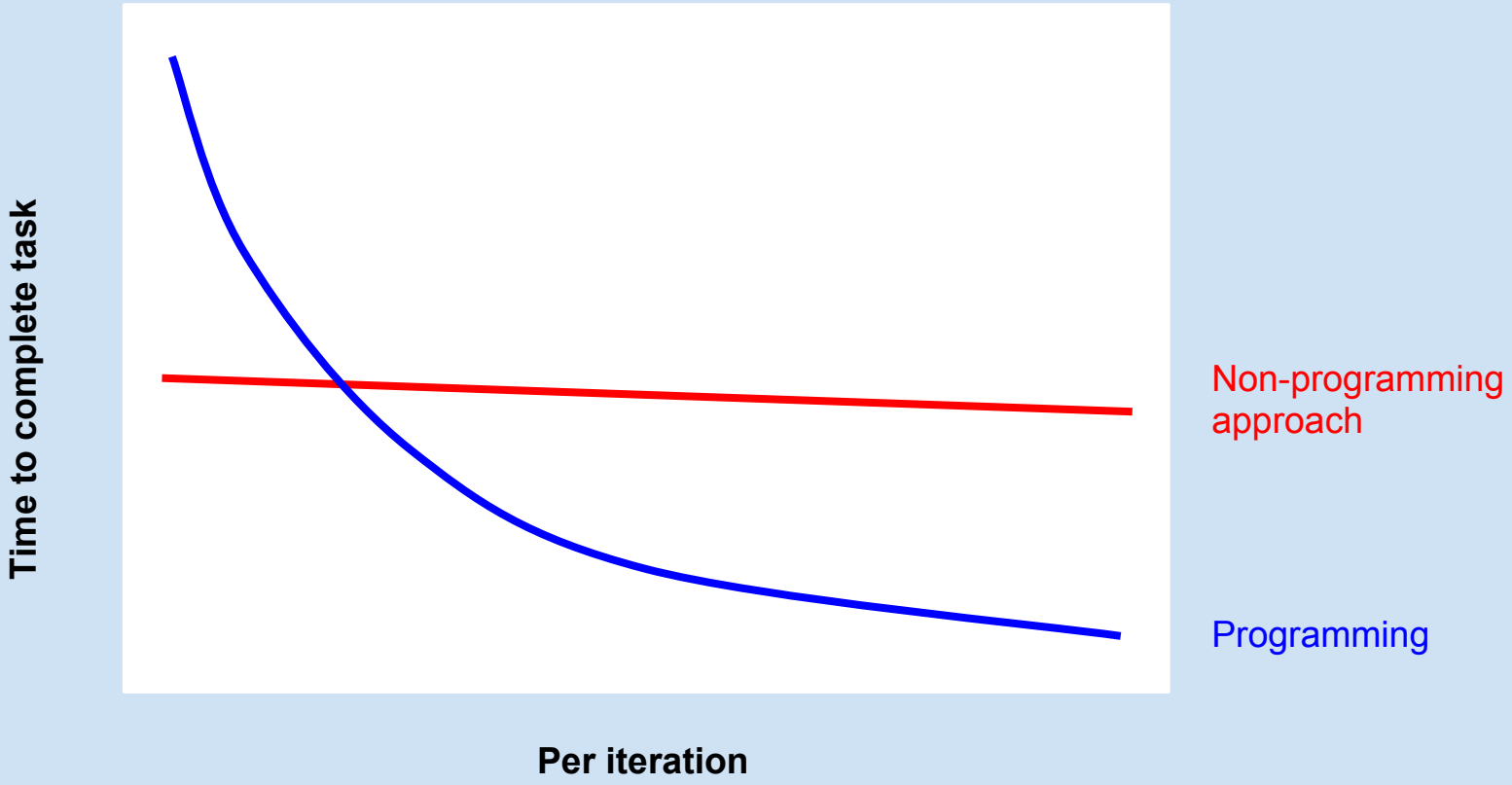
Exploring Open Source GIS Programming

Scott Parker, Katie Urey, Jack Newlevant, Mele Sax-Barnett

When to write a program

- When you expect your tool to have multiple uses and/or multiple iterations
- Something new or different that has to be fast, precise, accurate, reliable
- Need to match the user's model

When to write a program



What is open source?

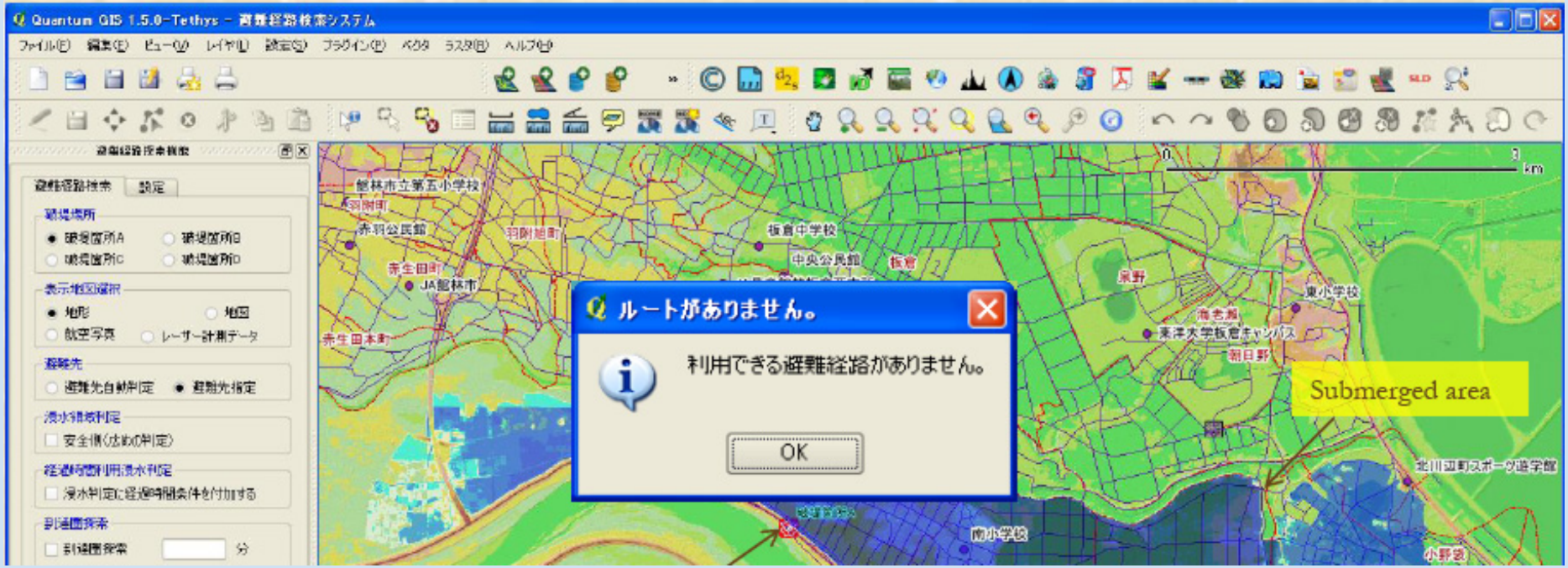
- Usually free to use
- Code base publicly available, public encouraged to contribute
- Can just use it, or also write code to make it better or fit your application
- Open source is not the same as open data

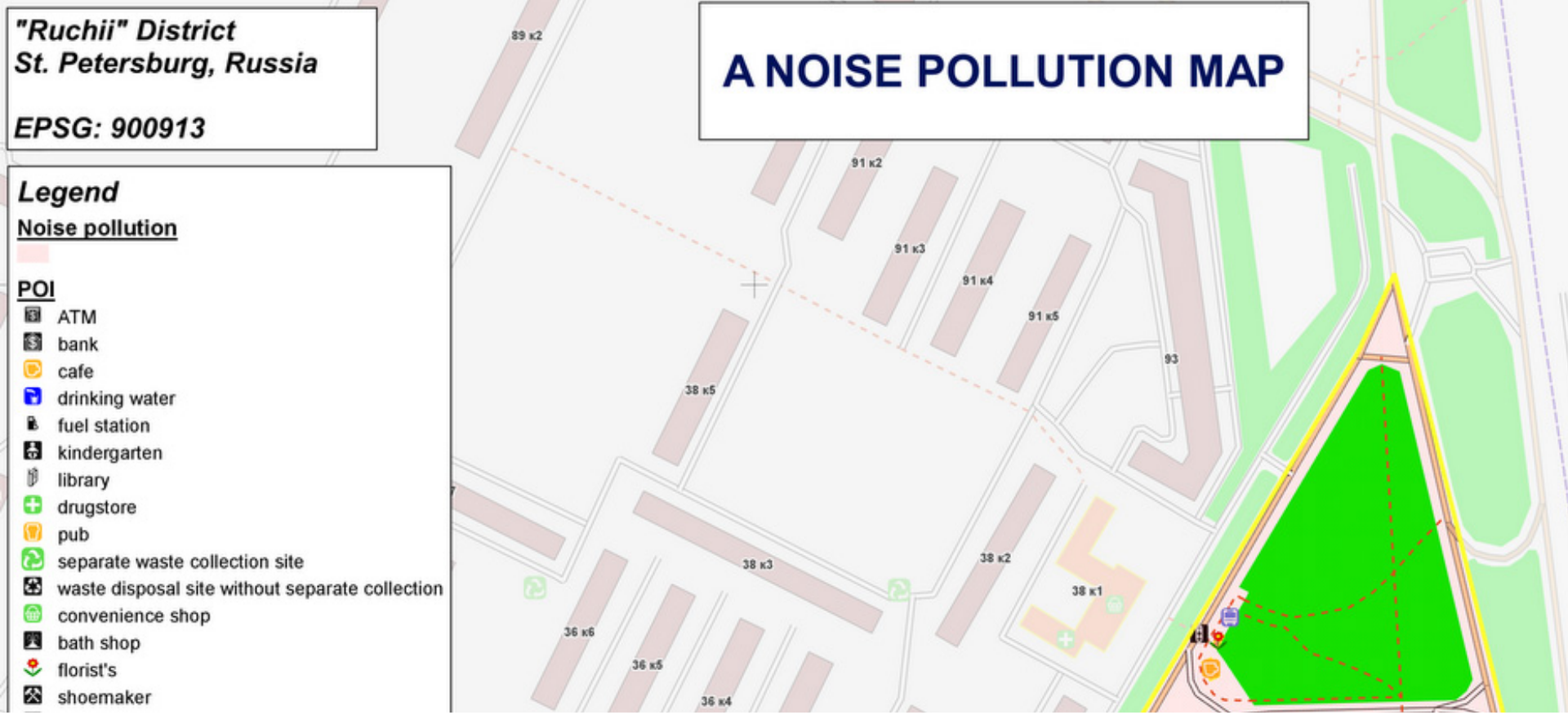
When to use open source

- When the overhead of acquiring expensive tools is high
  - Ideal for community groups
- When you don't want to explain what you're doing or ask for permission to do a unique project
- When you need to know how everything works

Community GIS Examples

There are no route to any shelter





Do you need maps?

