Mass Shooting in America

109550041 常乃璇

Motivations

May 14th, a mass shooting incident happened in Buffalo, New York, targeted at African American in a local supermarket. The news shattered my heart. A day later, another shooting happened at a Taiwanese church at Orange county, CA, which got a wide coverage across Taiwanese media.

Then in just under a week, right as I was trying to come up with a topic for my final homework, the most horrific tragedy I could've ever imagined took place in Uvalde, Texas. A mass shooting at an elementary school, 21 people, including 19 children, were killed. It really got the whole America thinking: What is wrong with our laws? What is wrong with our society?

A lot of people, including myself, got this "American dream" in our head. But all the shooting incidents along side with the abortion laws in some states, don't say gay bill, and transphobic laws, made me questioned: is America really going to be a safe place to live in, in the next twenty years?

Therefore, I want to make a simple application for people to quickly look up informations on mass shooting that took place in various states in the US, along with the number of gun law, and suicide information. Hope to raise awareness on how the lawmaker should take action on gun control and how mental health issue should be taken more seriously.

Application description

(I design the application similar to a website, because that's how I originally envisioned it to be.)

The main page listed out the total number of mass shooting cases, with total fatalities, and the total effected number (fatalities + injured).

Then user are able to,

(a) Look up cases by states: Type in a state, see all the cases, and choose to see the details of one case. User is also able to leave a message.

(b) Look up cases by year: Type in a year, see all the cases, and choose to see the details of one case. User is also able to leave a message.

(c) See the relation of the number of gun law and fatalities of mass shooting: Type in a state, and see each year's mass shooting related death and the number of gun laws.

(d) See the relation of the number of gun law and the suicide rate, death: Type in a state, and see each year's suicide rate, death, and the number of gun laws.

(**e) See the messages left by users**: See all the messages left by previous users, including the case they're messaging under, their username, the date they left the message, and the message itself.

User can reload page (a) to (e) or go back to the main page after executing a function.

Data collection

Sources:

I gathered the data of mass shooting in America, firearm law, and suicide rate from the below sources:

- Mother Jones: **US Mass Shootings, 1982–2022: Data From Mother Jones' Investigation**
  - https://www.motherjones.com/politics/2012/12/mass-shootings-mother-jones-full-data/
- State firm law: **State Firearms Laws Database, 1991-2020**
  - https://www.statefirearmlaws.org/resources
- CDC: **Suicide Mortality by State**
  - https://www.cdc.gov/nchs/pressroom/sosmap/suicide-mortality/suicide.htm
- World population: **List of State Abbreviations**
  - https://worldpopulationreview.com/states/state-abbreviations

Alteration:

I downloaded them in .csv format manually, and altered some of the data.

Alteration including:

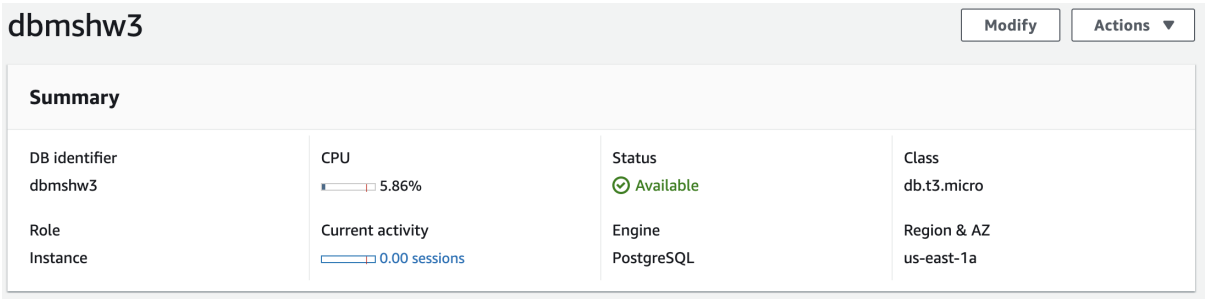
1. **Mass shooting:**
   a. Delete total victim (it's the summation of fatalities and injured)
   b. Unified all the "yes", "no", "white", "black", "male", "female" etc, into the same format (some of them are capitalized or shorten, some aren't). I was thinking that maybe I'll need to calculate a summation of some sort, therefore it'll be easier if all the data are the same format. I ended up not building that function though.
   c. There were a lot of way this data indicate "unknown", for example, *blank*, "Unknown", "-", "unclear", "unknown"…. In order to make it easier to process, I delete all these unknown information and have them be left *blank*.

