Right-size your PostGIS data

People
Postgres
Data

Ryan Lambert
ryan@rustprooflabs.com

RustProof Labs
bringing you data

```
SELECT *
    FROM pgconf.presenter
    WHERE id = 'ryanlambert'
```

- Owner / CEO
  @ RustProof Labs
- Publishes on blog.rustprooflabs.com
- DB Developer / Analyst
  @ Front Range CC

```
SELECT *
    FROM pgconf.presenter
    WHERE id = ‘ryanlambert’
```

- MySQL, mid-2000s
- MS SQL Server, 2009
- PostgreSQL, 2011
  - OpenStreetMap && PostGIS

Agenda

- Spatial data overview
- GIS tasks: Analysis vs. Thematic
- Simplification strategies
  - Polygons
  - Lines

GIS Data used…

© OpenStreetMap Contributors

Thank you!

https://www.openstreetmap.org/user/RustProof%20Labs

What is spatial data?

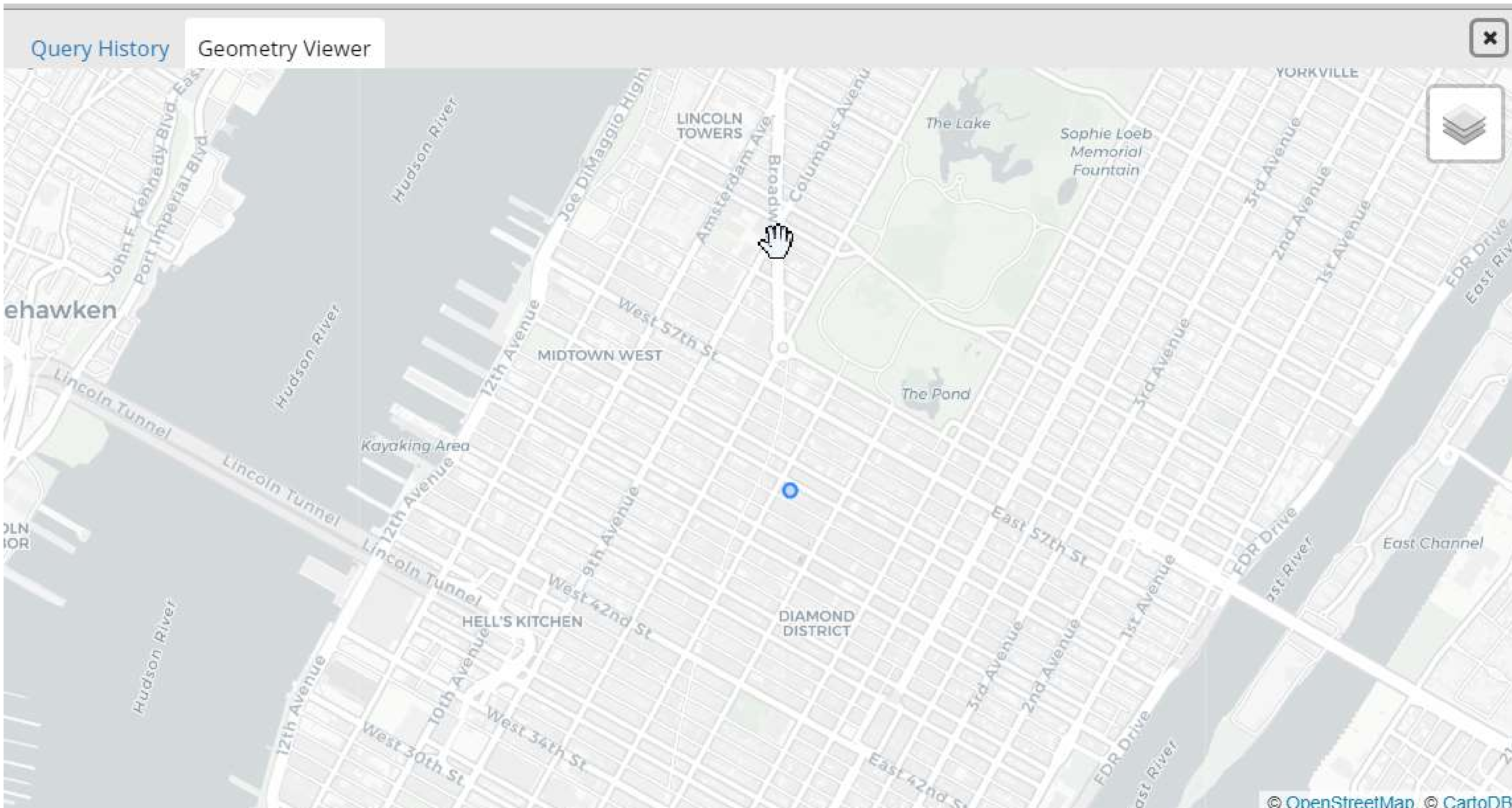
geom
3A10570514170BF31C504B166C1EA1B7B2815705141
8DCB2085241F783415A96C765C17E8720EFB2085241
209400E5341770EE6A1ADCB65C1375787F1420E5341
A84899C5141A81573169D7666C1661A8501BA9C5141
0544E0C5141C01B615068FE66C1EA23C7B25F0C5141
FE6804C5141F384521602ED66C1C5D28BF18B4C5141
BB0F802524108C29B9868B366C194D39821F0025241
F276FF651414BBCD7884C8066C1E53EE43070F65141
05DD43D514184A59918CA8366C18BFA52D1E73D5141
4F3289D514111F0D17DB4B365C1647DFCAA289D5141

What is spatial data?

- Data about the world around us.

- “GIS data is still just data!”
- … we have tools for that

```sql
SELECT ST_SetSRID(
 ST_Point(-73.9815, 40.7625), 4326);
```



Spatial data types

- POINT (Node)
- LINE
- POLYGON

Spatial data size

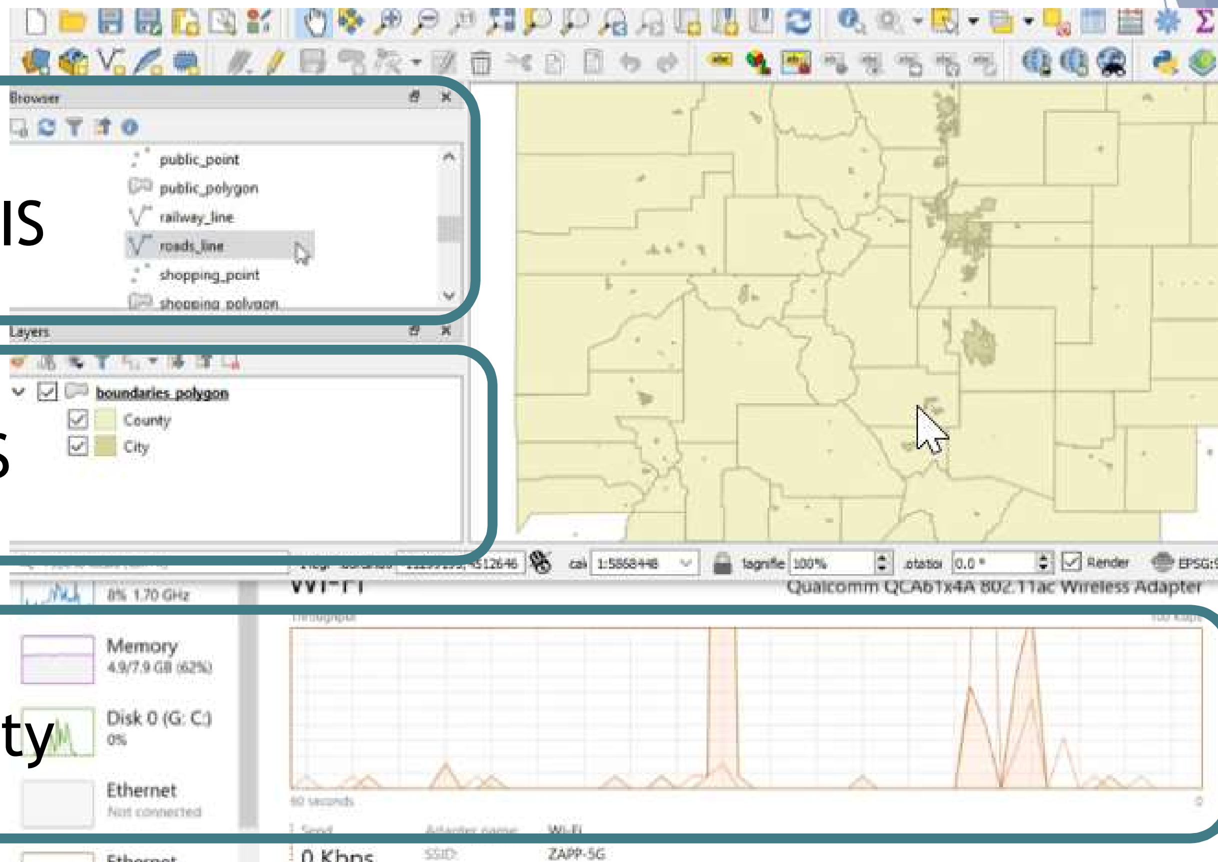
- Lines and Polygons can bloat quickly
- Similar to
  - JSON
  - BYTEA