Are you embroiled in an cartographic dispute? Do you disagree with the official version of your geography?..

This community began as the Grassroots Mapping project, an effort to produce Do-It-Yourself satellite imagery with balloons and kites, most notably during the **2010 BP oil spill.** We are now broadening our scope to explore new inexpensive and community-led means to measure and explore environmental and social issues

Programming in open source

Benefits

- *Full control if desired*
- *Structure for collaboration*
- *Richer toolbox*
- *Multi platform*
- *Programming skills you will develop have broad application*

Disadvantages

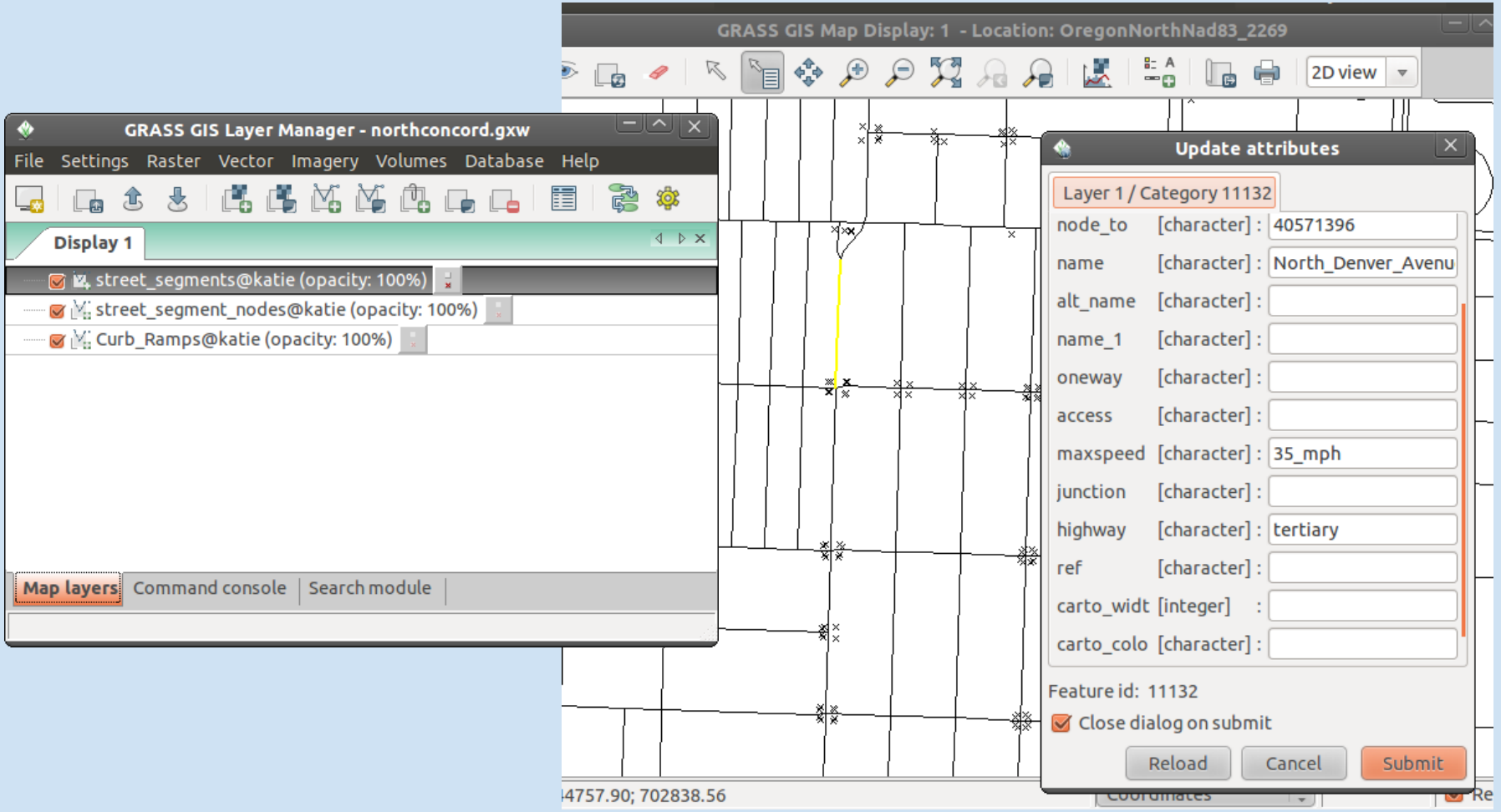
- *Time & cost to learn and maintain development environment*
- *Requires more general purpose programming skills*

Open source GIS tools

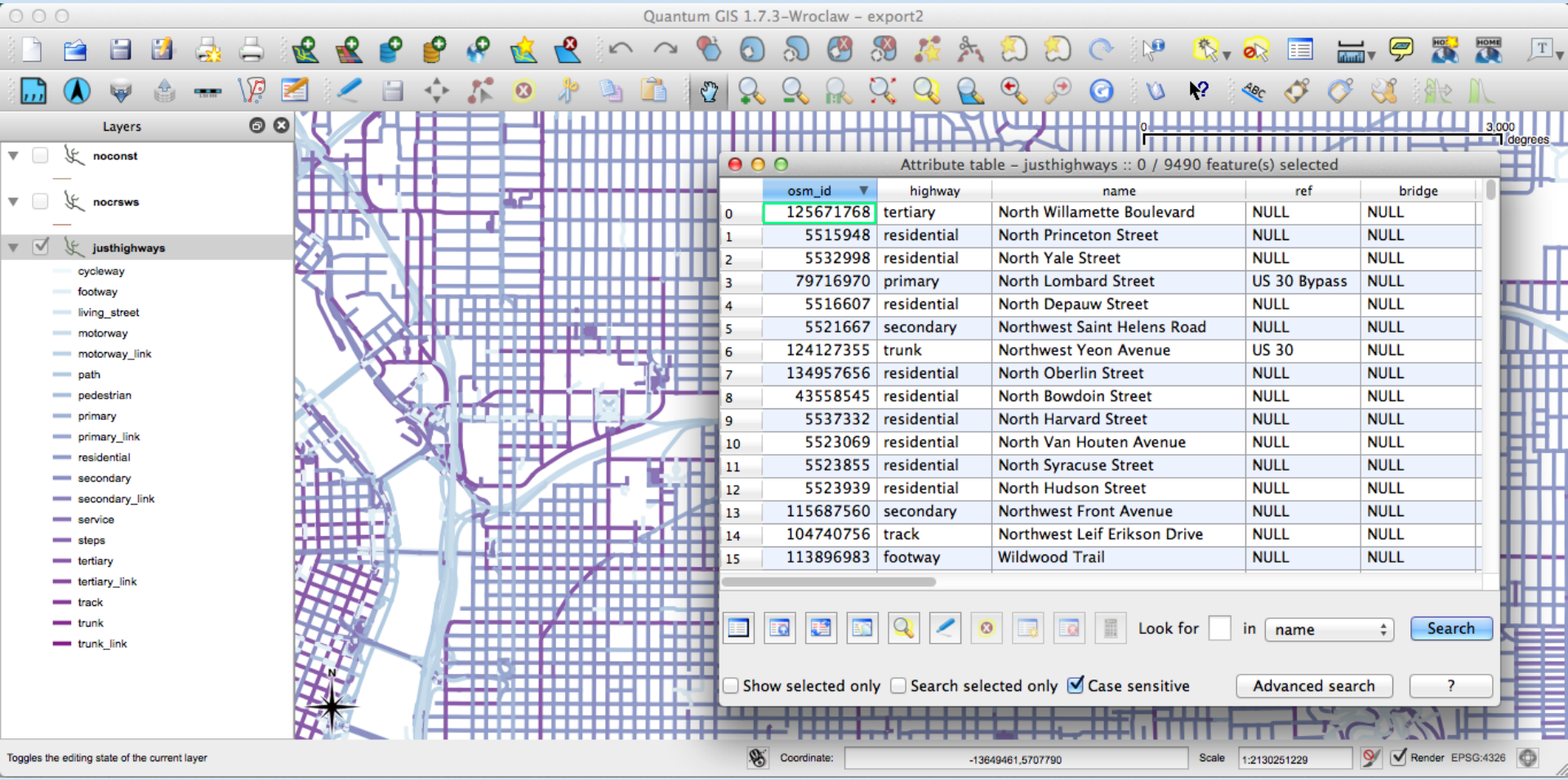
- GRASS - Desktop GIS
- QGIS - Desktop GIS
- PostGIS - Spatial relational database management system +
- OpenStreetMap - Open spatial data

Bonus: they all play well together!

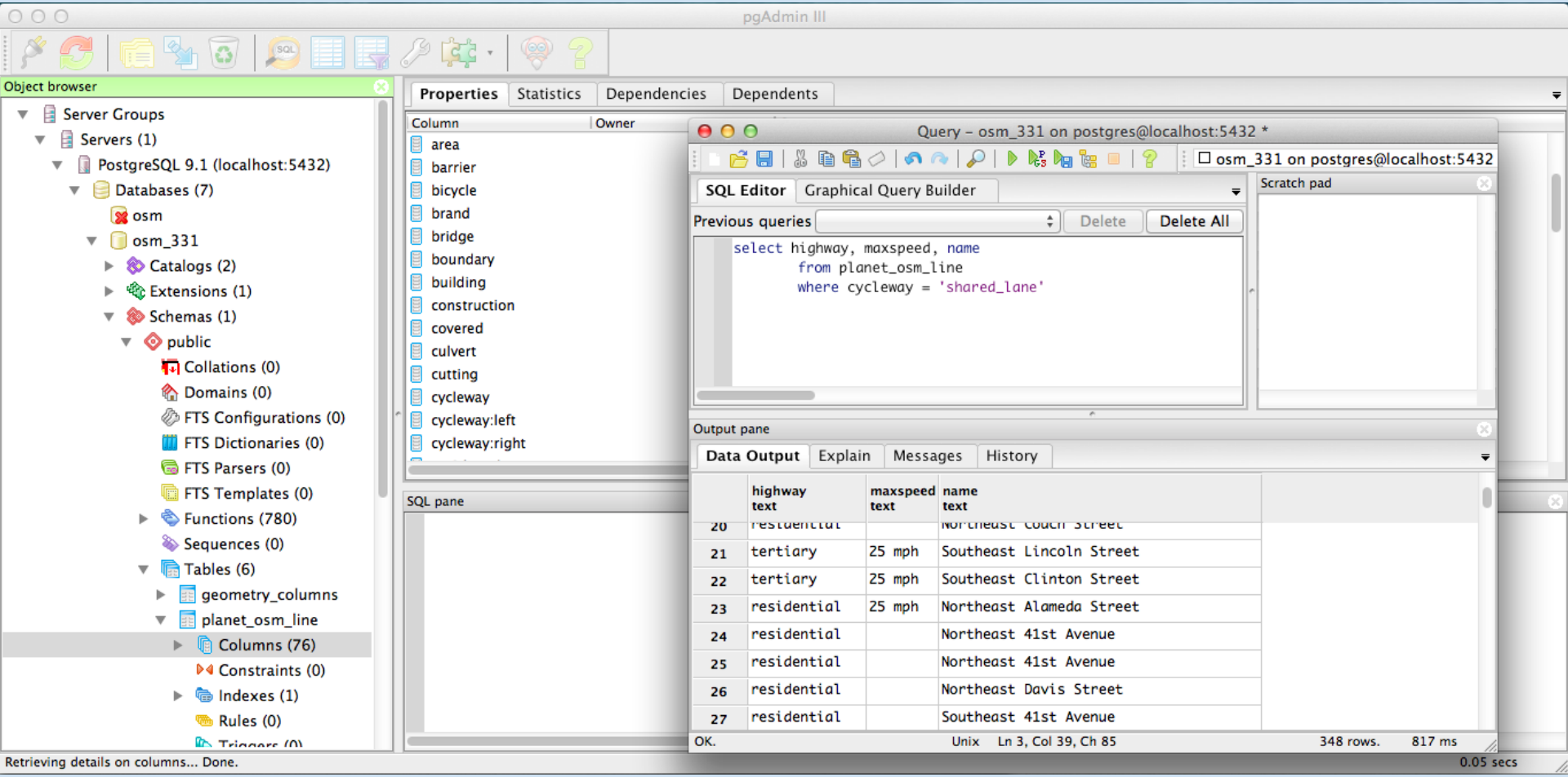
GRASS


GRASS GIS Map Display: 1 - Location: OregonNorthNad83_2269
2D view
GRASS GIS Layer Manager - northconcord.gxw
File Settings Raster Vector Imagery Volumes Database Help
Display 1
street_segments@katie (opacity: 100%)
street_segment_nodes@katie (opacity: 100%)
Curb_Ramps@katie (opacity: 100%)
Map layers
Command console
Search module
Update attributes
Layer 1 / Category 11132
node_to [character] : 40571396
name [character] : North_Denver_Avenu
alt_name [character] :
name_1 [character] :
oneway [character] :
access [character] :
maxspeed [character] : 35_mph
junction [character] :
highway [character] : tertiary
ref [character] :
carto_widt [integer] :
carto_colo [character] :
Feature id: 11132
Close dialog on submit
Reload
Cancel
Submit
4757.90; 702838.56

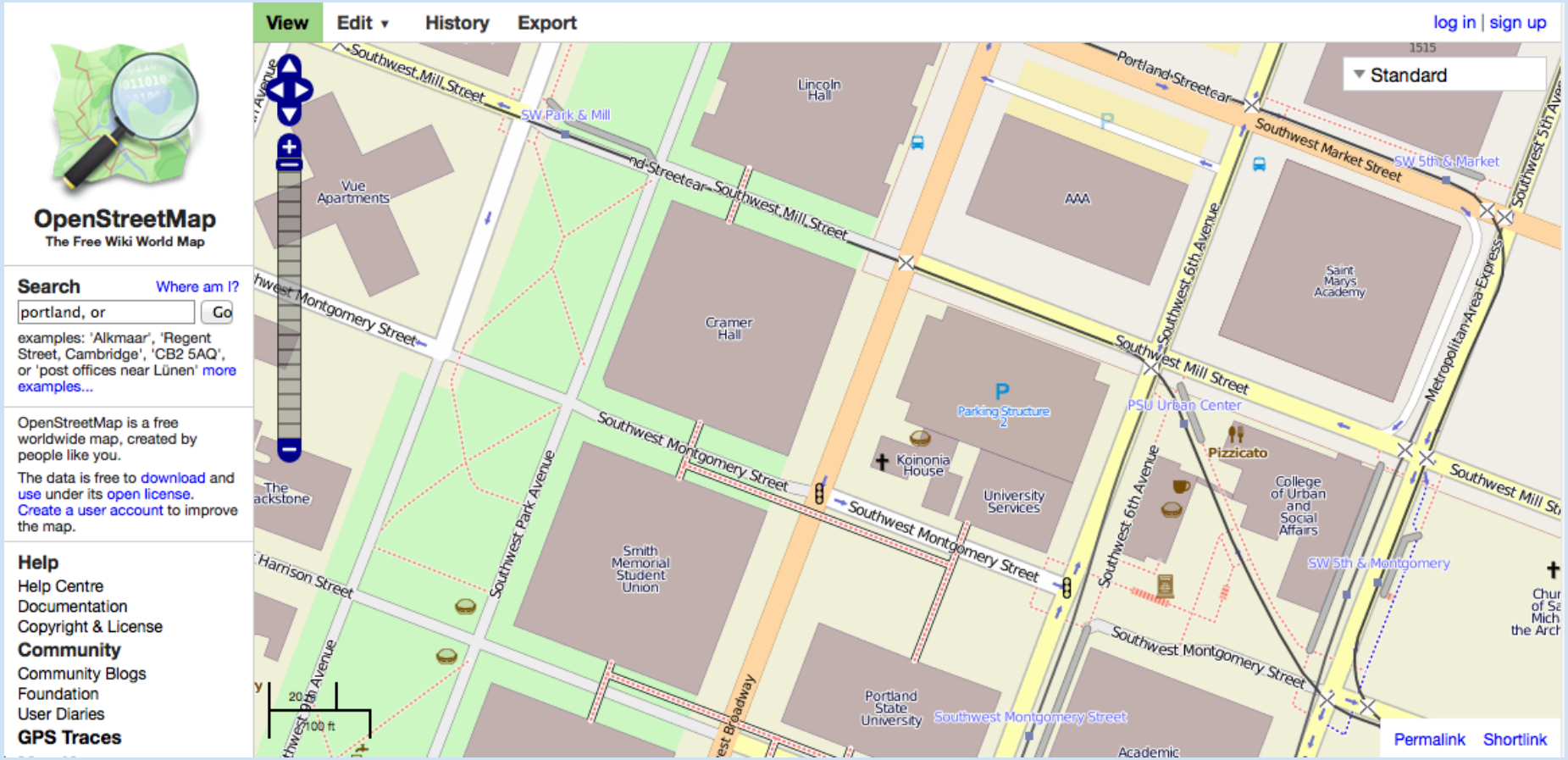
Quantum GIS


Quantum GIS 1.7.3-Wroclaw - export2
Layers
noconst
nocrsws
justhighways
cycleway
footway
living_street
motorway
motorway_link
path
pedestrian
primary
primary_link
residential
secondary
secondary_link
service
steps
tertiary
tertiary_link
track
trunk
trunk_link
Attribute table – justhighways :: 0 / 9490 feature(s) selected
osm_id highway name ref bridge
0 125671768 tertiary North Willamette Boulevard NULL NULL
1 5515948 residential North Princeton Street NULL NULL
2 5532998 residential North Yale Street NULL NULL
3 79716970 primary North Lombard Street US 30 Bypass NULL
4 5516607 residential North Depauw Street NULL NULL
5 5521667 secondary Northwest Saint Helens Road NULL NULL
6 124127355 trunk Northwest Yeon Avenue US 30 NULL
7 134957656 residential North Oberlin Street NULL NULL
8 43558545 residential North Bowdoin Street NULL NULL
9 5537332 residential North Harvard Street NULL NULL
10 5523069 residential North Van Houten Avenue NULL NULL
11 5523855 residential North Syracuse Street NULL NULL
12 5523939 residential North Hudson Street NULL NULL
13 115687560 secondary Northwest Front Avenue NULL NULL
14 104740756 track Northwest Leif Erikson Drive NULL NULL
15 113896983 footway Wildwood Trail NULL NULL
Look for in name Search
Show selected only Search selected only Case sensitive Advanced search ?
0 3,000 degrees
Toggles the editing state of the current layer
Coordinate: -13649461,5707790 Scale 1:2130251229 Render EPSG:4326

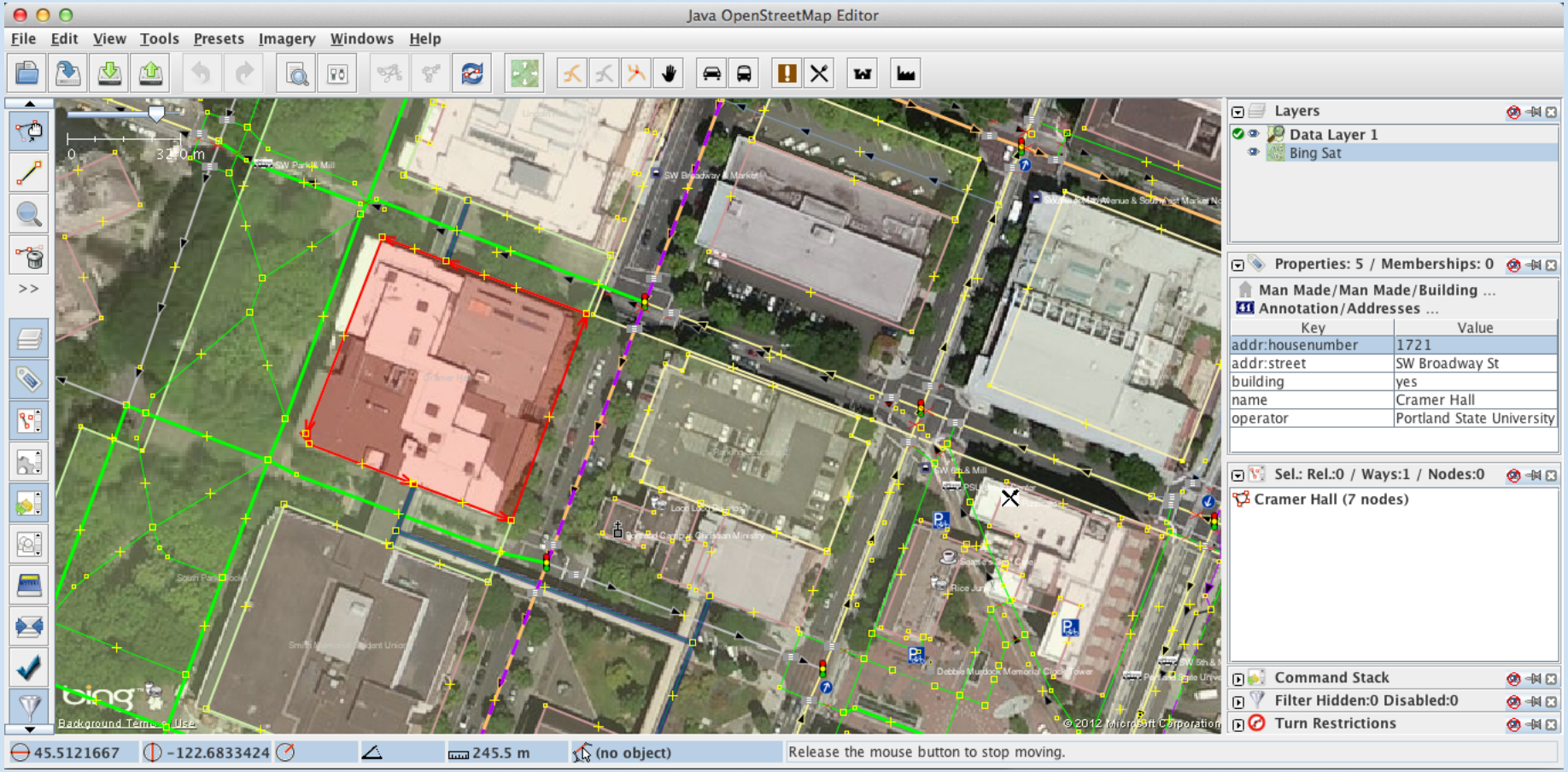
PostGIS


pgAdmin III
Object browser
Server Groups
Servers (1)
PostgreSQL 9.1 (localhost:5432)
Databases (7)
osm
osm_331
Catalogs (2)
Extensions (1)
Schemas (1)
public
Collations (0)
Domains (0)
FTS Configurations (0)
FTS Dictionaries (0)
FTS Parsers (0)
FTS Templates (0)
Functions (780)
Sequences (0)
Tables (6)
geometry_columns
planet_osm_line
Columns (76)
Constraints (0)
Indexes (1)
Rules (0)
Properties
Statistics
Dependencies
Dependents
Column
Owner
area
barrier
bicycle
brand
bridge
boundary
building
construction
covered
culvert
cutting
cycleway
cycleway:left
cycleway:right
SQL pane
Query - osm_331 on postgres@localhost:5432 *
osm_331 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
select highway, maxspeed, name
from planet_osm_line
where cycleway = 'shared_lane'
Scratch pad
Output pane
Data Output
Explain
Messages
History
highway text
maxspeed text
name text
21 tertiary 25 mph Southeast Lincoln Street
22 tertiary 25 mph Southeast Clinton Street
23 residential 25 mph Northeast Alameda Street
24 residential Northeast 41st Avenue
25 residential Northeast 41st Avenue
26 residential Northeast Davis Street
27 residential Southeast 41st Avenue
OK.
Unix Ln 3, Col 39, Ch 85
348 rows.
817 ms
Retrieving details on columns... Done.
0.05 secs

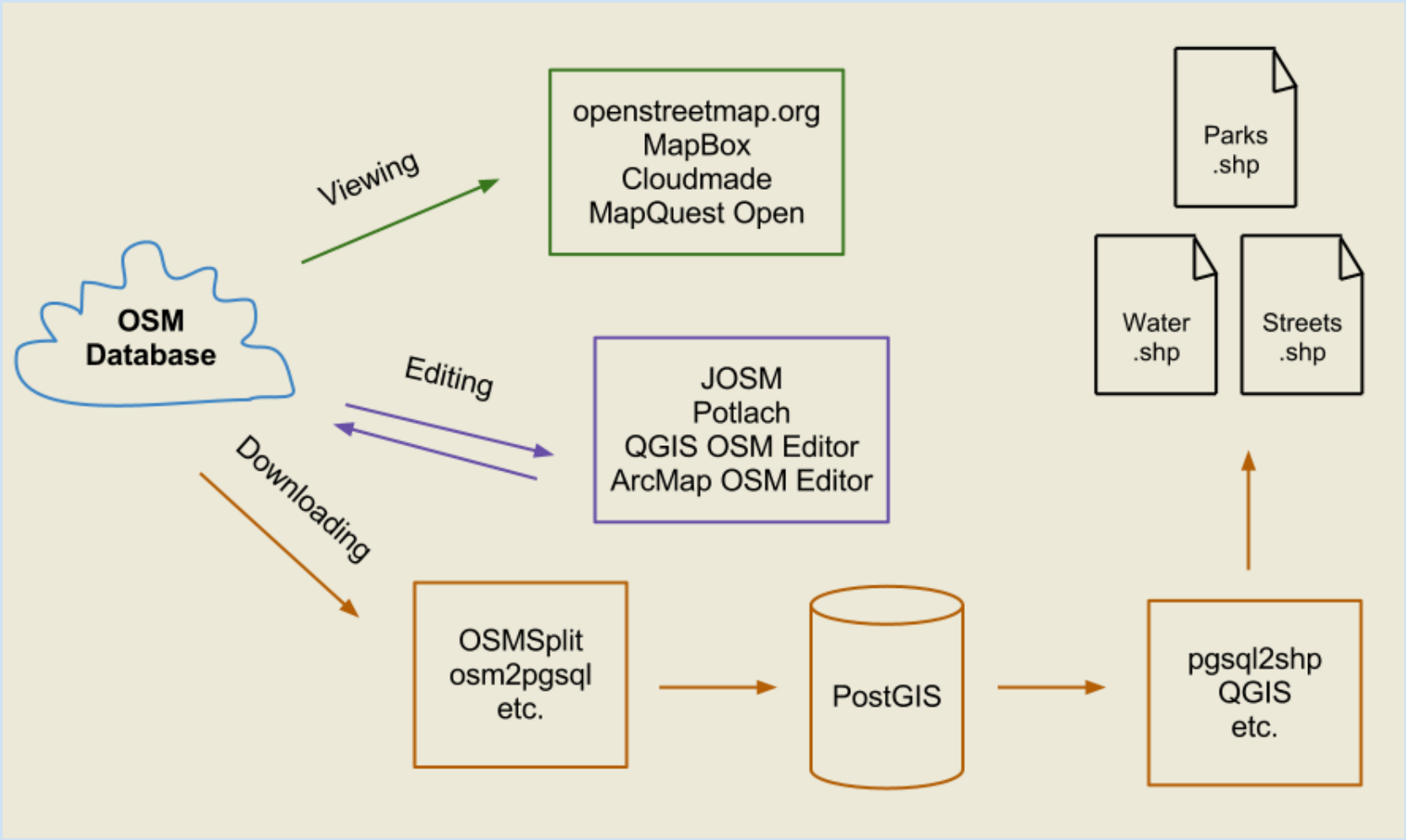
OpenStreetMap: open data



OpenStreetMap: open data



OpenStreetMap: open data

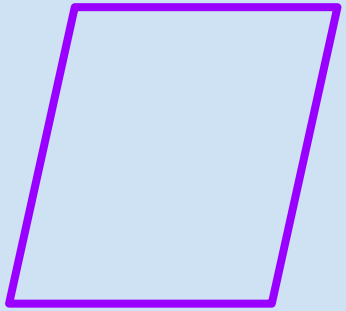


Example App:
Curb ramp inventory

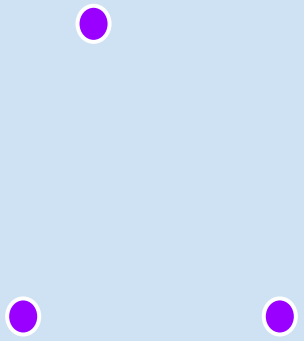
- Task: Create a mobile-friendly website that can be used by volunteers to do a curb ramp inventory
  - Free or very low cost
- Tools:
  - PostGIS
  - CartoDB
  - OpenLayers
  - HTML, javascript, jQuery, CSS

Step 1: Prepare dataset

- We had previously generated a walkway network from a street centerline file that included crosswalks, but what we needed was corners
- So, how do you turn this:



... into this?



... without ArcGIS?

Step 1: Prepare dataset

- Answer: With PostGIS!
- PostGIS not only manages your spatial data, it can run a number of really useful spatial functions *very quickly*
- All you have to do is write up some SQL
- ...and reuse it whenever you want!

7.8. Spatial Relationships and Measurements

ST_Area – Returns the area of the surface if it is a polygon or multi-polygon. For "geometry" type area is in SRID units. For "geography" area is in square meters.
ST_Azimuth – Returns the angle in radians from the horizontal of the vector defined by pointA and pointB
ST_Centroid – Returns the geometric center of a geometry.
ST_ClosestPoint – Returns the 2-dimensional point on g1 that is closest to g2. This is the first point of the shortest line.
ST_Contains – Returns true if and only if no points of B lie in the exterior of A, and at least one point of the interior of B lies in the interior of A.
ST_ContainsProperly – Returns true if B intersects the interior of A but not the boundary (or exterior). A does not contain properly itself, but does contain itself.
ST_Covers – Returns 1 (TRUE) if no point in Geometry B is outside Geometry A. For geography: if geography point B is not outside Polygon Geography A
ST_CoveredBy – Returns 1 (TRUE) if no point in Geometry/Geography A is outside Geometry/Geography B
ST_Crosses – Returns TRUE if the supplied geometries have some, but not all, interior points in common.
ST_LineCrossingDirection – Given 2 linestrings, returns a number between -3 and 3 denoting what kind of crossing behavior. 0 is no crossing.
ST_Disjoint – Returns TRUE if the Geometries do not "spatially intersect" - if they do not share any space together.
ST_Distance – For geometry type Returns the 2-dimensional cartesian minimum distance (based on spatial ref) between two geometries in projected units. For geography type defaults to return spheroidal minimum distance between two geographies in meters.
ST_HausdorffDistance – Returns the Hausdorff distance between two geometries. Basically a measure of how similar or dissimilar 2 geometries are. Units are in the units of the spatial reference system of the geometries.

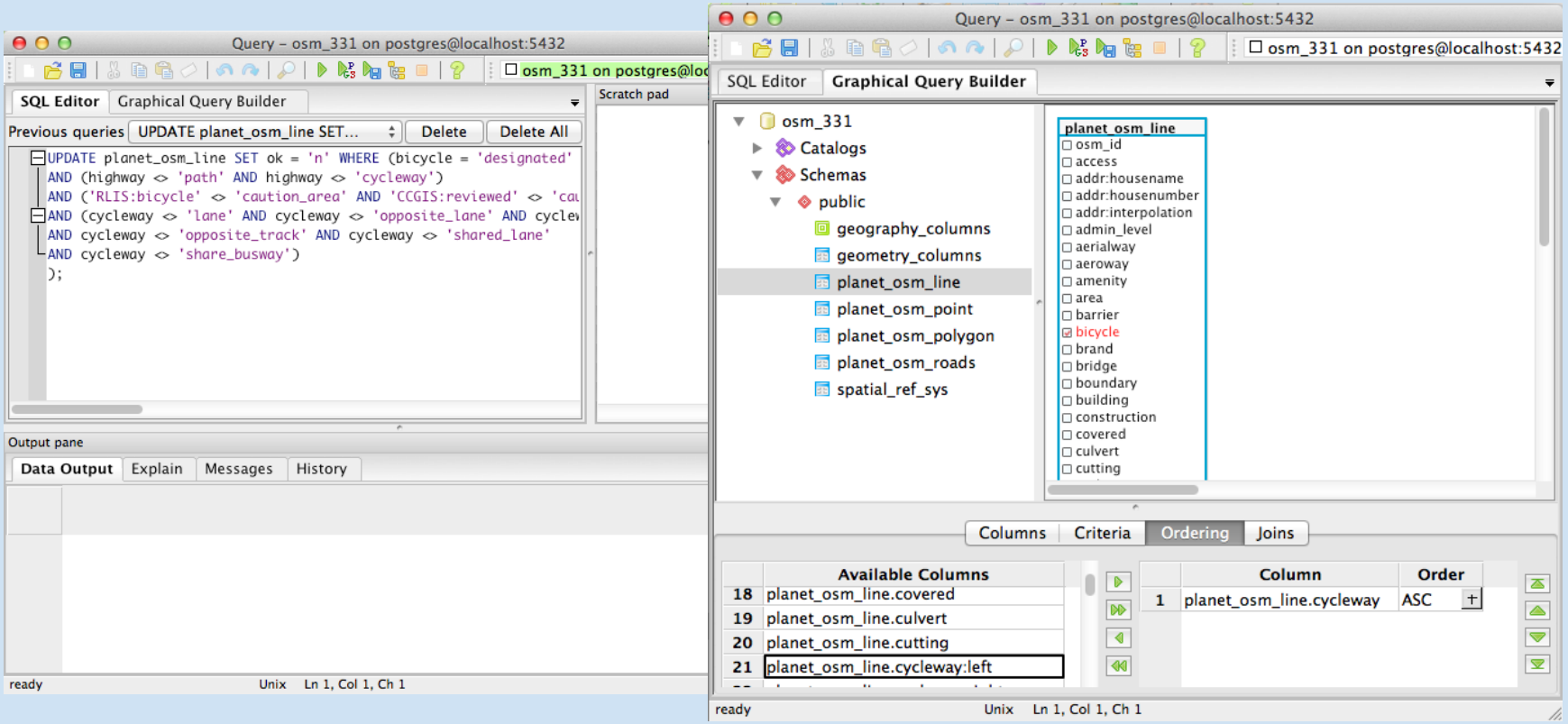
Step 1: Prepare dataset

- SQL is easy to learn
- PostGIS/PostgreSQL has great documentation and a broad user base
- This code creates a new table, then does an inner join of the crosswalk segments, using the function ST_StartPoint() to get the geometry of the point where the second of two touching crosswalks begins
- Finally, it brings in street names from a related table to give you a corner of X street and Y way

```
create table public.corner_pts (
    id serial primary key,
    intersection_id bigint,
    fm_bearing int,
    st_left_id bigint,
    st_right_id bigint,
    st_left_name text,
    st_right_name text,
    alignment text,
    the_geom geometry
);

INSERT INTO public.corner_pts (intersection_id, fm_bearing,
    st_left_id, st_right_id, alignment, the_geom)
SELECT DISTINCT
    c1.to_end_id as intersection_id,
    c1.fm_bearing,
    c1.to_join_id as st_left_id,
    c1.street_seg as st_right_id,
    c1.alignment,
    ST_StartPoint(c1.the_geom)
FROM public.corners c1, public.corners c2
where (c1.street_seg= c2.to_join_ID --and c1.to_end_id=39265
);

update public.corner_pts set st_right_name = s.full_name
from public."Streets_pdx" s
where st_right_id = s.localid;

update public.corner_pts set st_left_name = s.full_name
from public."Streets_pdx" s
where st_left_id = s.localid;
```

Creating SQL for PostGIS

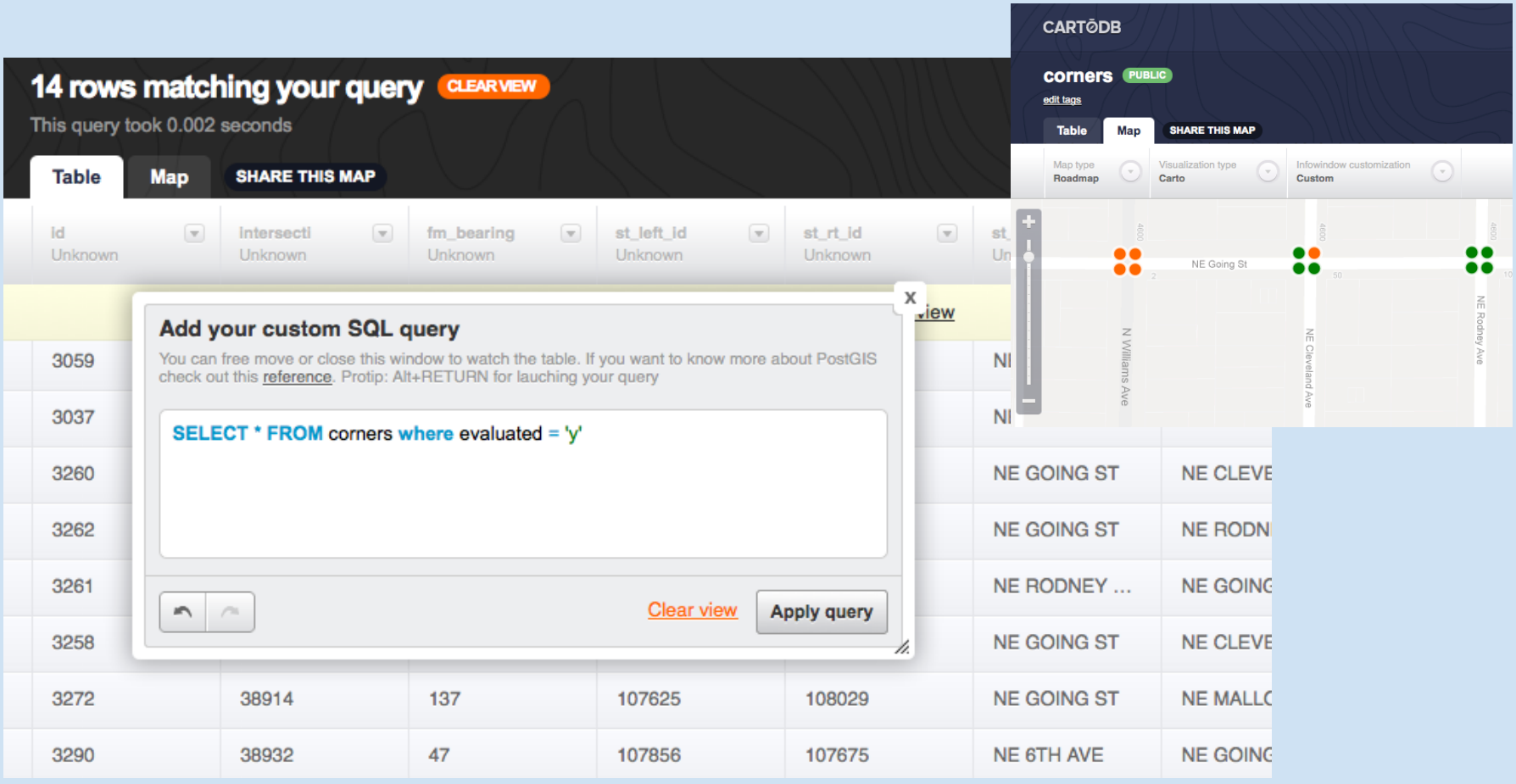
Write your code in a text editor of your choice and paste it in the query window, or use the Graphical Query Builder

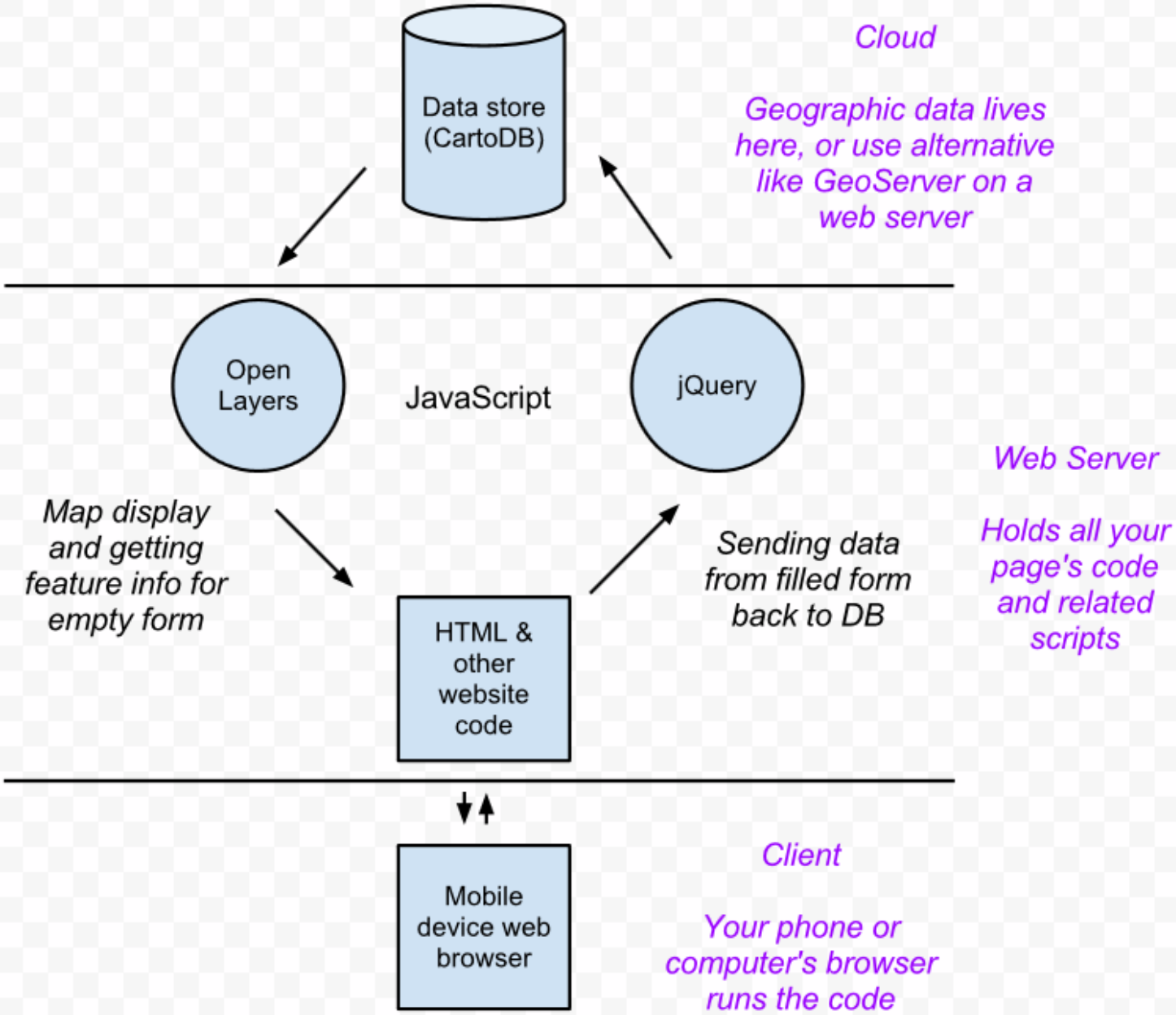


Step 2: Host the data

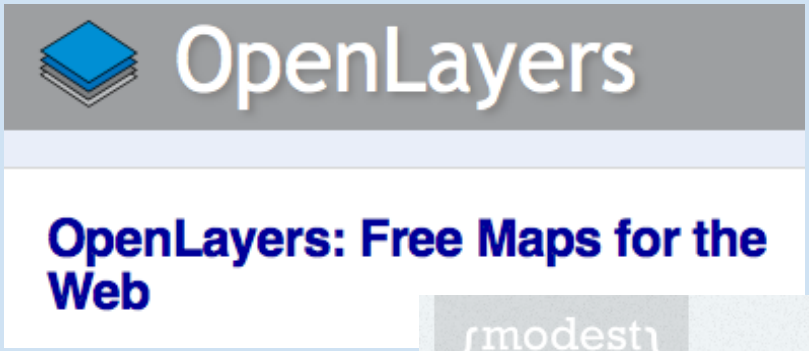
- There are many ways to display geographic data online, though you have two main options:
  - Tile your data or otherwise turn it into images
  - Host the data itself, making it truly interactive
  - With either, you can host data in the cloud or on your own web server
- Though affordable and easy open source options like MapBox & TileMill could provide some interactivity, a new tool called CartoDB was recently released that offered us exactly what we needed
- CartoDB = PostGIS in the cloud

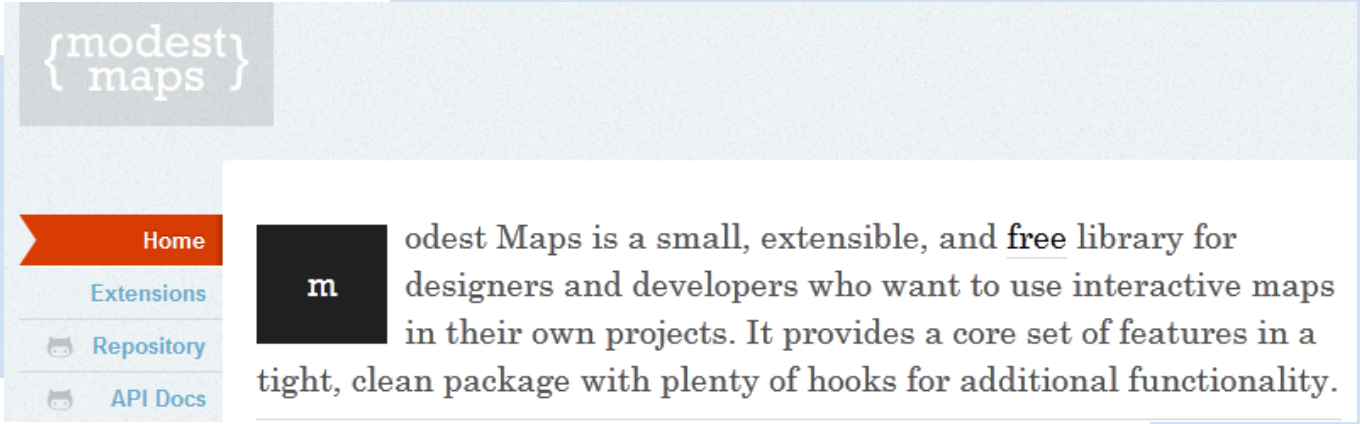
CartoDB web interface

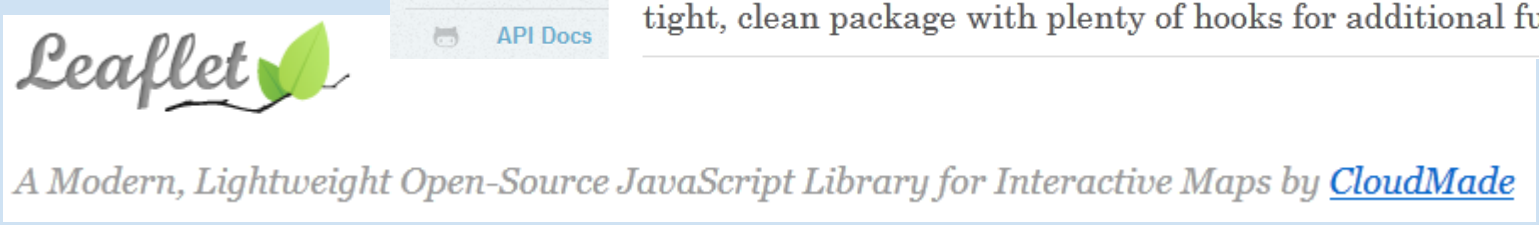



Data store
(CartoDB)
Cloud
Geographic data lives here, or use alternative like GeoServer on a web server
Open Layers
JavaScript
jQuery
Map display and getting feature info for empty form
Sending data from filled form back to DB
HTML & other website code
Web Server
Holds all your page's code and related scripts
Mobile device web browser
Client
Your phone or computer's browser runs the code

Step 3: Choose a web mapping library

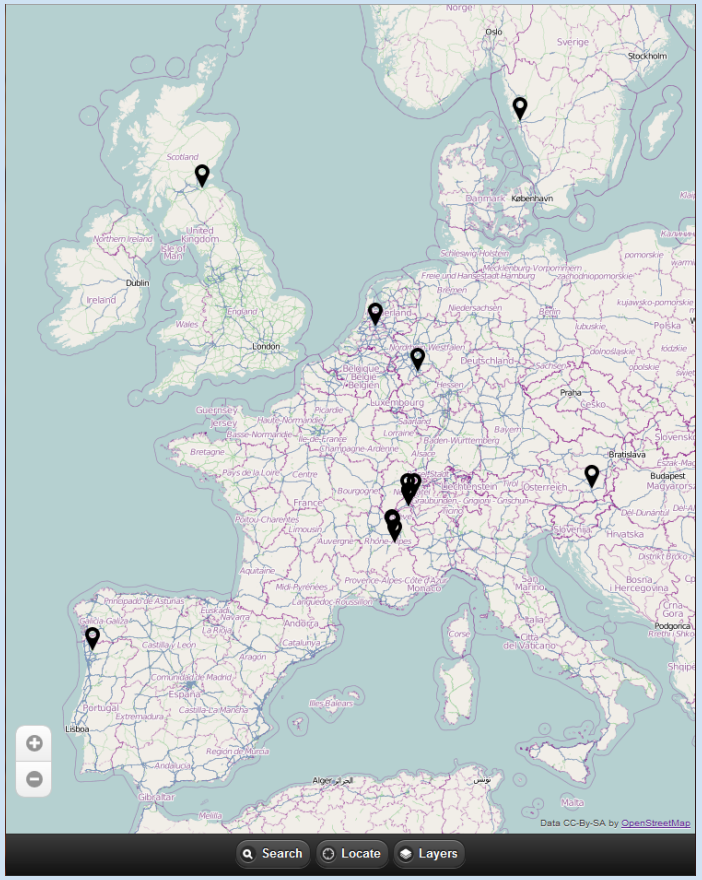


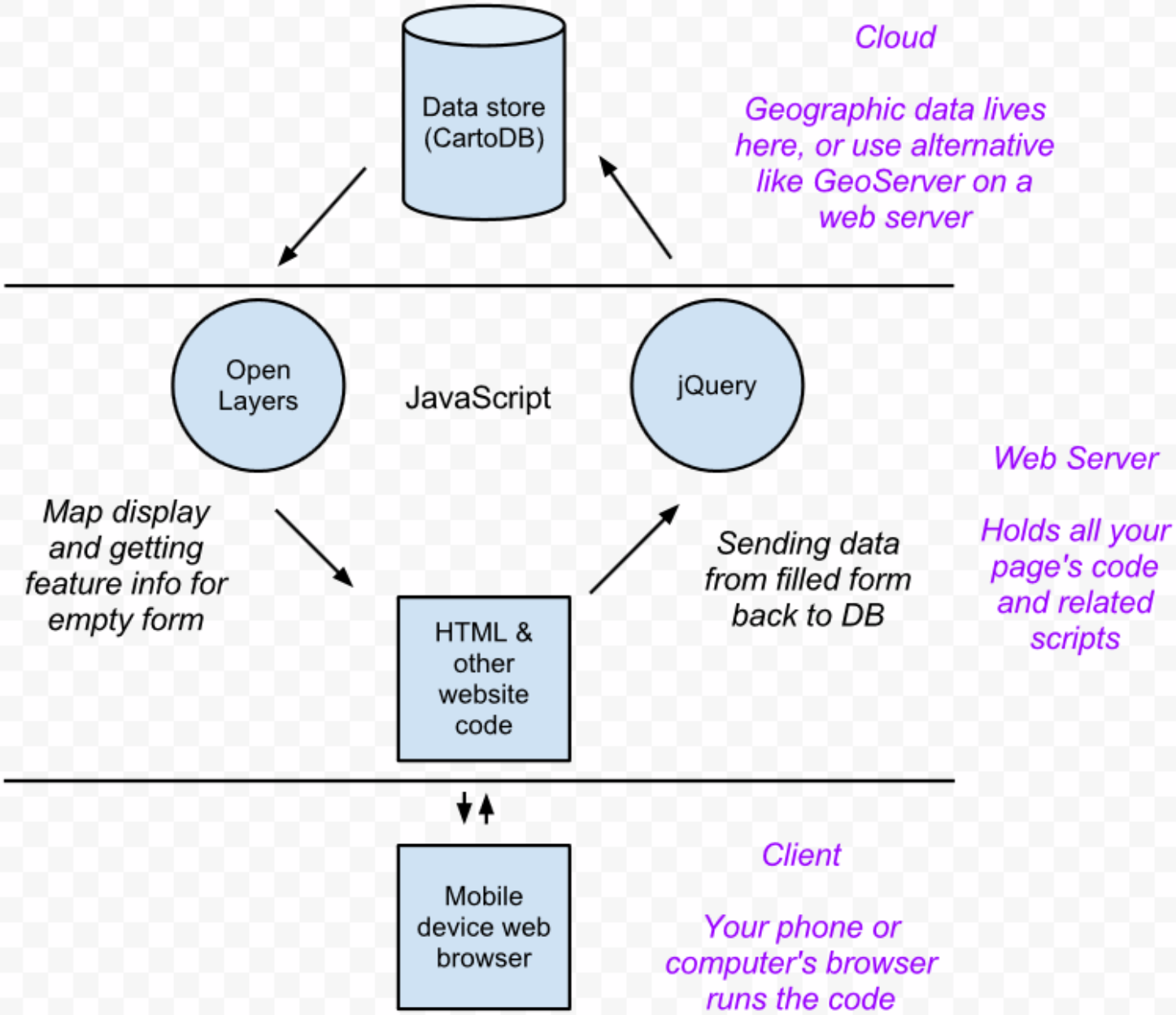




Step 3: Choose a web mapping library

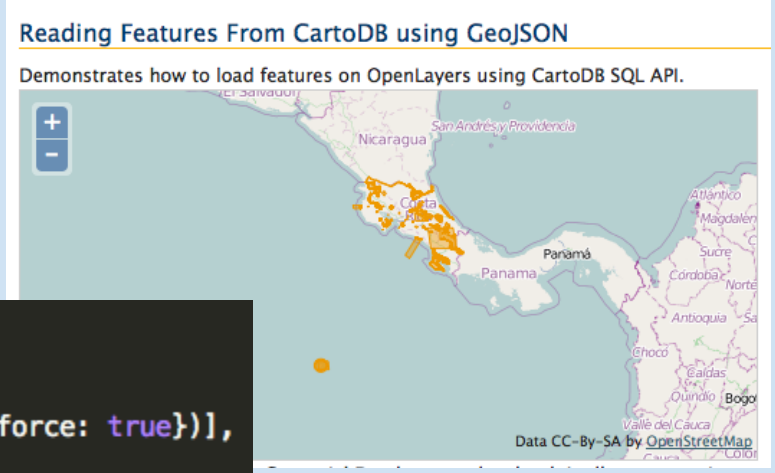
- I chose OpenLayers, since I was familiar with an example for mobile devices with most of the functionality I needed -- included start up HTML, javascript, jQuery, and CSS
- OpenLayers is powerful and well-established, but Leaflet or other libraries may have worked well too




Data store
(CartoDB)
Cloud
Geographic data lives here, or use alternative like GeoServer on a web server
Open Layers
JavaScript
jQuery
Map display and getting feature info for empty form
Sending data from filled form back to DB
HTML & other website code
Web Server
Holds all your page's code and related scripts
Mobile device web browser
Client
Your phone or computer's browser runs the code

Step 4: Try to make it all work together

- Telling OpenLayers how to talk to CartoDB
  - Map - no examples

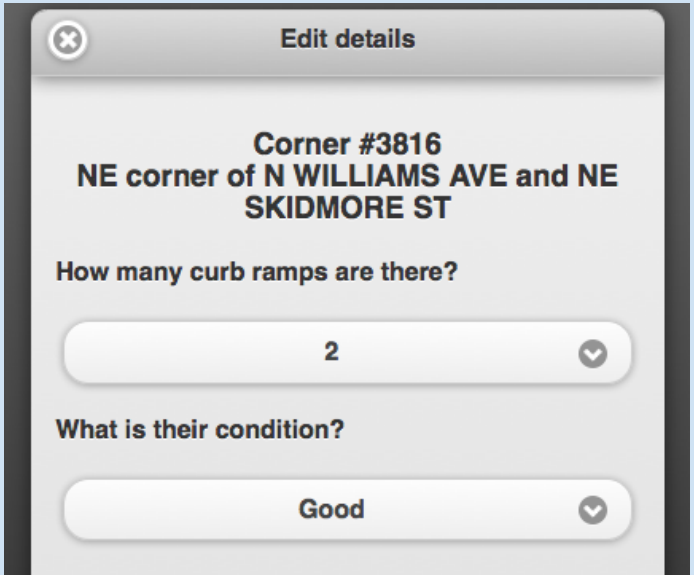



```
var cartoDB = new OpenLayers.Layer.Vector("Corners", {
    projection: gg,
    strategies: [new OpenLayers.Strategy.BBOX(),
        new OpenLayers.Strategy.Refresh({interval: 60000, force: true})],
        protocol: new OpenLayers.Protocol.Script({
        url: "http://pdxmele.cartodb.com/api/v2/sql",
        params: {
            q: "select * from corners where evaluated is null",
            format:"geojson"
        },
        format: new OpenLayers.Format.GeoJSON({
            ignoreExtraDims: true
        }),
        callbackKey: "callback"
    })
});
```

Ask for help!

Step 4: Try to make it all work together

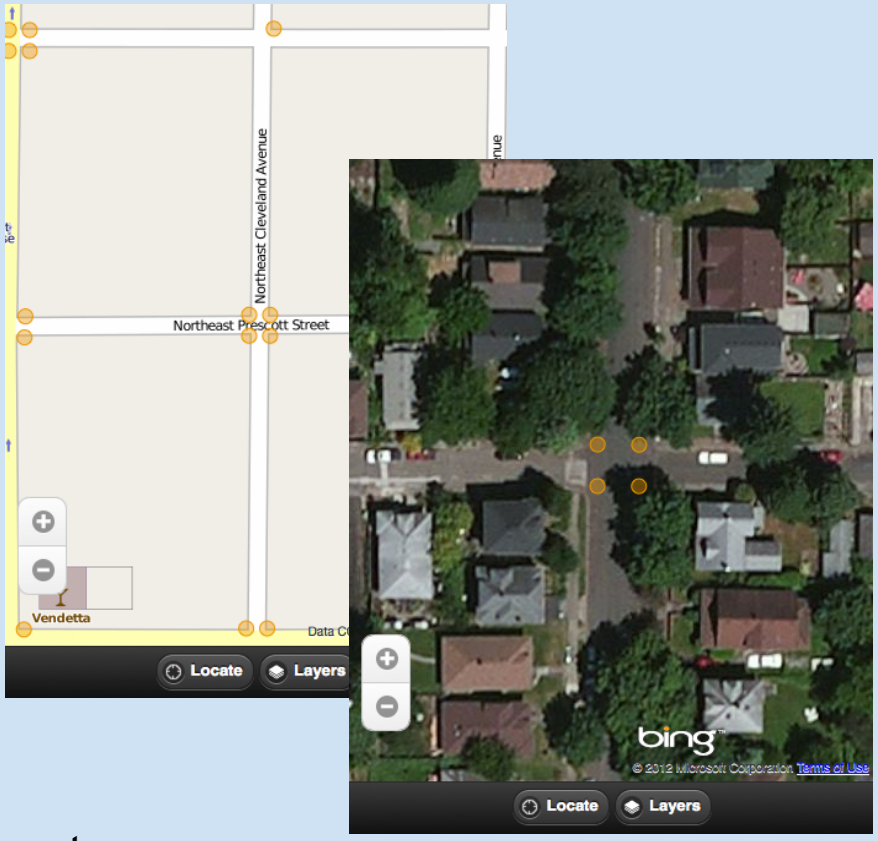
- Data
  - Populating forms
  - Sending forms
- Ask for help!




```
<script>
  //this function sends the form info to CartoDB
  function processCornerForm() {

    //grab the information from the form
    var id = document.getElementById("cornerID").
      innerHTML;
    var number = document.cornerForm.number.value;
    var condition = document.cornerForm.condition.value;
    var slope = document.cornerForm.slope.value;

    //put together the query
    var query = "q=UPDATE corners SET evaluated = 'y',
      number = '"+number+"', condition = '"+condition+"
      ', slope = '"+slope+"' where id = '"+id+"
      '&api_key=x";

    //for debugging
    //alert(query);
    //document.getElementById("querytest").innerHTML =
    query;

    //post the query!
    $.post("http://pdxmele.cartodb.com/api/v2/sql", query
      , alert ("Successfully posted to database"));
```

Step 5: Refine and enjoy!

