In [1]:
```
from pathlib import Path
import os
import sqlite3

import s3fs
import pandas as pd

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)


def read_cluster_csv(file_path, endpoint_url='https://storage.budsc.midwest-datas
    s3 = s3fs.S3FileSystem(
        anon=True,
        client_kwargs={
            'endpoint_url': endpoint_url
        }
    )
    return pd.read_csv(s3.open(file_path, mode='rb'))
```

Create and Load Measurements Table

In [2]:
```
def create_measurements_table(conn):
    sql = """
    CREATE TABLE IF NOT EXISTS measurements (
        visit_id integer NOT NULL,
        person_id text NOT NULL,
        quantity text,
        reading real,
        FOREIGN KEY (visit_id) REFERENCES visits (visit_id),
        FOREIGN KEY (person_id) REFERENCES people (people_id)
        );
    """

    c = conn.cursor()
    c.execute(sql)

def load_measurements_table(conn):
    create_measurements_table(conn)
    df = read_cluster_csv('data/external/tidynomicon/measurements.csv')
    measurements = df.values
    c = conn.cursor()
    c.execute('DELETE FROM measurements;') # Delete data if exists
    c.executemany('INSERT INTO measurements VALUES (?,?,?,?)', measurements)
```

Create and Load People Table

In [3]:
```
def create_people_table(conn):
    #generate SQL for people table based on fields in csv
    sql = """
    CREATE TABLE IF NOT EXISTS people(
    person_id text PRIMARY KEY,
    personal_name text,
    family_name text
    );
    """
    ## TODO: Complete SQL
    c = conn.cursor()
    c.execute(sql)

def load_people_table(conn):
    create_people_table(conn)
    ## TODO: Complete code
    df = read_cluster_csv('data/external/tidynomicon/person.csv')
    persons = df.values
    c = conn.cursor()
    c.execute('DELETE FROM people;') # Delete data if exists
    #three empty values
    c.executemany('INSERT INTO people VALUES (?,?,?)', persons)
```

Create and Load Sites Table

In [4]:
```
def create_sites_table(conn):
    sql = """
    CREATE TABLE IF NOT EXISTS sites (
        site_id text PRIMARY KEY,
        latitude double NOT NULL,
        longitude double NOT NULL
        );
    """

    c = conn.cursor()
    c.execute(sql)

def load_sites_table(conn):
    create_sites_table(conn)
    ## TODO: Complete code
    df = read_cluster_csv('data/external/tidynomicon/site.csv')
    sites = df.values
    c = conn.cursor()
    c.execute('DELETE FROM sites;') # Delete data if exists
    #three empty values
    c.executemany('INSERT INTO sites VALUES (?,?,?)', sites)
```

Create and Load Visits Table

In [5]:
```
def create_visits_table(conn):
    sql = """
    CREATE TABLE IF NOT EXISTS visits (
        visit_id integer PRIMARY KEY,
        site_id text NOT NULL,
        visit_date text,
        FOREIGN KEY (site_id) REFERENCES sites (site_id)
        );
    """

    c = conn.cursor()
    c.execute(sql)

def load_visits_table(conn):
    create_visits_table(conn)
    ## TODO: Complete code
    df = read_cluster_csv('data/external/tidynomicon/visited.csv')
    visits = df.values
    c = conn.cursor()
    c.execute('DELETE FROM visits;') # Delete data if exists
    #three empty values
    c.executemany('INSERT INTO visits VALUES (?,?,?)', visits)
```

