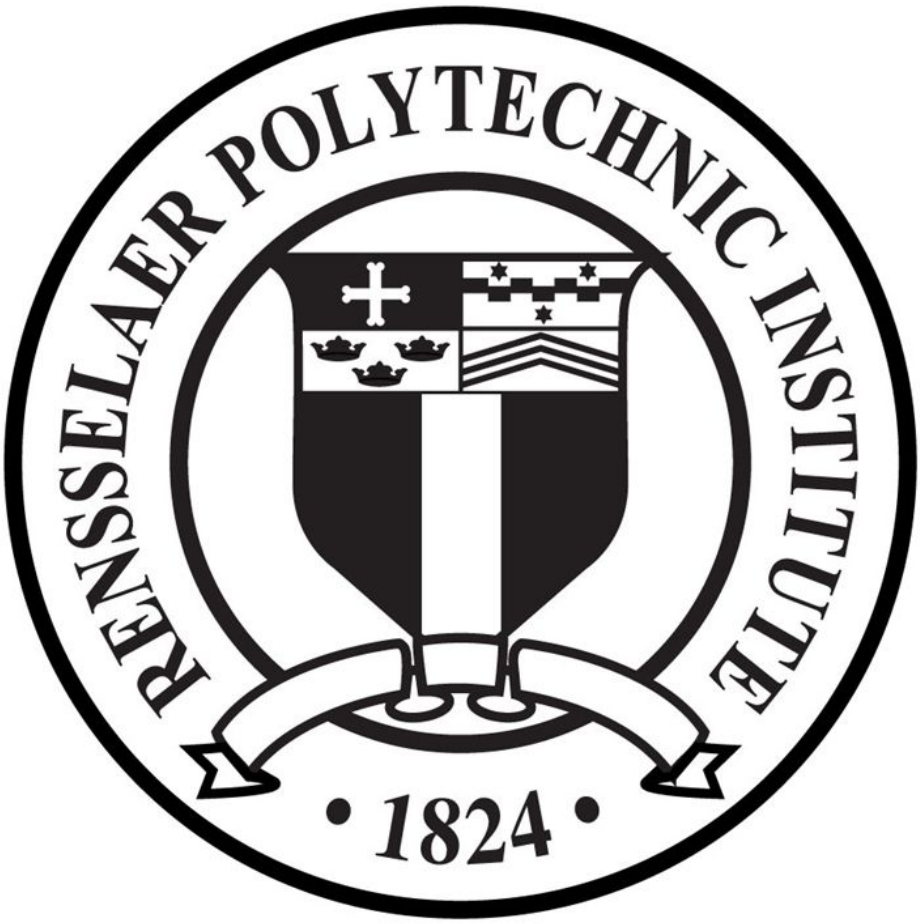
Project Schema

ITWS 6250 — Database Applications and Systems



Isaac Llewellyn & Shepard Gordon

Fall 2020

Summary

For this project, you will find multiple publicly available datasets that share common attributes (e.g., Zipcode), create a normalized schema describing the structure of the data, and produce an application that can populate your schema with the data—including the ability to refresh the data—and run queries on the data, producing useful output.

Objective

There are several objectives for this assignment.
• Gain an awareness of the scope of datasets publicly available for research purposes
• Demonstrate an ability to understand the structure of a dataset, as well as an ability to apply that
understanding, using concepts learned during class, to create an effective database schema
• Apply concepts learned during class to query the data, and extend those concepts to create an application allowing users to do the same

Description

There are a number of different sources of publicly available data. Both the State of New York and the Federal Government provide hundreds of datasets. There are numerous other sources of open data as well, but those two will get you started. Please pay attention to licenses for any datasets you use. Data itself is generally not eligible for copyright protection (at least in the United States), but schemas are, and there may be terms of service for accessing the data itself.

Select two datasets that are robust enough to be interesting (a dataset with only four columns and a few thousand rows probably doesn't qualify). Students taking the course for graduate credit need to select two additional datasets, for a total of four.

They should share a common attribute (or set of attributes). Create a SQL schema for your data, making sure that it's appropriately normalized. Students taking the course for graduate credit need to use a non-relational database and schema for some portion of their data. Create an application in Python 3 that will load the dataset into a Postgres database defined by your schema. The loading process should be able to be re-run with updated datasets to refresh the data in the database.

Take some time to explore the data by running some SQL queries. Once you have an idea of some of the more interesting aspects of the data, create an interface for your application that will allow the user to explore the data as well.

Your application shouldn't re-implement the wheel. You don't need to provide the user with a way to do whatever they want. It should provide more of a self-guided tour, rather than a detailed map. It should provide interactivity beyond simply allowing the user to run one of five or six static queries, but it doesn't have to allow them to write their own queries.

For example, there might be a dataset giving the results of health inspections of restaurants in New York.

Your application might allow the user to see which restaurants in their area had violations, or how often a given restaurant received a violation, or whether restaurants in a certain area get more violations than other areas.
The interface can be text-based. If you want to go further and provide visualizations, that's fantastic, but it isn't within the scope of the project (you will not be graded on the appearance of your interface). Your application should be able to be built easily, the data loaded easily, and used easily.

You will demonstrate your application for the class in a short presentation in which you will discuss your choice of datasets, outline the design of your schema, and demonstrate the types of queries your application can perform.

All work will be done either individually or in teams of two.

Deliverables

There are four main deliverables:

- *Project Memo*
- *Database Schema*
- *Project Code*
- *Presentation*

Due Dates

- The memo is due on Submitty by 11:59pm on Friday October 9
- The schema is due on Submitty by 11:59pm on Wednesday October 26.
- The data-loading code is not formally due, but students should aim to have it completed by Wednesday November 25.
- The completed application is due on Submitty by 11:59 on Sunday December 6.
- You should be prepared to present your project to the class during the lecture period on Wednesday

Database Schema
You will submit a single SQL file that can be run to create the schema for your database. The SQL file will also be due before the rest of the project and will be graded at that time so that feedback can be incorporated into the final product.
Students taking the course for graduate credit will need to bear in mind that some portion of their data will be stored in a non-relational database.
That aspect of the project is not due with this deliverable.

Database Schema

• Team Members:

Isaac Llewellyn -- ITWS Coterm Student
Shepard Gordon -- ITWS Graduate Student

• Datasets

[Current Season Spring Trout Stocking | State of New York]

Year INT	DEC Region INT	County TEXT	Town TEXT	Waterbody TEXT	Date TEXT	Number INT	Species Name TEXT	Size (inches) TEXT

[National Register of Historic Places | State of New York]

Resource Name TEXT	County TEXT	National Register Date DATE	National Register Number TEXT	Longitude (number_????Numeric_perhaps?????)	Latitude (number_???Numeric_perhaps?????)	Location (location)

[Recommended Fishing Rivers And Streams | State of New York]

Waterbody Name TEXT	Fish Species Present at Waterbody TEXT	Comments TEXT	Special Regulations on Waterbody TEXT	County TEXT	Types of Public Access TEXT	Public Fishing Access Owner TEXT	Waterbody Information TEXT	Longitude (number_????Numeric_perhaps?????)	Latitude (number_????Numeric_perhaps?????)	Location (location)

[Fish Stocking Lists (Actual): Beginning 2011 | State of New York]

Year INT	County TEXT	Waterbody TEXT	Town TEXT	Month (month)	Number (number_????Numeric_perhaps?????)	Species TEXT	Size (Inches) (number_????Numeric_perhaps?????)

– How you plan to join the datasets

We plan on joining the datasets by their county fields, allowing people to explore the relationship between all New York fish stocking vs the subset of New York trout stocking, relating them in comparison to recommended historic places, fishing rivers and streams close to their location.

```
CREATE TABLE County_information (

County_name         TEXT ,
Town_name         TEXT,
PRIMARY KEY (County_name, Town_name)

);

CREATE TABLE Stocking_information (
StockingIDD serial primary key,
Year         INT,
Waterbody         TEXT,
Month         varchar(40),
Number         INT,
Species         TEXT,
Size_Inches         TEXT,
Future         boolean,
County_name         TEXT,
Town_name         TEXT,
FOREIGN KEY (County_name, Town_name)
REFERENCES County_information (County_name, Town_name)
);

CREATE TABLE Waterbody_information (

Waterbody_Name             TEXT,
Fish_Species_Present       TEXT,
Comments                   TEXT,
Special_Regulations        TEXT,
Types_of_Public_Access     TEXT,
Public_Fishing_Access_Owner TEXT,
latitude                   float,
longitude                  float,
Location                   POINT,
Waterbody_information  text,
County_name         TEXT,
PRIMARY KEY(Waterbody_Name, latitude, longitude)

);

CREATE TABLE County_historic (

Resource_Name            TEXT,
National_Register_Date     DATE,
National_Register_Number   TEXT  PRIMARY KEY,
Location           point,
County_name         TEXT

);
```

``` lang=psql

TA / Instructor Grading Total
2 / 5
Normalization (Graded by: Johnson)
-3  Is there a reason you separated out fish_stocking and trout_stocking? It seems those could be one table with a single boolean (or other type) attribute to indicate whether it was referring to trout or something else. Check,.

I also suspect that you want to create a fish_species table, along with a "join table" indicating which fish species are present in which counties or bodies of water, rather than using your fish_species_present attribute (reach out to me for an explanation if that doesn't make sense). Perhaps -- Thoughts?

You also probably want to separate out your waterbody data into a table that's separate somehow from town and county information (unless **multiple counties stock fish in the same body of water**?).
4 / 5
Data Types (Graded by: Johnson)
-1  Consider whether using varchar for month types is ideal (what if you wanted to do range queries?)

How well do your location data line up with each other? Since you're using a point to refer to things like a body of water. How close would two points have to be in order for them to be assumed to be equivalent?

They line up with google map pins / similar pins relative to the site. They do not overlap and point to locations that people would park or stop to go fishing. Many points may exist along a river or waterbody

Eg
https://www.google.com/maps/place/41%C2%B055'24.9%22N+74%C2%B051'30.2%22W/@41.9235226,-74.8583517,3a,75y,26.72h,77.21t/data=!3m7!1e1!3m5!1sDsSHnVO4hQNPC9TNzo5N5w!2e0!6s%2F%2Fgeo3.ggpht.com%2Fcbk%3Fpanoid%3DDsSHnVO4hQNPC9TNzo5N5w%26output%3Dthumbnail%26cb_client%3Dmaps_sv.tactile.gps%26thumb%3D2%26w%3D203%26h%3D100%26yaw%3D330.73947%26pitch%3D0%26thumbfov%3D100!7i3328!8i1664!4m5!3m4!1s0x0:0x0!8m2!3d41.9235772!4d-74.858399
5 / 5
Accuracy (Graded by: Johnson)
Does your schema capture the original data

0  I deducted the point in the Effectiveness category, but I think you're going to have a problem getting your County_information to accurately reflect anything, because you'll end up with multiple entries for each county (one for each month and waterbody).
7 / 10
Effectiveness (Graded by: Johnson)
Is your schema likely to be effective

-3  **I think your County_information table foreign keys are backwards. You want your county (and probably a separate**
```

waterbody) table to be the "source of truth" for county information, and then have the stocking and rec tables refer to it, rather than the other way around. Otherwise, how does it handle fish stocking for more than one month in the same county? The same goes for bodies of water. What do you have in place to make sure your various tables refer to the same bodies of water?

I'd like to see better use of constraints (particularly unique constraints--since you're using an artificial primary key, you lose some of the protection that a natural primary key provides, but you can replace it with a unique constraint).
Overall note from Samuel:

I think this is a good start, but there's plenty of room for improvement. I'd focus first on fixing your foreign keys, and then on extracting out waterbody and fish species information into their own tables.

Then make sure you have a viable approach to linking the historic sites with the bodies of water. (Can you link them by county? Or are you ready to do range queries with the location point data?)

Forigen keys changed.
Waterbody and stocking information consolidated
We can link data via town information