In [95]:

```
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
        #creating lookup functions with json files
        self._person_lookup = self._load_json(people_json)
        self._visit_lookup = self._load_json(visited_json)
        self._site_lookup = self._load_json(sites_json)
        self._measurements_lookup = self._load_json(measurements_json)
        #load db with data
        self._load_db()

    #need to include self as a paramater
    def _load_json(self,json_path):
        with open(json_path) as f:
            return json.load(f)

    def _get_site(self, site_id):
        #site_id is originally in a set
        #need to retrieve independent value by using pop()
        return self._site_lookup[str(site_id.pop())]
    def _get_measurements(self, person_id):
        measurements = []
        for values in self._measurements_lookup.values():
            #measurements.extend([value for value in values if str(value['person_
            #TO-DO
            #extend the measurements list by the entire dictionary entry
            if str(values['person_id']) == str(person_id):
                measurements.append(values)
        return measurements
    def _get_visit(self, visit_id):
        visit_values = self._visit_lookup.values()
        for value in visit_values:
            if value['visit_id'] == visit_id:
                visit = value
        #visit has a composite key with site_id
        #TO-DO: site_id
        #search for the site_id within the visit since they are composite keys
        site_id = set([visit['site_id'] for v in visit])
        #TO-DO: site
        #retrieve the site for that specific site_id
        site = self._get_site(site_id)
        #assign the site to the visit
        visit['site'] = site
        return visit
    def _load_db(self):
        self._db = TinyDB(self._db_path)
        persons = self._person_lookup.items()
        for person_id, record in persons:
            measurements = self._get_measurements(person_id)
            visit_ids = set([measurement['visit_id'] for measurement in measureme
            visits = []
            for visit_id in visit_ids:
                visit = self._get_visit(visit_id)
                visit['measurements'] = [
                    measurement for measurement in measurements
                    if visit_id == measurement['visit_id']
                ]
                visits.append(visit)
            record['visits'] = visits
            self._db.insert(record)
```

In [96]:

```
db_path = results_dir.joinpath('patient-info.json')
if db_path.exists():
    os.remove(db_path)

db = DocumentDB(db_path)
```

In [ ]: