parameters
----------
None

Returns
-------
None
```

2.6.3.4 run_tests_on_cleaning()

```
SlurmJob.SlurmJob.run_tests_on_cleaning (
              self )


Create and submit a job to run multiple tests on the cleaning portion
Creates the following sbatch files:
cleaning_tests.sbatch
Uses the following methods:
File.check_for_duplicates()
File.compare_to_edit
Author.test_no_duplicates_in_author_and_collab
Author.check_no_lost_collabs
Check output files for results

Requirements
------------
openalex_author_dump.json
openalex_work_dump.json
openalex_collab_dump.json
openalex_institution_dump.json
openalex_author_dump_edit.json
openalex_work_dump_edit.json
openalex_collab_dump_edit.json
openalex_institution_dump_edit.json

Parameters
----------
None

Returns
-------
None
```

2.6.3.5 upload_to_neo4j()

```
SlurmJob.SlurmJob.upload_to_neo4j (
              self,
              url,
              username,
              password,
              database )
```

```
Create and submit a job to upload all nodes to Neo4j
Creates the following sbatch files:
upload_inst.sbatch
upload_collabs.sbatch
upload_authors.sbatch
upload_works.sbatch
upload_authored.sbatch
Uses the following methods:
Institution.upload_institutions()
Author.upload_collabs_parallel()
Author.upload_current_gt_authors()
Work.upload_works_parallel()
Work.create_authored_relationship()
Uses job dependencies to ensure the jobs run in that order

Requirements
------------
openalex_author_dump_edit.json
openalex_work_dump_edit.json
openalex_collab_dump_edit.json
openalex_institution_dump_edit.json

Parameters
----------
url: str
    URL of the database to connect and upload the nodes to
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to upload to

Returns
-------
None
```

The documentation for this class was generated from the following file:

- SlurmJob.py

2.7 Work.Work Class Reference

Public Member Functions

- remove_duplicate_works ()
- upload_works_parallel (batch_number, url, username, password, database, batch_size=8000)
- get_max_batch_number (batch_size=8000)
- create_authored_relationship (url, username, password, database)
- run_method (arguments)

2.7.1 Detailed Description

```
A class to hold various methods relating to works

Imports
-------
Uses the File and Database classes from this project
json
sys
```

```
Methods
-------
remove_duplicate_works():
    Removes duplicate works and creates a new json file without the duplicates
upload_works_parallel(batch_number, url, username, password, database, batch_size=8000):
    Uploads works within the work json file, set up to make use of a slurm array job
get_max_batch_number(batch_size=8000):
    Prints the length of the array used when uploading the works in parallel
    Use to determing correct size of array for an array job
    Prints the result to output file
create_authored_relationship(url, username, password, database):
    Creates the authored relationship within the database
run_medthod(arguments):
    Used to run any method in the class to easily access methods in an sbatch file
```

2.7.2 Member Function Documentation

2.7.2.1 create_authored_relationship()

```
Work.Work.create_authored_relationship (
              url,
              username,
              password,
              database )
```

```
Creates the authored relationship within the database

Requirements
------------
None

Parameters
----------
url: str
    URL of the database to connect and upload the nodes to
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to upload to

Returns
-------
None
```

2.7.2.2 get_max_batch_number()

```
Work.Work.get_max_batch_number (
              batch_size = 8000 )
```

```
Prints the length of the array used when uploading the works in parallel
Use to determine correct size of array for an array job
Prints the result to output file

Requirements
------------
openalex_work_dump_edit.json

Parameters
----------
```

```
batch_size: int (optional)
    Represents the size of each batch, the number of nodes to be uploaded in each sub-job
    If the size is too large, Neo4j will run out of memory and the upload will not complete

Returns
-------
None
```

2.7.2.3 remove_duplicate_works()

```
Work.Work.remove_duplicate_works ( )
```

```
Removes duplicate works and creates a new json file without the duplicates

Requirements
------------
openalex_work_dump.json

Parameters
----------
None

Returns
-------
None
```

2.7.2.4 run_method()

```
Work.Work.run_method (
              arguments )
```

```
Used to run any method in the class to easily access methods in an sbatch file

Requirements
------------
None

Parameters
----------
arguments: list
    List of methods to execute

Returns
-------
None
```

2.7.2.5 upload_works_parallel()

```
Work.Work.upload_works_parallel (
              batch_number,
              url,
              username,
              password,
              database,
              batch_size = 8000 )
```

```
Uploads works within the work json file, set up to make use of a slurm array job

Requirements
------------
openalex_work_dump_edit.json

Parameters
----------
batch_number: int
    Represents which batch to upload, based on batch_size
    with the default number for batch_size (8000):
    A batch number of 0 uploads the first 8000, 1 the next 8000, etc
url: str
    URL of the database to connect and upload the nodes to
username: str
    Username of Neo4j user
password: str
    Password of Neo4j user
database: str
    Specific database to upload to
batch_size: int (optional)
    Represents the size of each batch, the number of nodes to be uploaded in each sub-job
    If the size is too large, Neo4j will run out of memory and the upload will not complete
    Default value: 8000

Returns
-------
None
```

The documentation for this class was generated from the following file:

- Work.py

Index