Course: DSC650 Assignment 2.2

Author: Anjani Bonda

Date: 3/25/2023

In [13]:
```
## create a method to load json file
def _load_json(json_path):
    '''loads data from .json file'''
    with open(json_path) as f:
        return json.load(f)
```

In [14]:
```
from pathlib import Path
import json
import os

from tinydb import TinyDB

current_dir = Path(os.getcwd()).absolute()
results_dir = current_dir.joinpath('results')
kv_data_dir = results_dir.joinpath('kvdb')
kv_data_dir.mkdir(parents=True, exist_ok=True)


class DocumentDB(object):
    def __init__(self, db_path):
        ## You can use the code from the previous exmaple if you would like
        people_json = kv_data_dir.joinpath('people.json')
        visited_json = kv_data_dir.joinpath('visited.json')
        sites_json = kv_data_dir.joinpath('sites.json')
        measurements_json = kv_data_dir.joinpath('measurements.json')

        self._db_path = Path(db_path)
        self._db = None
        ## TODO: Implement code
        self.person_lkp = _load_json(people_json)
        self.visit_lkp = _load_json(visited_json)
        self.site_lkp = _load_json(sites_json)
        self.measure_lkp = _load_json(measurements_json)

        self._load_db()

    ## Create a method to get sites based on site_id
    def _get_site(self, site_id):
        '''return sites based on site_id'''
        return self.site_lkp[str(site_id)]

    ## Create a method to get measurements based on person_id
    def _get_measurements(self, person_id):
        '''return measurements based on person_id'''
        measurements = []
        measurements.extend([
            values for values in self.measure_lkp.values()
            if str(values['person_id']) == str(person_id)
        ])
        return measurements

    ## Create a method to get visits based on visit_id
    def _get_visit(self, visit_id):
        '''returns visit info about a specific site visit_id'''

        for key, value in self.visit_lkp.items():
            k = key.replace('(',"").split(",")
            if str(k[0]) == str(visit_id):
                visit = value
        ## call get_sites method based on site_id
        site_id = visit['site_id']
        site = self._get_site(site_id) # retrieve info about site
        visit['site'] = site # Append site info to visit info
        return visit


    def _load_db(self):
        self._db = TinyDB(self._db_path)
        ## TODO: Implement code
        persons = self.person_lkp.items()
        for person_id, record in persons:
            # return individual's list of records:
            measurements = self._get_measurements(person_id)
            # extract set of unique visit_id's from id's in list of measurement
            visit_ids = set([measurement['visit_id'] for measurement in measure
            visits = []
            for visit_id in visit_ids: # iterate through set of individual's vi
                visit = self._get_visit(visit_id) # returns info from visit
                # add measurement info from visit
                visit['measurements'] = [
                    measurement for measurement in measurements
                    if visit_id == measurement['visit_id']
                ]
                visits.append(visit)
            record['visits'] = visits # add visit info to record
            self._db.insert(record)
```

In [15]:
```
db_path = results_dir.joinpath('patient-info.json')
if db_path.exists():
    os.remove(db_path)
try:
    db = DocumentDB(db_path)
except:
    print("The tinyDB creation has failed")
else:
    print("The tinyDB creation has completed successfully")
```

```
The tinyDB creation has completed successfully
```

In [ ]: