Week2_A3

March 29, 2021

1 NOpitz DSC650 Week2 Assignment 2_3

[20]:
```
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-datascience.com'):
    s3 = s3fs.S3FileSystem(
        anon=True,
        client_kwargs={
            'endpoint_url': endpoint_url
        }
    )
    return pd.read_csv(s3.open(file_path, mode='rb'))
```

2 Create and Load Measurements Table

[29]:
```
def create_measurements_table(conn):
    sql ="""
    CREATE TABLE IF NOT EXISTS measurements (
    visitc.close()_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 (person_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)
```

3 Create and Load People Table

[30]:
```
def create_people_table(conn):
    sql = """
    CREATE TABLE IF NOT EXISTS people (
        person_id text  PRIMARY KEY,
        personal_name text,
        family_name text,
        FOREIGN KEY (person_id) REFERENCES sites (person_id)
        );
    """

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

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

4 Create and Load Sites Table

[31]:
```
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)
    df = read_cluster_csv('data/external/tidynomicon/site.csv')
    sites = df.values
    c = conn.cursor()
    c.execute('DELETE FROM sites;') # Delete data if exists
    c.executemany('INSERT INTO sites VALUES (?,?,?)', sites)
```

5 Create and Load Visits Table

[32]:
```
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)

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

6 Create DB and Load Tables

[33]:
```
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()
conn.close()
```

```
---------------------------------------------------------------------------
OperationalError                          Traceback (most recent call last)
<ipython-input-33-7805012a53f2> in <module>
      2 conn = sqlite3.connect(str(db_path))
      3 # TODO: Uncomment once functions completed
----> 4 load_people_table(conn)
      5 load_sites_table(conn)
      6 load_visits_table(conn)

<ipython-input-30-d2cb9f36ab27> in load_people_table(conn)
     18     c = conn.cursor()
     19     #c.execute('DELETE FROM people;') # Delete data if exists
---> 20     c.executemany('INSERT INTO people VALUES (?,?,?)', people)

OperationalError: database is locked
```