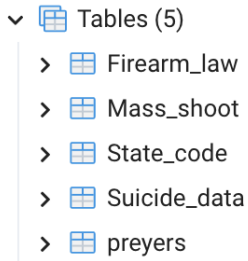
3. **Firearm law:**
    a. This dataset not only contains the total law count by year and state, but also has the number of different kind of firearm law. I don't need that many attributes and therefore I only saved the ones I need.
3. **Suicide data:**
    a. There's an URL attribute which I discarded.
    b. The state is listed as short code of length 2, therefore I need an extra table to look up the short code and the full state name.
4. **State Code:**
    a. There's an Abbreviation attribute which I discarded.

Building Database and importing:

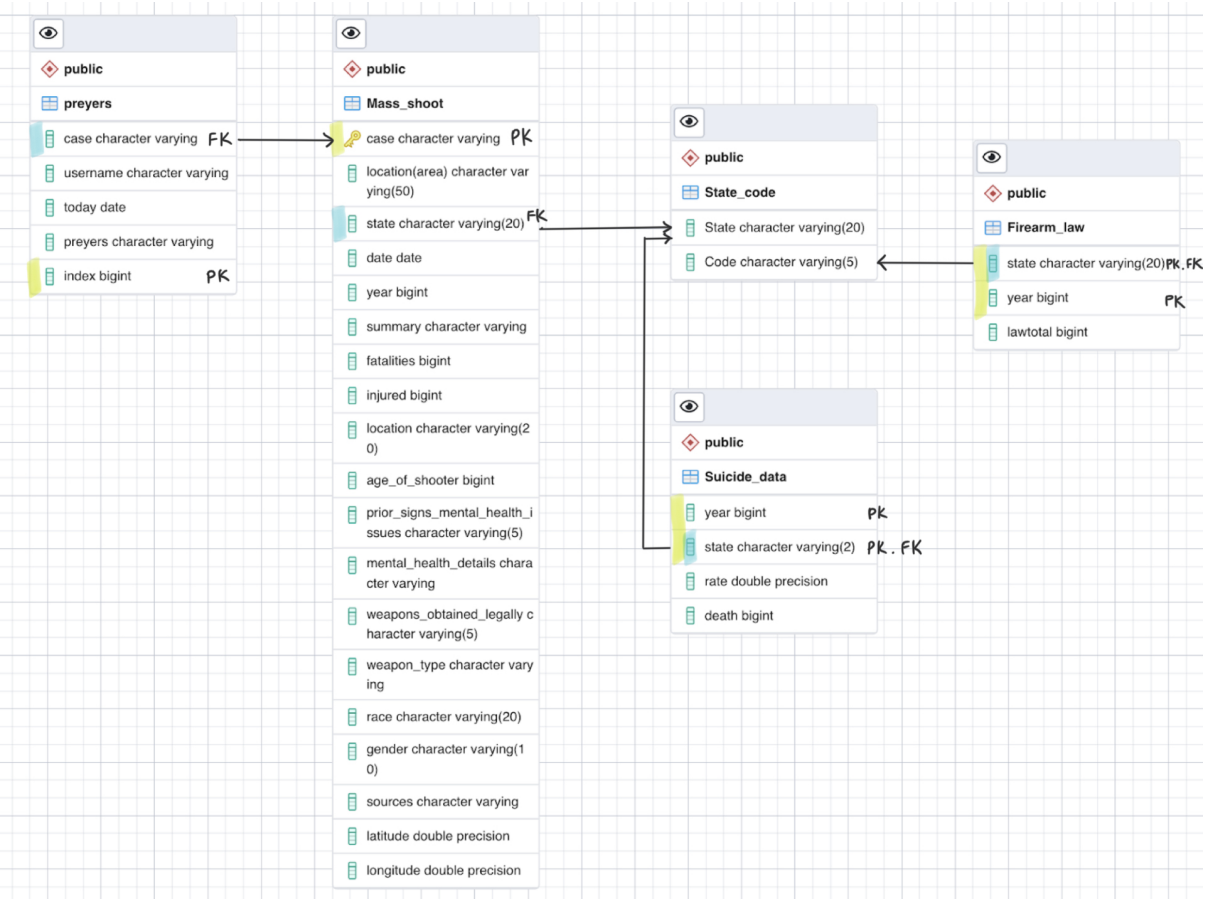
The database is set up the same way as HW1 and HW2 on AWS RDS service. The tables are also created and imported the same way by the UI of pgAdmin. As mentioned above, I altered the data a little before importing.

The "preyers" table is created by the UI of pgAdmin, the data in it are inserted using the application.





Database schema



Mass_shoot:

This table documented the mass shooting cases.

Constraint:

a. case name is the primary key

b. state is a foreign key

c. case, location(area), state, date, year, summary, fatalities, injured, and location are NOT NULL.

d. Tradeoff: Location (area) actually contained the “state” as well, but I still left the “state” attribute. Because the original data had the location information listed as it was, and separating county and state is going to be a lot of work. And the separated state attribute is key for referencing other tables.

e. Tradeoff: Date and year are in the exact same situation as location and state, since I’m using state and year information so much, it just is more intuitive to have them as their own attributes. But also the original data has state and year incorporated in location and data, it’s not an easy task to separate it out. Therefore I decided to keep both.

f. “total victim” attribute was removed from the original data, because it’s dependent on “fatalities” and “injured”.

It is in 1NF, because there are partial dependency with “state” and “location(area)”, “date” and “year”. Separating them requires a lot of alteration to the data collected from an existing source.

State_code:

This table keep track of the state name and the 2 letter code for a state. Which is necessary since our mass shooting cases data, firearm law data have full state names as their attributes but suicide rate has the 2 letter code as attribute.

Contraint:

a. state and code are NOT NULL

It is in BCNF.

Firearm_law:

This table documented the number of firearm related laws by states and year.

Constraint:

a. state and year is the primary key
b. state is a foreign key
c. state, year, and lawtotal are NOT NULL

It is in BCNF.

Suicide_data:

This table documented the suicide rate (death/100,000) and death by state and year.

Constraint:

a. year and state (which is the 2 letter code) is the primary key
b. state is a foreign key
c. year, state, rate, and death are NOT NULL

It is in BCNF.

Preyers (typo for Prayers):

This table contained the messages left by user of this application, with a username, the date the message was left (obtained by python datetime library), and an index.

Constraint:

a. index is the primary key

b. case is a foreign key

c. case, username, today, preyers, index are NOT NULL

It is in BCNF.

I added index later in the project (using ALTER TABLE for adding attribute, and UPDATE for adding value to existing rows), because I realized if the manager of the db want to delete one of the messages, without an index, if there are multiple messages left by the same username, on the same day, with the same context. It'll be impossible to access only one of them. The index information is only for the manager of the db but not the user.

Application's functions and the related SQL queries used for the function

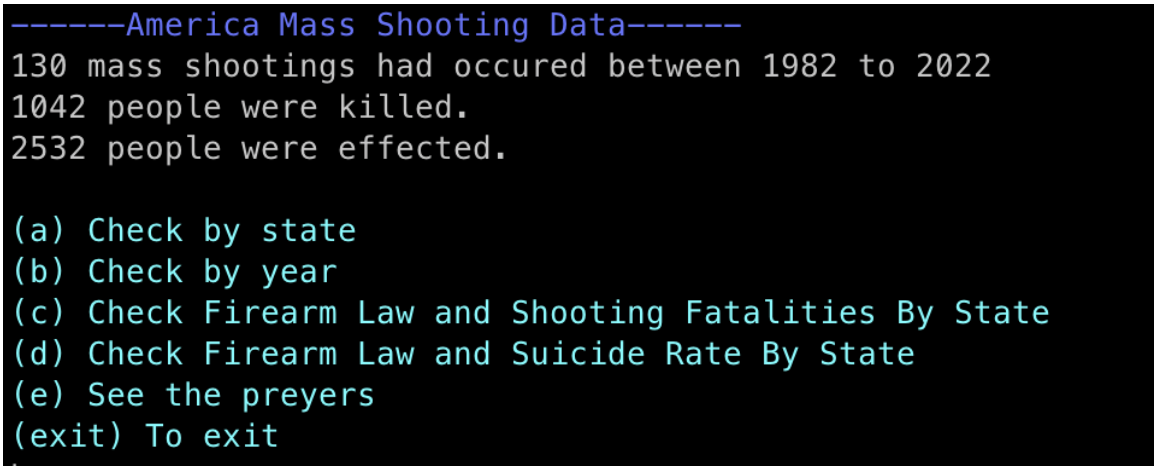
💡 How to execute:

```
pip3 install psycopg2
python3 app.py
```

