# AWS

All machines are created in a Virtual Private Cloud.

There are 4 subnets 2 private and 2 public and span 2 availability zones.

10.0.0.\* is public on zone B

10.0.1.\* is private on zone B

10.0.2.\* is private on zone A

10.0.3.\* is public on zone A

web machines should be evenly distributed across the 2 availability zones so that if one goes down, the other can handle traffic.

All machines should be created in the private subnets for security. Nothing outside of the VPC can access machines in the private subnets directly.

When you create an Elastic Load Balancer in the VPC and then add web machines to it, the ELB can be accessed externally and it passes traffic through to the web machines behind it.

Ideally, databases will have a primary on one availability zone and then a failover on a another. Right now, there are no failovers setup. This needs to happen.

## GISPlanning.net

There are 2 domain controllers, dc1.gisplanning.net and dc2.gisplanning.net that handle the domain. All machines are named their domain names in AWS console. (this is a manual process any new machines should be labeled in the console whatever name you give them in the domain)

## AWS KeyPair

When creating new machines, you are required to use a keypair that AWS uses to encrypt the administrator password. You will need to decrypt the password to login to the machine. Here is the key I’ve been using for all boxes.

*[The RSA private key that was here has been copied into our password store. The public key was not provided. -Travis]*

Just copy and paste it into a file called GISP.pem and save.

## Passwords

gispmaps4u!

is the standard password I’ve been using for everything.

## Databases

gisp-prod-db01 and gisp-stg-db01 have scripts that run each night to backup/compress/and upload to S3 each database.

If you go to s3 and go to the bucket gispbackups, you’ll see a folder with each server name with subfolders of database name that have subfolders of the date and those contain backup pieces.

The gispbackups bucket has an auto expirey setting so that anything older than 2 weeks gets deleted.

## Scripts

I’ve placed all sql scripts that I had or used into a folder on gisp-stg-db01 under c:/jonathansscripts

## Dataflow

All of the source files I’ve used for data are in the S3 bucket DataSources. I use SSIS (projects live in bitbucket) to pull in the source data files into a raw database. From there, using another SSIS project for each application, I pull data from the raw databases and merge it into SizeUp and ZPDC databases.

A lot of the queries for sizeup dataflows need to have the Year/Quarter values adjusted in the query itself. I never got around to parameterizing them so for now they are manual.

There are some regular SQL scripts that are used in place of SSIS packages. These are stored on gisp-stg-db01.gisplanning.net/C:/jonathansscripts

The basic overview for updating data in sizeup:

1) take a backup of SizeUpData.

2) Restore it to gisp-calc-db01 as NewData

3) Download the data source files and place them in a folder.

4) Using the RawData project, find the package for data you want, and point the SSIS inputs to the new files you just downloaded.

5) run the package.

6) open SizeUp.Import SSIS package solution.

7) Run the UpdateIndustry package.

8) Run the BusinessImport package.

9) wait

10) wait some more (this process can take 3-4 days)

11) Shrink the logs on NewData and backup the database.

12) Restore NewData to gisp-stg-db01 as SizeUpData

13) Test thoroughly

14) Restore to gisp-prod-db01 as SizeUpData

# SizeUp

SizeUp achitecture is using MVC.NET 4. Most of the UI is rendered using ajax calls and very little server side processing is done. SizeUp consumes it’s own API to populate the data. API documentation is located [www.sizeup.com/developers/documentation](http://www.sizeup.com/developers/documentation)

The API requires an api token that is generated upon the request of the API loader script. Each subsequent call requires this token to be passed for the call to be valid. The token consists of the APIKeyId and number of ticks since 1970 and then encrypted in a string. We can use this to validate the age of the token and which API key the token was created for. If a token is older than 10 min we will not allow the request to succeed. The API Loader handles requesting a fresh token every 5 min. The user/developer of the API should never have to worry about API tokens.

## SizeUp Databases

The databases that sizeup needs to run are

SizeUpData

SizeUpMembership

SizeUpAnalytics

SizeUpUserData

SizeUpAPI

## SizeUpData

Handles all business/demographic/calculation data and is a read heavy database.

Tables:

**Business** has all of the businesses and related info like address, industry, phone, etc.

**BusinessCity** is the mapping table between businesses and cities. Since some cities geographic areas overlap each other it is possible that a business can be in multiple cities.

