Project Description:

This project tries to cover the knowledge of Python, SQL, regular expressions, and the Linux command line tools to extract information from web. We will get pdf file from Norman, Oklahoma police department and process the pdf file to create a database about it name normanpb.db from sqlite.

The main steps are download data, extract data and create database and show information from it. The functions will be written in a python file named project0.py. We will write test files about the functions from project0, and we will write a main.py to use all the functions.

Download data. We will write a function named fetchincidents, parameter is url to download data from the website. Incident data is returned from the function.

Extract data. We will extract data from incident data from a function named extractincidents and return incidents from it.

Create database. We will first create database named normanpd.db from function createdb and return the database object, then insert data from processed incidents and use a function named populated, the parameter will be db and incidents, then the data will be inserted to database, and then we can use status function to print incident counts, the parameter is db, the print rule is as below: the list should be sorted first by the total number of incidents and secondarily, alphabetically by the nature. Each field of the row should be separated by the pipe character (|).

How to run

pipenv run python project0/main.py --incidents <url>

Functions

project0.py \

fetchincidents()- this function has a url parameter to indicate where to download data, this function will use urllib to download data from the url and return the data we downloaded from the url.

hasLower()- this function has a line parameter to judge if there exists lower case from the line, if there exists a lower case from line it will return true, otherwise it will return false.

get\_incident()- this function has a incident\_text parameter, it is a str we will parse to get one incident data, if the incident data has correct format as incident\_time, incident\_number, incident\_location, nature and incident\_ori it will return a tuple of it, otherwise it will return none.

extractincidents()-this function has a incident\_data parameter, ti use pdfReader to read incident\_data we get from fetchincidents() function (pdfFileReader is discarded for new pypdf), and we use get\_incident() to get valid information from the parsed data, many data may be discarded as pdfReader may not parse the data correctly, like it maybe parse one line data to two lines, and when we parse the data it is in incorrect format.

createdb()- this function has no parameter and create a sqlite database named normanpd.db, then return the database object.

populatedb()-this function has two parameters as db and incidents, it will insert incident data to database from database object and incidents data.

status()- this function has a parameter as db, it will print the database data in the rule we are required.

test\_download.py \

test\_download()- this function has a url parameter to indicate where to download data, this function will test fetchincidents function from project0.py.

test\_extract.py \

test\_extract\_incidents()- this function has a url parameter to indicate where to download data, this function will test fetchincidents and extractincidents functions from project0.py.

test\_status.py \

test\_status ()- this function has a url parameter to indicate where to download data, this function will test fetchincidents, extractincidents, createdb, populatedb, status functions from project0.py.

Bugs and Assumptions

Bugs:

When the network is slow, the download data part maybe block and you will wait long time to complete the download process.

Assumptions:

As for download data, we directly process it in the fetchincidents function and don’t care about temporary files. For the data processed by pdfReader, we just discard the incorrect format data if it is wrongly processed by pdfReader, like it may break one line to two lines, for the space problem, pdfReader may wrongly discard space, and we ignore this problem and see it as one line, e.g. if the space between incident\_time and incident\_number are wrongly ignored by pdfReader, we will consider the line as incorrect format and discard it. Our pypdf version is higher than 3.0.0.