First, install "psycopg2" module.

Second, run "app.py" file.

1. Main page:



a. total mass shooting number

```
SELECT COUNT(*)
FROM public."Mass_shooting"
```

b. total fatality

```
SELECT SUM(fatalities)
FROM public."Mass_shoot"
```

c. total effected (fatalities + injured)

```
SELECT SUM(fatalities) + SUM(injured)
FROM public."Mass_shoot"
```

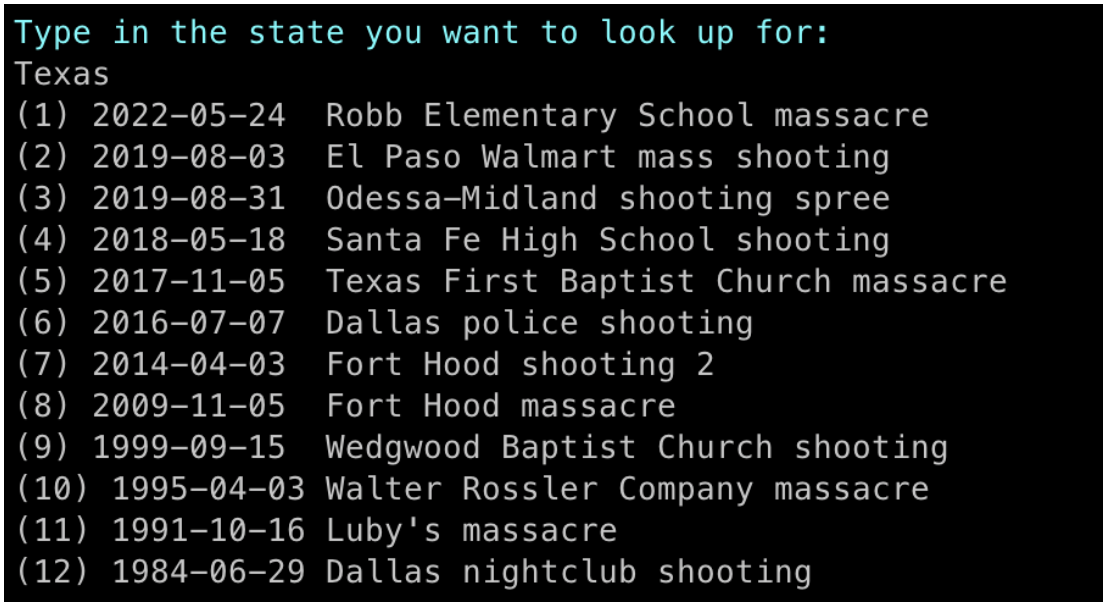
2. (a) Check by State:

a. Listed all the states in the “State_Code”

```
------America Mass Shooting Data by States------
These are all the states in America:
Alabama               Alaska               Arizona
Arkansas              California           Colorado
Connecticut           Delaware             District of Columbia
Florida               Georgia              Hawaii
Idaho                 Illinois             Indiana
Iowa                  Kansas               Kentucky
Louisiana             Maine                Maryland
Massachusetts         Michigan             Minnesota
Mississippi           Missouri             Montana
Nebraska              Nevada               New Hampshire
New Jersey            New Mexico           New York
North Carolina        North Dakota         Ohio
Oklahoma              Oregon               Pennsylvania
Rhode Island          South Carolina       South Dakota
Tennessee             Texas                Utah
Vermont               Virginia             Washington
West Virginia         Wisconsin            Wyoming
```

```
SELECT "State"
FROM "State_code"
```

b. List all the cases from the state user enters (ex: Texas)




```
SELECT "case", "date"
FROM "Mass_shoot"
WHERE "state" = %(sta)s
```

c. List all the information of the case the user enters (ex: 4)


```
SELECT *
FROM "Mass_shoot"
WHERE "case" = %(cas)s;
```

```
SELECT "lawtotal"
FROM "Firearm_law"
WHERE "state" = %(sta)s and "year" = %(yr)s
```

d. Allow user to leave a message, inserting it into “preyer” table

```
Do you want to leave a message?
(y) yes
(n) no
y
----------Send preyer----------
Type in a username: Sofie
Leave some message:
(Please leave positive messages only.)
Sending a lot of love
```

Get the number of total messages, in order to give an index to the new one.

```
SELECT COUNT(*) FROM preyers
```

```
INSERT INTO PREYERS ("case", "username", "today", "preyers", "index")
           VALUES (%(cas)s, %(nam)s, %(dat)s, %(pre)s, %(len)s)
```

3. (b) Check by Year:
    a. Listed all year range of the data

```
------America Mass Shooting Data by Year------
The data range from 1982 to 2022
```

```
SELECT min(year)
FROM "Mass_shoot"
```

```
SELECT max(year)
FROM "Mass_shoot"
```

    b. List all the cases of the year the user enters (ex: 2021)

```
Type in the year you want to look up for:
2021
(1) 2021-11-30  Oxford High School shooting
(2) 2021-05-26  San Jose VTA shooting
(3) 2021-04-15  FedEx warehouse shooting
(4) 2021-03-31  Orange office complex shooting
(5) 2021-03-22  Boulder supermarket shooting
(6) 2021-03-16  Atlanta massage parlor shootings
```

```
SELECT "case", "date"
FROM "Mass_shoot"
WHERE "year" = %(yr)s
```

c. List all the information of the case the user enters (The same as 2.)

d. Allow user to leave a message (The same as 2.)

4. (c) Check Firearm Law and Shooting Fatalities By State:

a. List all the states in the “State_Code” (The same as 2.)

b. List year, that year mass shooting fatalities, and firearm law numbers of the state user choose

```
Type in the state you want to look up for:
Florida
Year    Killed  Firearm law number
2019    8       30
2018    17      30
2017    10      21
2016    49      21
2013    7       21
1999    5       22
1996    6       21
1990    10      No record is found
1987    6       No record is found
1982    8       No record is found
```

```
WITH t1 AS(
  SELECT "state", year, sum("fatalities") as death
  FROM public."Mass_shoot"
```

```
  GROUP BY state, year
)

SELECT t1."state", t1.year, t1.death, F."lawtotal"
FROM t1
LEFT JOIN public."Firearm_law" as F
on t1."state" = F."state" and t1."year" = F."year"
WHERE t1."state" = %(sta)s
ORDER BY year DESC
```

I choose to use LEFT JOIN to keep all the mass shooting fatalities, because first, I want that to be the main focus of this function, second, the firearm law data are more incomplete.

5. (d) Check Firearm Law and Suicide Rate By State:
   a. List all the states in the “State_Code” (The same as 2.)
   b. List year, that year mass shooting fatalities, and firearm law numbers of the state user choose

```
Type in the state you want to look up for:
Oklahoma
Year    Suicide Rate    Suicide Death   Firearm law number
2020    21.9            869             8
2019    20.5            816             8
2018    20.0            790             9
2017    19.1            756             9
2016    21.0            822             9
2015    20.3            790             9
2014    19.1            736             9
2005    14.8            522             10
(Suicide rate is calculated by number if death/10000)
```

```
WITH t1 AS(
  SELECT "State", year, rate, death
  FROM public."Suicide_data" as A
  LEFT JOIN public."State_code"
  ON A."state" = "State_code"."Code"
)

SELECT "State", t1.year, "rate", "death", "lawtotal"
FROM t1
LEFT JOIN public."Firearm_law"
ON t1."State" = "Firearm_law"."state" and t1.year = "Firearm_law"."year"
WHERE t1."state" = %(sta)s
ORDER BY t1."year" DESC
```

I choose to use LEFT JOIN to keep all the suicide, because first, I want that to be the main focus of this function, second, the firearm law data are more incomplete.

6. (e) See the prayers:

```
--------Prayers to the victims---------
To victim of Robb Elementary School massacre
Username: Emerald
Date: 2022-06-21
"Children deserve to grow up"

To victim of University of Iowa shooting
Username: Eric
Date: 2022-06-21
"Mental health issue should not be overlooked..."

To victim of Virginia Tech massacre
Username: Roland
Date: 2022-06-23
"Gun should be forbidden by law"

To victim of Trolley Square shooting
Username: Ryder
Date: 2022-06-23
"praying for all the family"

To victim of Santa Fe High School shooting
Username: Sofie
Date: 2022-06-23
"Sending a lot of love"
```

```
SELECT *
FROM "preyers"
ORDER BY INDEX ASC
```

Some functions I considered developing include: listed out each year case count, location analysis (location as in: school, home, workplace, etc), gender analysis, prior sign of mental health analysis, etc. I choose not to do some of them due to the concern that it might be putting the spotlight on individual people who commit the shooting, but I'd like this project to focus on how law maker should stand up and make the change.

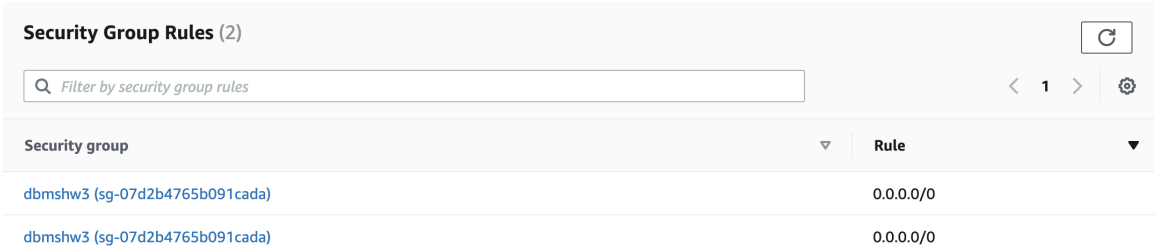
Problems and What I've Learned:

1. I ran into a timeout issue with the server.

   Found this post: https://serverfault.com/questions/656079/unable-to-connect-to-public-postgresql-rds-instance

   Saved my life! (I was literally crying when this issue happen.)

   The solution is to check security group, and allow everyone to access (not secure at all, but at least I get to work on my project now).



2. I learned how to connect postgres database with python using psycopg2. It was rather simple and easy. This website is very useful: https://www.postgresqltutorial.com/postgresql-python/connect/

```
import psycopg2

conn = psycopg2.connect(host = "*Endpoint*" ,
                        database = "*database name*",user = "postgres",
                        password = "******",port = "5432")
...
conn.close()
```

   Use cursor to execute the sql. And fetch the result with fetchone(), fetchmany(), or fetchall() (depending on your query and how you want to use the return value).

```
cursor = conn.cursor()
sql ="""SELECT sth FROM table;"""
cursor.execute(sql)
rows = cursor.fetchall()

for row in rows:
  print(row[0])
  ...
```

   I've also learned how to connect using PHP, because I originally wanted to build a website. I was unsuccessful with the web server and stuff. (I was trying the php local server, but I wasn't able to connect to the database successfully. I think if I were to use xampp or apache, I'll be able to succeed, but I was too unfamiliar with either of them and way too close to deadline.) I spent around 2.5 days learning all about php, javascript, and html, though I failed, still feel like I've learned a lot. I'll for sure make it work next time!

3. For aesthetic sake, I learned to use "format", "datetime", and changing text color in python. Which are all super easy, really useful, but you just don't normally feel the need to learn it.