Record of Successful Database Operations

This notebook contains the CRUD functions created in the `mylib` directory, and tests each to prove that they successfully operated.

In [11]:
```
# extract.py
import requests
import os


def extract(
    url="https://s3.amazonaws.com/dl.ncsbe.gov/ENRS/2020_11_03/polling_place
    filepath="data/pollingplaces_2020.csv",
    directory="data",
):
    """Extract to file path"""
    if not os.path.exists(directory):
        os.makedirs(directory)
    with requests.get(url, timeout=5) as r:
        with open(filepath, "wb") as f:
            f.write(r.content)
    return filepath


if __name__ == "__main__":
    if os.path.exists("/Users/pdeguz01/Documents/git/PeterdeGuzman_Mini5"):
        os.chdir("/Users/pdeguz01/Documents/git/PeterdeGuzman_Mini5")
    else:
        print("Directory does not exist.")

    extract()
```

In [12]:
```
# transform

import sqlite3
import csv


def load(dataset="./data/pollingplaces_2020.csv"):
    data = open(dataset, newline="", encoding="utf-16")
    # NCSBE data includes null bytes, which must be removed
    payload = csv.reader((line.replace("\0", "") for line in data), delimite
    conn = sqlite3.connect("pollingplaces_2020.db")
    c = conn.cursor()
    # generate new table for the database
    c.execute("DROP TABLE IF EXISTS pollingplaces_2020")
    c.execute(
        """
            CREATE TABLE pollingplaces_2020 (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            election_dt DATE,
            county_name TEXT,
```

```
            polling_place_id INTEGER,
            polling_place_name TEXT,
            precinct_name TEXT,
            house_num INTEGER,
            street_name TEXT,
            city TEXT,
            state TEXT,
            zip TEXT)
            """
    )
    # insert values
    c.executemany(
        """
            INSERT INTO pollingplaces_2020 (
            election_dt,
            county_name,
            polling_place_id,
            polling_place_name,
            precinct_name,
            house_num,
            street_name,
            city,
            state,
            zip)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
            """,
        payload,
    )
    conn.commit()
    conn.close()
    return "pollingplaces_2020.db"


if __name__ == "__main__":
    load()
```

In [44]:

```
def query_create():
    conn = sqlite3.connect("pollingplaces_2020.db")
    cursor = conn.cursor()
    # create query
    cursor.execute(
        """
        INSERT INTO pollingplaces_2020
        (election_dt,county_name,polling_place_id, polling_place_name, preci
        house_num, street_name, city, state,zip)
        VALUES(11/03/2020, 'DURHAM', 99, 'GROSS HALL', 'DUKE MIDS',
        140, 'SCIENCE DRIVE', 'DURHAM', 'NC', '27708')
        """
    )
    conn.commit()
    conn.close()
    return "Create Success"


def query_read():
    conn = sqlite3.connect("pollingplaces_2020.db")
```

```
    cursor = conn.cursor()
    # execute read
    cursor.execute("SELECT * FROM pollingplaces_2020 LIMIT 10")
    conn.close()
    return "Read Success"


def query_update():
    conn = sqlite3.connect("pollingplaces_2020.db")
    cursor = conn.cursor()
    # update
    cursor.execute("UPDATE pollingplaces_2020 SET county_name = 'DURHAM' WHE
    conn.commit()
    conn.close()
    return "Update Success"


def query_delete():
    conn = sqlite3.connect("pollingplaces_2020.db")
    cursor = conn.cursor()
    # delete
    cursor.execute("DELETE FROM pollingplaces_2020 WHERE id = 10")
    conn.commit()
    conn.close()
    return "Delete Success"
```

First, I create a simple test to determine if the `query_create()` function works properly.

In [40]:
```
query_create()

# Identifying if a polling place with precinct_name "DUKE MIDS" was actually
conn = sqlite3.connect("pollingplaces_2020.db")
cursor = conn.cursor()
query = "SELECT COUNT(*) FROM pollingplaces_2020 WHERE precinct_name = ?"
cursor.execute(query, ("DUKE MIDS",))
exists = cursor.fetchone()[0] > 0

if exists:
    print(
        "A row with precinct_name 'DUKE MIDS' exists, and therefore the quer
    )
else:
    print("No row with precinct_name 'DUKE MIDS' found.")

cursor.close()
conn.close()
```

```
A row with precinct_name 'DUKE MIDS' exists, and therefore the query_create
() function worked successfully.
```

Next, I create a simple test to determine if the `query_update()` function works properly.

In [46]:
```
query_update()
# Identifying if the county_name of the 20th record was successfully updated
conn = sqlite3.connect("pollingplaces_2020.db")
cursor = conn.cursor()
query = "SELECT county_name FROM pollingplaces_2020 WHERE id = 20"
cursor.execute(query)
result = cursor.fetchone()

if result and result[0] == "DURHAM":
    print(
        "The county_name of the 20th record is 'DURHAM', therefore the query
    )
else:
    print("The 20th record's county name is not 'DURHAM'.")

cursor.close()
conn.close()
```

```
The county_name of the 20th record is 'DURHAM', therefore the query_update()
function worked successfully.
```

Finally, I create a simple test to determine if there `query_delete()` function works properly.

In [42]:
```
query_delete()
# Identifying if the 10th record was successfully deleted.
conn = sqlite3.connect("pollingplaces_2020.db")
cursor = conn.cursor()
query = "SELECT COUNT(*) FROM pollingplaces_2020 WHERE id = 10"
cursor.execute(query)
exists = cursor.fetchone()[0] > 0

if exists:
    print(
        "A row with id = 10 exists, therefore the query_delete() function di
    )
else:
    print("No row with id = 10 found.")

cursor.close()
conn.close()
```

```
No row with id = 10 found.
```