Create DB and Load Tables

In [7]:
```
db_path = results_dir.joinpath('patient-info.db')
conn = sqlite3.connect(str(db_path))
# TODO: Uncomment once functions completed
load_people_table(conn)
load_sites_table(conn)
load_visits_table(conn)
load_measurements_table(conn)

conn.commit()
#selecting from tables for testing
cur = conn.cursor()
#pulling table names
print("--------------------Tables in the database------------------")
cur.execute('SELECT name from sqlite_master where type = "table"')
print(cur.fetchall())
print("------------------------------------------------------------")

#testing join between tables
cur2 = conn.cursor()
print("------------------Join between all tables-------------------")
cur2.execute("""
SELECT p.*,m.*,v.*,s.*
FROM people as p
INNER JOIN measurements as m
ON p.person_id = m.person_id
INNER JOIN visits as v
on m.visit_id = v.visit_id
INNER JOIN sites as s
on s.site_id = v.site_id
""")
rows = cur2.fetchall()

for row in rows:
    print(row)
print("------------------------------------------------------------")
#end
conn.close()
```

```
--------------------Tables in the database------------------
[('people',), ('sites',), ('visits',), ('measurements',)]
------------------------------------------------------------
------------------Join between all tables-------------------
('dyer', 'William', 'Dyer', 619, 'dyer', 'rad', 9.82, 619, 'DR-1', '1927-02-0
8', 'DR-1', -49.85, -128.57)
('dyer', 'William', 'Dyer', 619, 'dyer', 'sal', 0.13, 619, 'DR-1', '1927-02-0
8', 'DR-1', -49.85, -128.57)
('dyer', 'William', 'Dyer', 622, 'dyer', 'rad', 7.8, 622, 'DR-1', '1927-02-10',
'DR-1', -49.85, -128.57)
('dyer', 'William', 'Dyer', 622, 'dyer', 'sal', 0.09, 622, 'DR-1', '1927-02-1
0', 'DR-1', -49.85, -128.57)
('pb', 'Frank', 'Pabodie', 734, 'pb', 'rad', 8.41, 734, 'DR-3', '1930-01-07',
'DR-3', -47.15, -126.72)
('lake', 'Anderson', 'Lake', 734, 'lake', 'sal', 0.05, 734, 'DR-3', '1930-01-0
7', 'DR-3', -47.15, -126.72)
('pb', 'Frank', 'Pabodie', 734, 'pb', 'temp', -21.5, 734, 'DR-3', '1930-01-07',
'DR-3', -47.15, -126.72)
```

```
('pb', 'Frank', 'Pabodie', 735, 'pb', 'rad', 7.22, 735, 'DR-3', '1930-01-12',
'DR-3', -47.15, -126.72)
('pb', 'Frank', 'Pabodie', 735, 'pb', 'sal', 0.06, 735, 'DR-3', '1930-01-12',
'DR-3', -47.15, -126.72)
('pb', 'Frank', 'Pabodie', 735, 'pb', 'temp', -26.0, 735, 'DR-3', '1930-01-12',
'DR-3', -47.15, -126.72)
('pb', 'Frank', 'Pabodie', 751, 'pb', 'rad', 4.35, 751, 'DR-3', '1930-02-26',
'DR-3', -47.15, -126.72)
('pb', 'Frank', 'Pabodie', 751, 'pb', 'temp', -18.5, 751, 'DR-3', '1930-02-26',
'DR-3', -47.15, -126.72)
('lake', 'Anderson', 'Lake', 752, 'lake', 'rad', 2.19, 752, 'DR-3', None, 'DR-
3', -47.15, -126.72)
('lake', 'Anderson', 'Lake', 752, 'lake', 'sal', 0.09, 752, 'DR-3', None, 'DR-
3', -47.15, -126.72)
('lake', 'Anderson', 'Lake', 752, 'lake', 'temp', -16.0, 752, 'DR-3', None, 'DR
-3', -47.15, -126.72)
('roe', 'Valentina', 'Roerich', 752, 'roe', 'sal', 41.6, 752, 'DR-3', None, 'DR
-3', -47.15, -126.72)
('lake', 'Anderson', 'Lake', 837, 'lake', 'rad', 1.46, 837, 'MSK-4', '1932-01-1
4', 'MSK-4', -48.87, -123.4)
('lake', 'Anderson', 'Lake', 837, 'lake', 'sal', 0.21, 837, 'MSK-4', '1932-01-1
4', 'MSK-4', -48.87, -123.4)
('roe', 'Valentina', 'Roerich', 837, 'roe', 'sal', 22.5, 837, 'MSK-4', '1932-01
-14', 'MSK-4', -48.87, -123.4)
('roe', 'Valentina', 'Roerich', 844, 'roe', 'rad', 11.25, 844, 'DR-1', '1932-03
-22', 'DR-1', -49.85, -128.57)
------------------------------------------------------------
```

In [ ]: