Project 4

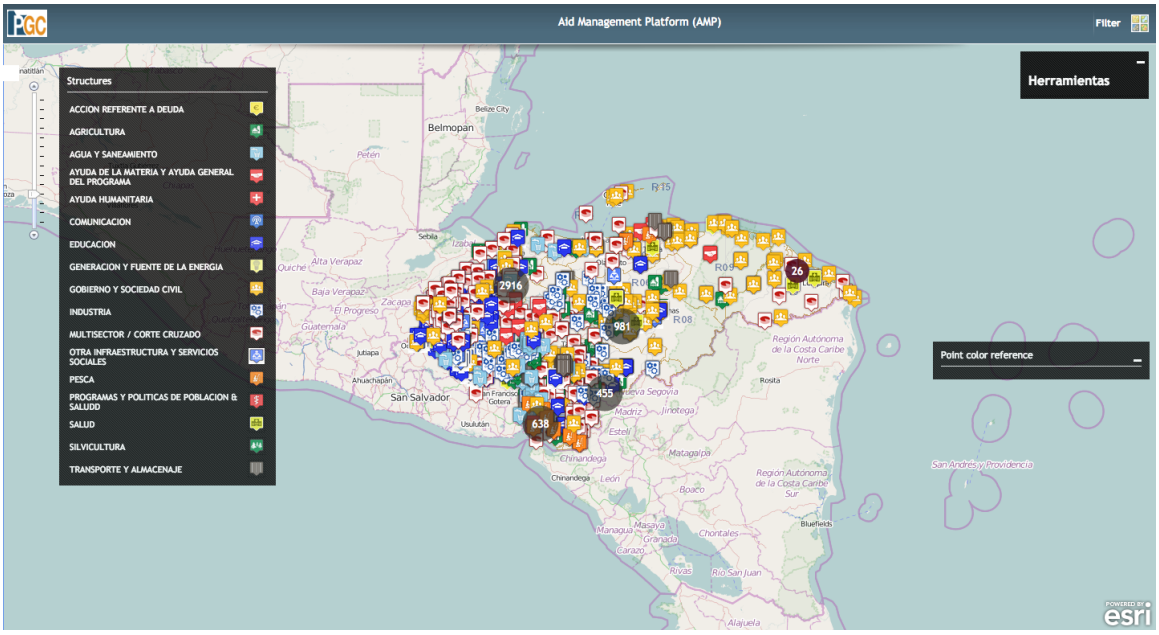
Benjamin Arancibia

Part 1: Use Case and Obtain the Data

The data used in this project is a brand new dataset that is only about a week old and was first published on Friday November 7th, 2014. It is a dataset of all active foreign aid project locations in Honduras. Some quick stats about the dataset: 420 foreign aid projects geocoded that led to more than 5,000 project locations in Honduras. It is very rare for an active foreign aid project to only have 1 location. The dataset that I am using for this project is simpler version of the full dataset. The columns in the dataset for this project are: ID, Title, Latitude, Longitude, Type (sector), and Description. Usually this type of dataset has 20+ columns more, but this is raw dataset that still needs to be cleaned and merged with the data currently present in the Honduras database.

This data is important because the Government of Honduras, Donors in Honduras, and Civil Society groups have no idea where project locations are in Honduras. Creating this geocoded data allows users to merge this project information with other geospatial information to perform better analysis. Ultimately, this better information will allow for better decisions and an improvement in planning by the Government of Honduras.

The data can be found at this website: http://pgc.sre.gob.hn/ free for download and open to everyone.



Part 2: Bring the data into R

I imported the csv into R using the typical read .csv command. The only issue I had is in the .csv the IDs are viewed in scientific notation, this needs to be altered when imported into R. The

reason why this needs to be changed is because many IDs start with 8717 and the projects can be merged accidently. This results in around 3000 locations for one project. The structure of the dataframe is as follows.

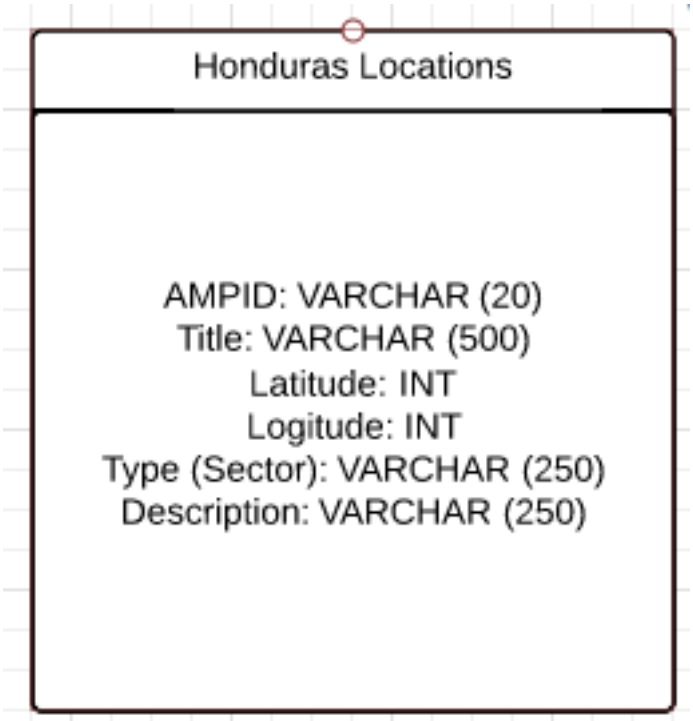
AMPID = Project ID in Government of Honduras's system
Title = Title of Project in Government of Honduras's system
Latitude = latitude of project
Longitude = longitude of project
Type = Sector of Project in Government of Honduras's system.
Description = Description of project if any

For type (sector), there can be multiple sectors. Right now they are identified as multisector and as part of the cleanup of the dataset data transformations will remove multisector and fill in the appropriate sectors. Also, part of the cleanup of the data will be resolving encoding issues.

Part 3: Bring the Data into PostgresSQL

I created a table in PostgreSQL for the csv file. This data could have been easily separated into different tables. An example of different tables could be a table with ID, TITLE, Type, and Description and another table with ID, Latitude, and Longitude. For this dataset, it does not make sense to separate the two, but when combined with the other data after the data clean up it will be split into multiple tables with location fields in one table, identification fields in one table, and amounts, dates, and components in another table.

The current table:



I created the table using the terminal and then imported the data. Command line code is located in the appendix.

Part 4: Bring the Data into MongoDB

The csv was loaded into MongoDB using the mongoimport utility, which can be found at this website: http://docs.mongodb.org/manual/reference/program/mongoimport/.

The data was imported into one collection, locations. The collection in MongoDB is structured similar to the postgres.

Part 5: Compare Methods

RData
Advantages – Rdata presents some interesting advantages over Postgres and MongoDB. R can consume csvs and APIs (this project used a csv, but it will be available via API soon), which allows for easy data importing. The structure of this data fits nicely into R since it is only one table. Also, R allows the users to do immediate stats and visualization within the program such as ggplot2. These types of libraries are not available for Postgres and MongoDB.

Disadvantages – There are limitations to Rdata though which include database structure and having a lot of people work on the same database. Having a couple csvs are great for R, but this dataset will need to be merged eventually into the full database. This full database is already in Postgres and has around 150 tables with 200 cached views for quick data queries on the UI side of a data platform. Also, having multiple people collaborate on the same set of persistent data in RData is a nightmare. R doesn't really provide an advantage for this dataset.

PostgreSQL
Advantages - The dataset works really well in Postgres. It is simple do quick sanity checks on the data to see like unique project IDs, etc. It would be easy to start merging this data in the current Postgres database of all projects. The tool also allows for easier collaboration on a database as compared to Rdata. It is also easy to create visualizations of the location data by just referencing the production database or a local database.
Disadvantages – The main disadvantage of Postgres is that it can be slow. This dataset only has one table so speed is not an issue; however, when it is merged into the larger database with around 150 tables speed will an issue. The other issue is that the servers that house this data are located in areas with low bandwidth. That's the reason for some many caches views to load the data quicker.

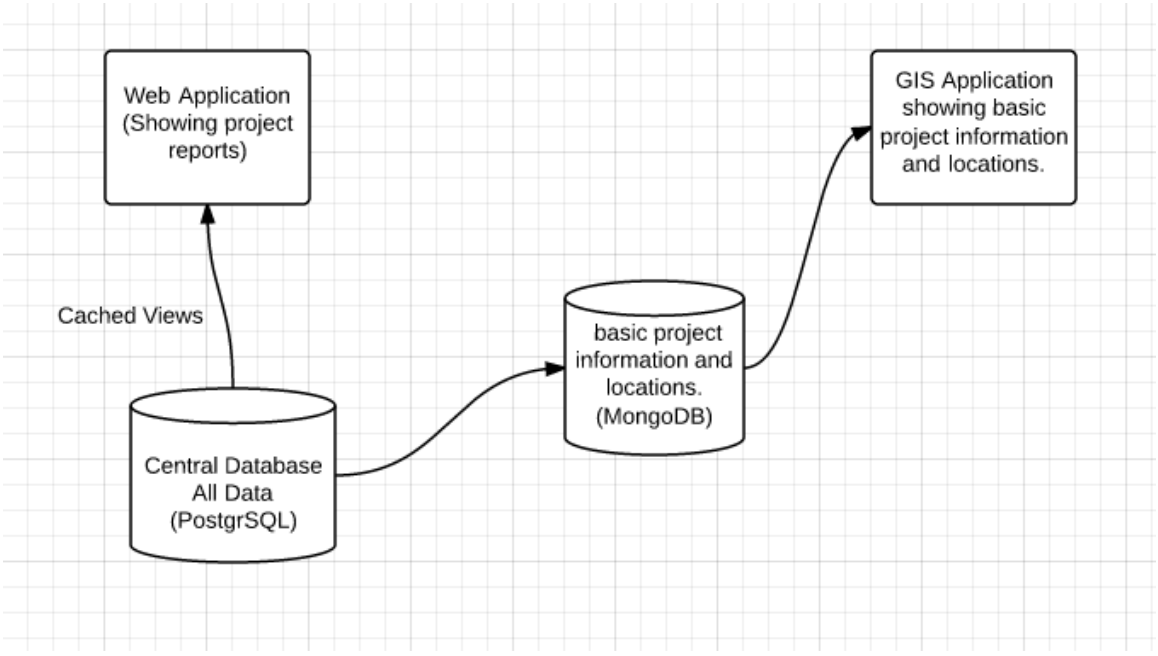
MongoDB
Advantages – All of the collaboration that applies to Postgres applies to MongoDB. MongoDB is really flexible and really fast.
Disadvantages – MongoDB is a great platform, but the data is so complicated that saving it in a NoSQL format does not give the best advantage for queries. There are a lot of fields that have multiple subfields. Also, the total dataset for the number of projects is only around 1000 rows, but the project locations are around 5000+ rows. These numbers could increase, but are not that significant for the scaling abilities MongoDB provides.

Conclusion

All three data platforms could be used on this simple dataset; however, when it is integrated into the larger dataset of all foreign aid projects in Honduras the only viable options are PostgreSQL and Mongodb.

Since the larger dataset is already in Postgres it is recommended to use Postgres for this smaller dataset and create cached views of data to make web applications load quicker in low bandwidth countries.

If there is time to do a complete overhaul of the architecture of the system, I would create a module system with one central database and different integration layers. These layers would be a hybrid of MongoDB and Postgres. I have used this type of architecture before and it is great for large amounts of location information when visualizing the data on a map. See below for quick example.



Code Appendix

Import Data into R

```
setwd("/users/bcarancibia/CUNY_IS_607/Project_4")

geo_hn <- read.table(file = "import_draft_v3.csv", header = TRUE, sep = ",")
options("scipen"=100, "digits"=10)

save(geo_hn, file="geo_hn.RData")
```

Import Data into Postgres

```
bcarancibia=# CREATE DATABASE Honduras_geo;
CREATE DATABASE
bcarancibia=# \connect honduras_geo;
You are now connected to database "honduras_geo" as user "bcarancibia".
honduras_geo=#

CREATE TABLE "Locations"
(
"AMPID" "char" NOT NULL,
"TITLE" "char" NOT NULL,
"LATITUD" float NOT NULL,
"LONGITUD" float NOT NULL,
"TYPE" "char" NOT NULL,
"DESCRIPTION" "char"
)

WITH (   OIDS=FALSE
);
```

Import Data into MongoDB

```
> use honduras_geo
switched to db honduras_geo
> db
honduras_geo
>

mongoimport –db Honduras_geo –collection locations –type csv –file
import_draft_v3.csv –headerline
```