

PostGIS on Rails

Matt Nemenman
matt@apartmentlist.com
@quarterdome

About me

- Programmer
- Love maps
- Building map based rental search engine @ Apartment List

Location Aware Apps

- 📍 Where am I?
- 📍 What is around me?

Where am I?

- Latitude and Longitude
 - HTML5 geolocation or GPS
- Address
 - Geocoding
 - Reverse geocoding

Where am I?

- What if you need to know ...
 - Neighborhood
 - School district
 - National park
 - Earthquake safety zone

What is around me?

- Yelp and Google Places API
 - Restaurants, bars, etc.
 - Points of interest

What is around me?

- What if you want to know ...
- What are three neighborhoods closest to me?
- Average rent for 1 bedroom in Lower Pacific Heights
- Crime rate on my city block
- Who is around me?

When 3rd party APIs fall short . . .

- Get your own data set
- Build your own solution

Spatial Systems

- MongoDB
- Solr
- MySQL
- Oracle / DB2
- PostGIS

PostGIS

- Geospatial extension to Postgres
- New datatypes
- Functions to work with those datatypes
- Spatial indices using GiST

```
create extension postgis;
```

A Simple Example



A simple example

- Yosemite Park Ranger
- Tracking bears equipped with GPS transmitters
- Want to show all the bears on Google Maps

Bears (database migration)

```
create_table :bears do |t|
  t.column :lat, :float
  t.column :lon, :float
end

add_index :bears, :lat
add_index :bears, :lon
add_index :bears, [:lat, :lon]
```

Bears (model)

```
class Bear < ActiveRecord::Base  
  
  def self.bbox(sw_lon, sw_lat,  
               ne_lon, ne_lat)  
    self  
      .where( :lon => sw_lon..ne_lon )  
      .where( :lat => sw_lat..ne_lat )  
  end  
  
end
```

A PostGIS example (migration)

```
create_table :bears do |t|
  t.column :coordinates,
            :geometry,
            :srid => 4326
end

add_index :bears,
          :coordinates,
          :spatial => true
```

A PostGIS example (model)

```
def self.box(sw_lon, sw_lat, ne_lon, ne_lat)

  factory = Rails.application.spatial_factory

  sw = factory.point(sw_lon, sw_lat)
  nw = factory.point(sw_lon, ne_lat)
  ne = factory.point(ne_lon, ne_lat)
  se = factory.point(ne_lon, sw_lat)

  ring = factory.linear_ring([sw, nw, ne, se])
  bbox = factory.polygon(ring)

  self
    .where('ST_Intersects(coordinates, :bbox)',
           :bbox => bbox)
end
```

A PostGIS example

```
1_000_000.times do
  Bear.create!( ... )
end
```

1,000,000 bears

- PostGIS is 1.5x to 50x faster

Active Record PostGIS Adapter

- Migration support
- Automatic conversion of PostGIS datatypes to Ruby (RGeo) objects and back

```
gem 'pg'  
gem 'rgeo'  
gem 'activerecord-postgis-adapter'
```

Rails Migrations

```
add_column :bears,  
:coordinates,  
:geometry, :srid => 4326
```

```
add_index :bears,  
:coordinates,  
:spatial => true
```

Postgres Table

```
=> \d bears
                                         Table "public.bears"
   Column    |          Type
-----+-----
  id      | integer
coordinates | geometry(Geometry,4326)
Indexes:
  "pins_pkey" PRIMARY KEY, btree (id)
  "index_pins_on_coordinates" gist (coordinates)
```

PostGIS Data

```
=> select id, coordinates from bears limit 4;  
      id |          coordinates  
-----+-----  
      1 | 0101000020E61000002A8A632C341F664021805D8DDBEB4BC0  
      2 | 0101000020E61000004DF900A54A5866C0A2BAB6AC827D50C0  
      3 | 0101000020E61000002450EA628F5259C01C789C77C2883040  
      4 | 0101000020E610000038760C7B85443C4013206005DC2C48C0  
( 4 rows )
```

RGeo

```
##> bear=Bear.first
=> #<Bear id: 1, coordinates:
#<RGeo::Geos::CAPIPointImpl:0x3fd52ab501b4 "POINT
(176.9751188224921 -55.84263770165877)">>

##> bear.coordinates.x
=> 176.9751188224921

##> bear.coordinates.y
=> -55.84263770165877
```

More examples

```
# \d parks
```

Column	Type
id	integer
name	character varying(255)
boundary	geometry(Geometry,4326)

Indexes:

```
"parks_pkey" PRIMARY KEY, btree (id)
"index_parks_on_name" btree (name)
"index_parks_on_boundary" gist (polygon)
```

```
# select id, name, boundary from parks limit 2;
```

id	name	boundary
1	Yosemite	0103000020E6100000010000...
2	Yellowstone	0103000020E6100000010000...