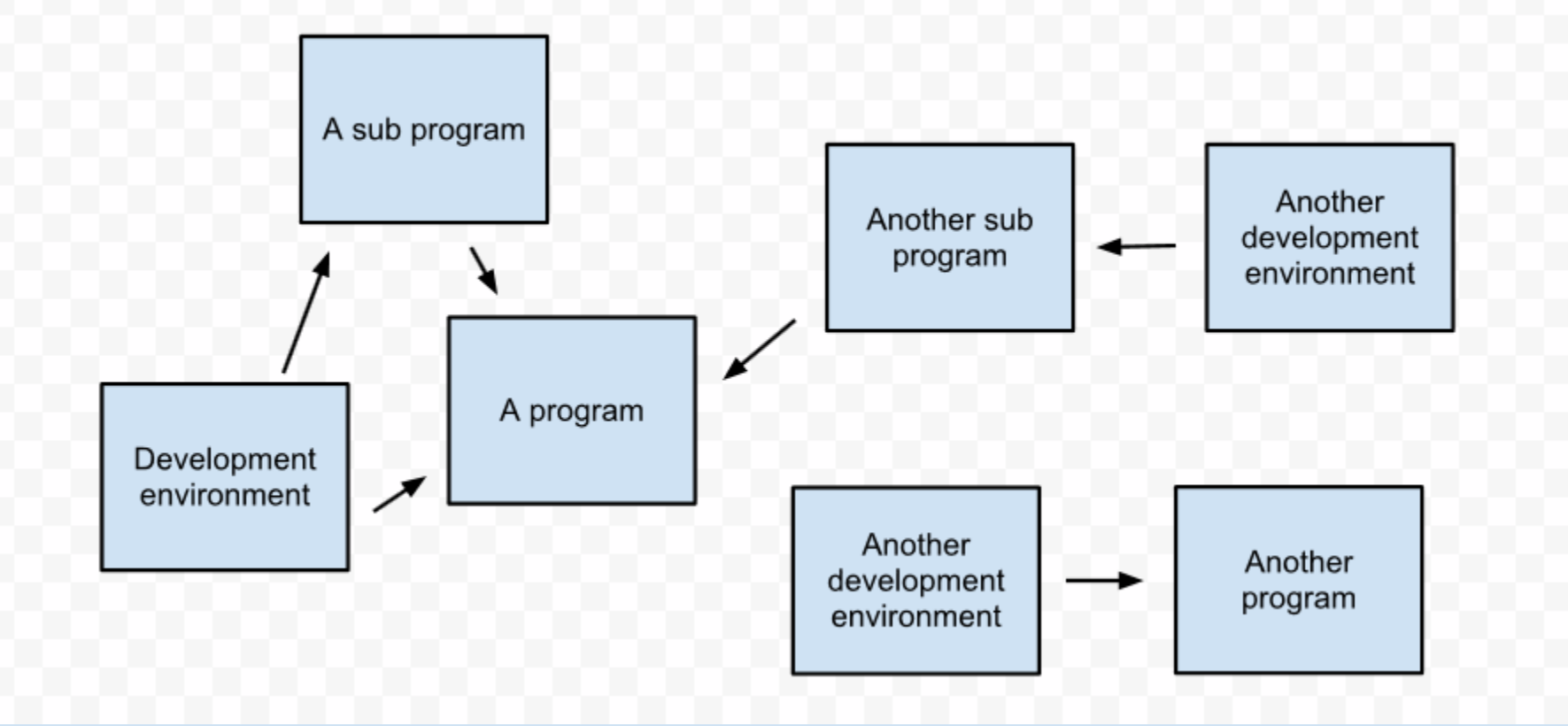
- Choosing base layers
  - OpenStreetMap
  - Bing aerials
  - MapBox?
- Refreshing data
- Playing with styling
- Seek better hosting



http://pdx.be/wpcint

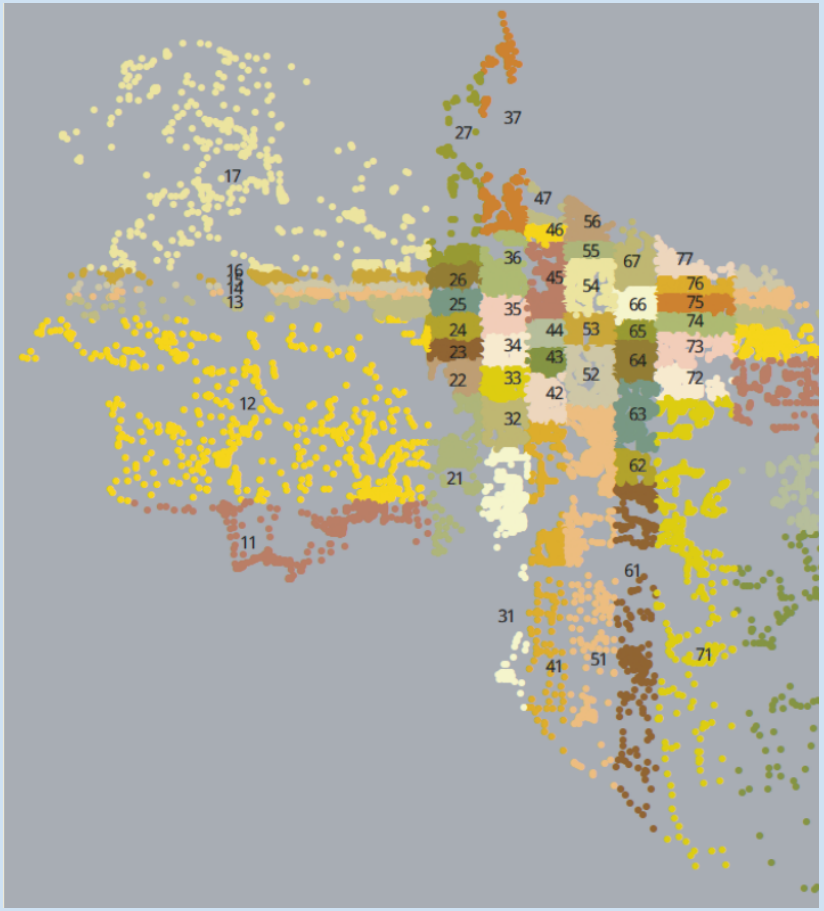
https://github.com/pdxmele/wpc-gis-inventory

QGIS programming environment demo



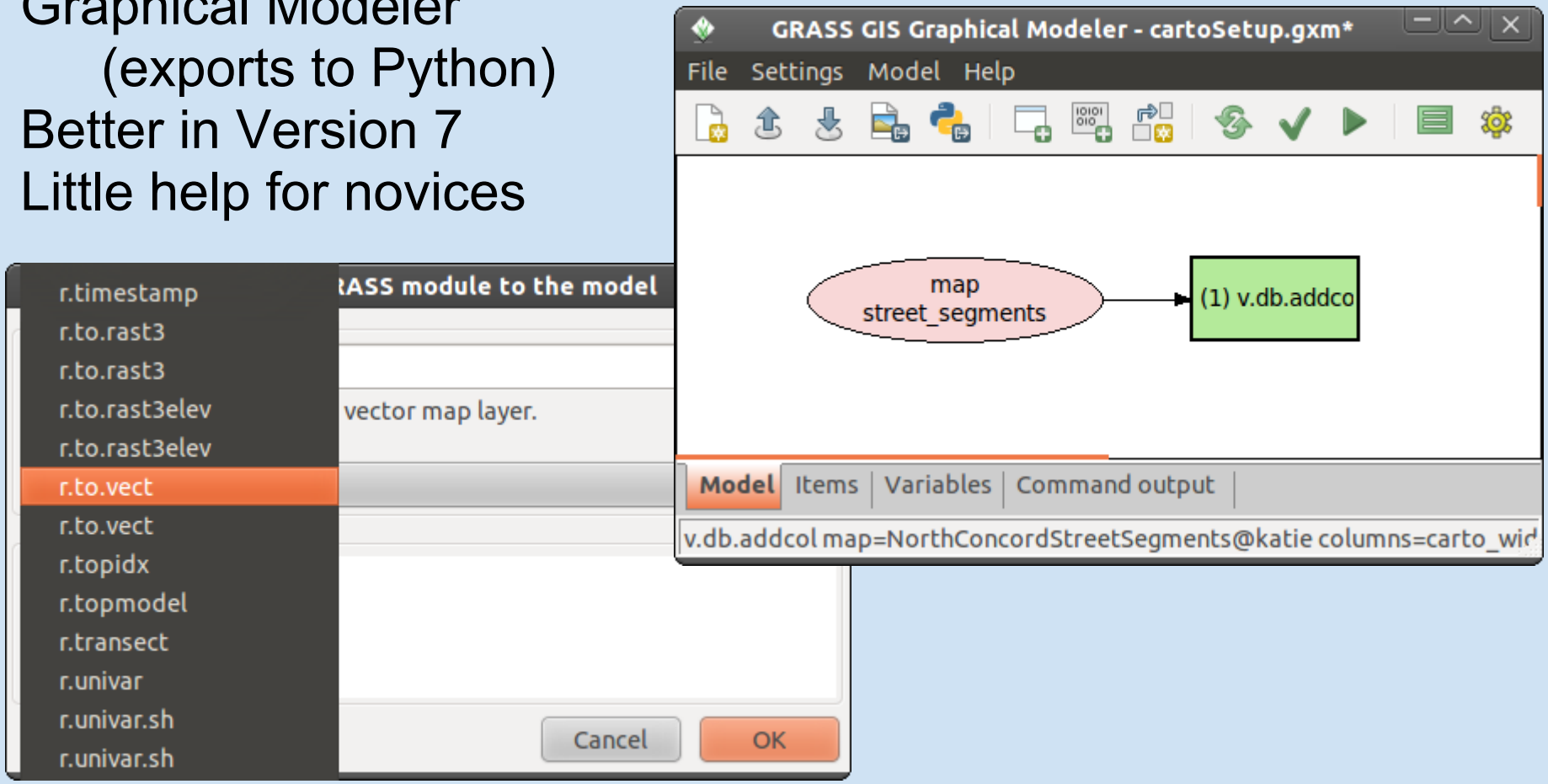
Jack's applications

- Creating and improving bicycling data
- Writing tools to assist in the process
  - Cracking closed systems
  - Coverage mapping with KML
  - Capture the flags - finding missed segments



GRASS GIS Graphical Modeler

Graphical Modeler
(exports to Python)
Better in Version 7
Little help for novices



Grass Python Scripting

Refer to the Programmer's Manual
http://grass.osgeo.org/programming6/index.htm
Test within Grass Environment

```
File  Edit  View  Search  Terminal  Help
>>>
>>>
>>>
>>>
>>> import grass.script as grass
>>> for map in grass.list_strings('vect'):
...     print map
...
Curb_Ramps@katie
street_segment_nodes@katie
Curb_ramps_concord@katie
street_segments@katie
NorthConcordStreetSegments@katie
streets_conflated@katie
concordextent@katie
streetsegments_concord@katie
network@katie
sw_clip@katie
pl_st_segments@katie
>>> quit()
GRASS 6.4.1 (OregonNorthNad83_2269):/home/Grass/Python >
```

Exploring open source programming

- Open source programming can be challenging at first, but help is available: come visit us at the Jam! We meet Mondays from 4-6 pm in CH469.

- Questions?