**BusinessData** is a standard fact table with Year, Quarter, Industry, GeographicLocation dimensions. A GeographicLocation can be a city, state,county,zip etc. This table is mainly used to do percentile and percentage calculations.

**ConsumerExpenditures and Demographics** is a standard fact table that has Year, GeographicLocationId as dimensions and all the variable values are in a single row for a given year, geographic location.

**GeographicLocation** is a parent table for ZipCode, City,County,Metro,State,Nation,Region,Division and ensures that the PrimaryKey is unique across all geographic locations.

**GeographicLocationGeographicLocation** contains precalculated intersections for each geographiclocation. The table was calculated by running spatial intersections on the polygons for each geo location. If there is an intersection, there is a record in this table. Using this you can find out which zipcodes are in a city/county/state etc or which cities are in which counties.

It is important to note the hierarchy of geographic locations.

ZipCodes do not overlap other zipcodes and can span city/county/metro/state boundaries.

Cities can overlap each other, and can span county/metro boundaries but cannot span state boundaries.

Counties do not overlap each other and will not extend outside of a state.

Metros are a collection of counties but can span state lines.

States: I hope you know what a state is.

**GeographicLocationRank** contains the ranks each geographicLocation is compared to other geographic locations of the same type. For example, each state is compared to each other state for averageRevenue in Pharmacies, so you will find a rank 1 through 51 (dont forget DC) for each state where the state that ranks 1 under pharmacies has the highest averageRevenue.

**Geography** contains the polygons and centroid and bounding box calculations for each polygon. There are Display Polygons (hi res) and calculation polygons (low res). Display polygons are used for drawing tiles on heatmaps etc.

**IndustryCity** Mapping between industry and cities. If a city is in here, then there is at least one business in that city in that industry.

**IndustryData** fact table that contains the aggregations of revenue, and employees for each Industry/GeographicLocation combo.

**Place** is a custom geographicLocation that is specific to SizeUp. A place is defined as a city/county combo. The reasons for this are:

Since a city can span multiple counties, we need a way to determine which county to associate to a user selection. We force the user to select the county when they are picking a city/industry on the home page.

**ResourceString** contains all the html and strings for the dashboard mainly. It allows textual updates to happen without a code push…. this could be improved upon.

**ZipCodeCounty and ZipCodePlace** should be no longer used and replaced by GeographicLocationGeographicLocation but this would need to be verified.

## SizeUpMembership

The user account data is stored here. This is a standard aspnet Membership database.

## SizeUpAnalytics

Strictly a write intensive database. All analytics, errors, events are stored here. There is a pageview table and an APIRequest table.

They log things such as SessionId and InstanceId. The definition of a sessionId is a sudoUnique string that gets stored as a cookie in the client browser. Each APIRequest gets a sessionId passed and is used to link multiple calls to the same session.

An InstanceId is defined as a single load of the API. A sudoUnique string is created for each instance that can be used to identify all of the requests that happened for a particular page load.

There are multiple instances for a given session.

There are multiple api requests for a given instance.

## SizeUpAPI

Stores API Keys and the roles for features that each key is able to access.

APIKey has a list of API Keys.

Role are the various roles available.

APIKeyRoleMapping maps the roles for each APIKey

APIKeyDomain is not used. It was going to be used to block access to the API from invalid domains. This is too problematic and difficult to maintain. We run the risk of blocking developers and client staging environments.

The recommended approach to handle abuse/unauthorized use of the API is to allow it but detect it. If you start to see a bunch of requests coming from [www.intuit.com](http://www.intuit.com) using the SBA APIKey, then you know you have a problem and you can reissue SBA a key and take steps litigous steps in the direction of intuit.

## SizeUpUserData

Stores extra data that the user enters. For example, when a user enters their revenue for Pharmacies in San Francisco, when they come back, we populate their revenue back in and run the report without them having to reenter the data. All of that is stored in this database.

# ZPDC

The code is in BitBucket and Database is ZPDC on gisp-stg-db01. Everything is pretty straightforward. Call comes in, data is requested from the database and rendered out to HTML.

All data is requested using EntityFramework. The database structure is a main fact table for demographics with Variable, GeographicLocation, Year as dimensions.

I’ve only focused on the community profile page and community search. Advanced search is unfinished.