How many bears are in Yosemite now?

```
##> park = Park.find_by_name('Yosemite')

##> bears = Bear.where('ST_Intersects(coordinates, :bounds)' ,
  :bounds => park.boundary)

##> bear_count = bears.count
```

How Many Bears in Yosemite and Yellowstone (Ruby)?

```
##> yosemite = Park.find_by_name('Yosemite')

##> yellowstone = Park.find_by_name('Yellowstone')

##> bounds = yosemite.boundary + yellowstone.boundary

##> bears = Bear.where('ST_Intersects(coordinates, :bounds)' ,
  :bounds => bounds)

##> bear_count = bears.count
```

How Many Bears in Yosemite and Yellowstone (SQL)?

```
select count(*) from bears
inner join parks
  on ST_Intersects(bears.coordinates,
                    parks.boundary)
where parks.name in ('Yosemite',
                      'Yellowstone');
```

Three parks closest to me?

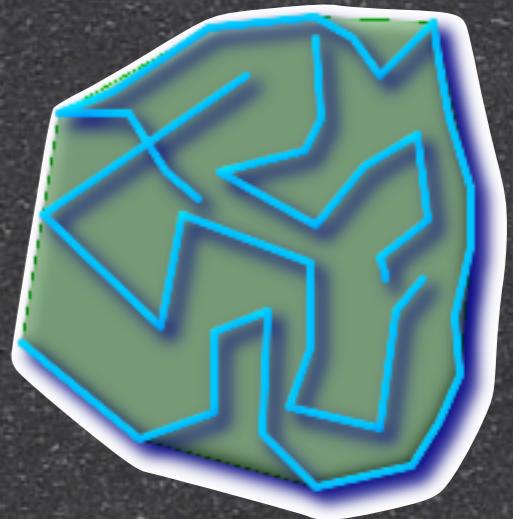
- Distance operator (KNN) is a feature of Postgres 9.1 and above

```
select id, name,  
       boundary <-> ST_Point(37.775, -122.44) as distance  
from parks  
order by distance  
limit 3;
```

What else is
possible?

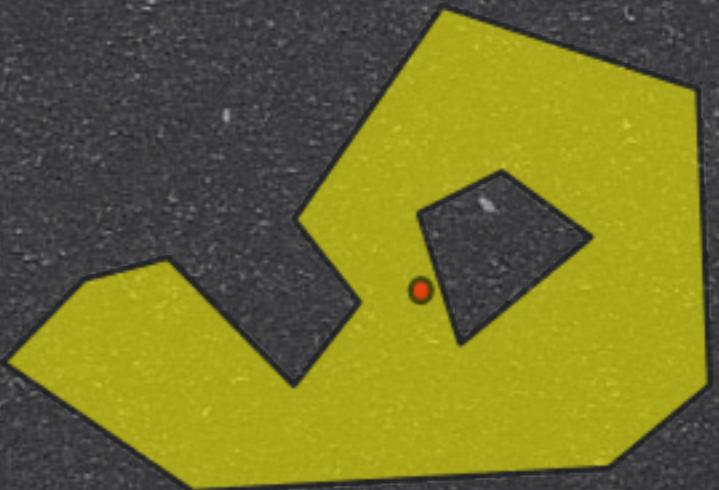
Geometry Simplification

- ST_Simplify
- ST_ConvexHull
- ST_ConcaveHull



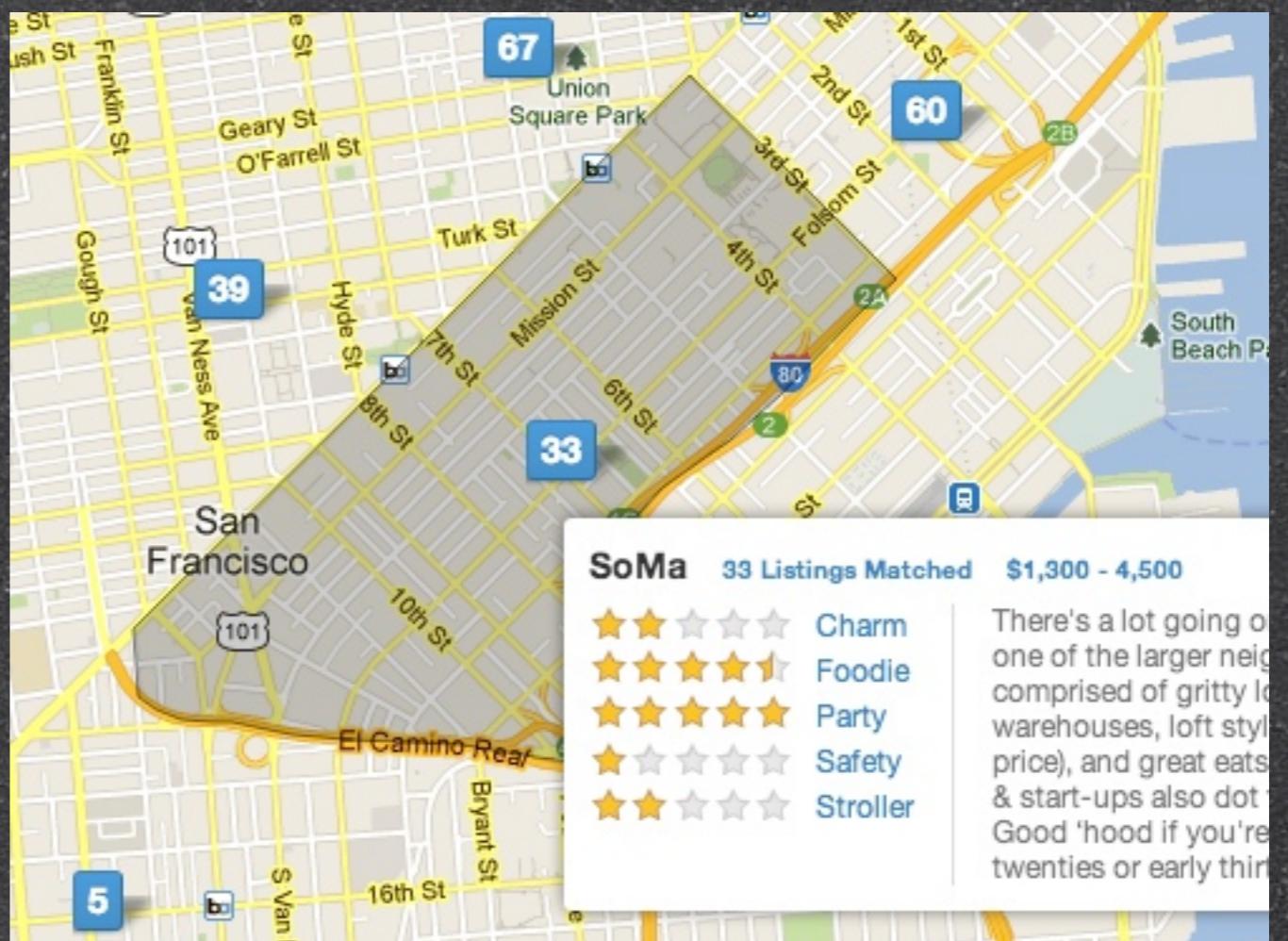
Spatial Relationships

- ST_Centroid
- ST_Contained
- ST_Area
- ST_Perimeter
- ST_DWithin



Format Conversions

- ST_AsGeoJSON
- ST_AsText



Do Try It at Home

- Heroku Postgres
 - <https://devcenter.heroku.com/articles/heroku-postgres-extensions-postgis-full-text-search>
- Postgres.app
 - <http://postgresapp.com/>
- `select postgis_full_version();`

Data Sources (free)

- US Census Data (Tiger)
 - <http://www.census.gov/geo/maps-data/data/tiger.html>
- Zillow Neighborhoods
 - <http://www.zillow.com/howto/api/neighborhood-boundaries.htm>

Data Sources (commercial)

- Maponics
 - <http://www.maponics.com/>
- Urban Mapping
 - <http://www.urbanmapping.com/>
- Onboard Informatics
 - <http://www.onboardinformatics.com/>

Links

- PostGIS
 - <http://postgis.net/>
- Active Record PostGIS Adapter
 - <https://github.com/dazuma/activerecord-postgis-adapter>
- RGeo
 - <https://github.com/dazuma/rgeo>

Q & A

We are hiring @apartmentlist!