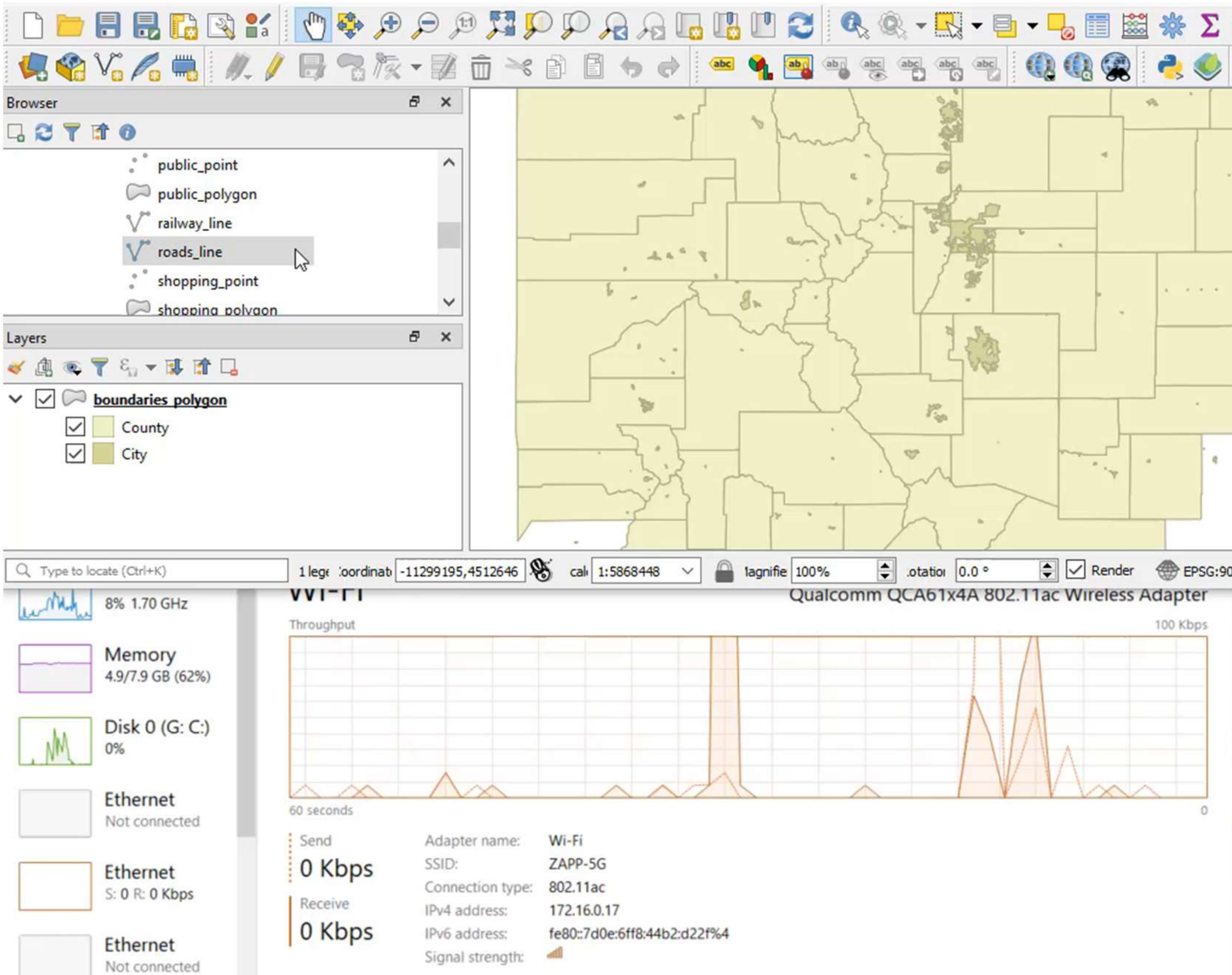
Puts on GIS analyst hat…

Tables in PostGIS

Layers in QGIS

Network Activity





Browser
public_point
public_polygon
railway_line
roads_line
shopping_point
shopping_polygon
Layers
boundaries_polygon
County
City
Type to locate (Ctrl+K)
-11299195,4512646
1:5868448
100%
0.0 °
Render
8% 1.70 GHz
Memory
4.9/7.9 GB (62%)
Disk 0 (G: C:)
0%
Ethernet
Not connected
Ethernet
S: 0 R: 0 Kbps
Ethernet
Not connected
Qualcomm QCA61x4A 802.11ac Wireless Adapter
Throughput
100 Kbps
60 seconds
0
Send
0 Kbps
Receive
0 Kbps
Adapter name: Wi-Fi
SSID: ZAPP-5G
Connection type: 802.11ac
IPv4 address: 172.16.0.17
IPv6 address: fe80::7d0e:6ff8:44b2:d22f%4
Signal strength:


THIS IS STUPID.

Extra challenge

- Feels slow
- DB doesn't always register performance issues

GIS Analyst Tasks

Analysis vs. Thematic

Using spatial data: Analysis

- I need coffee, quick, where's the nearest location?
- Distance from buildings to fire hydrants?
- Is my house in a flood plain?

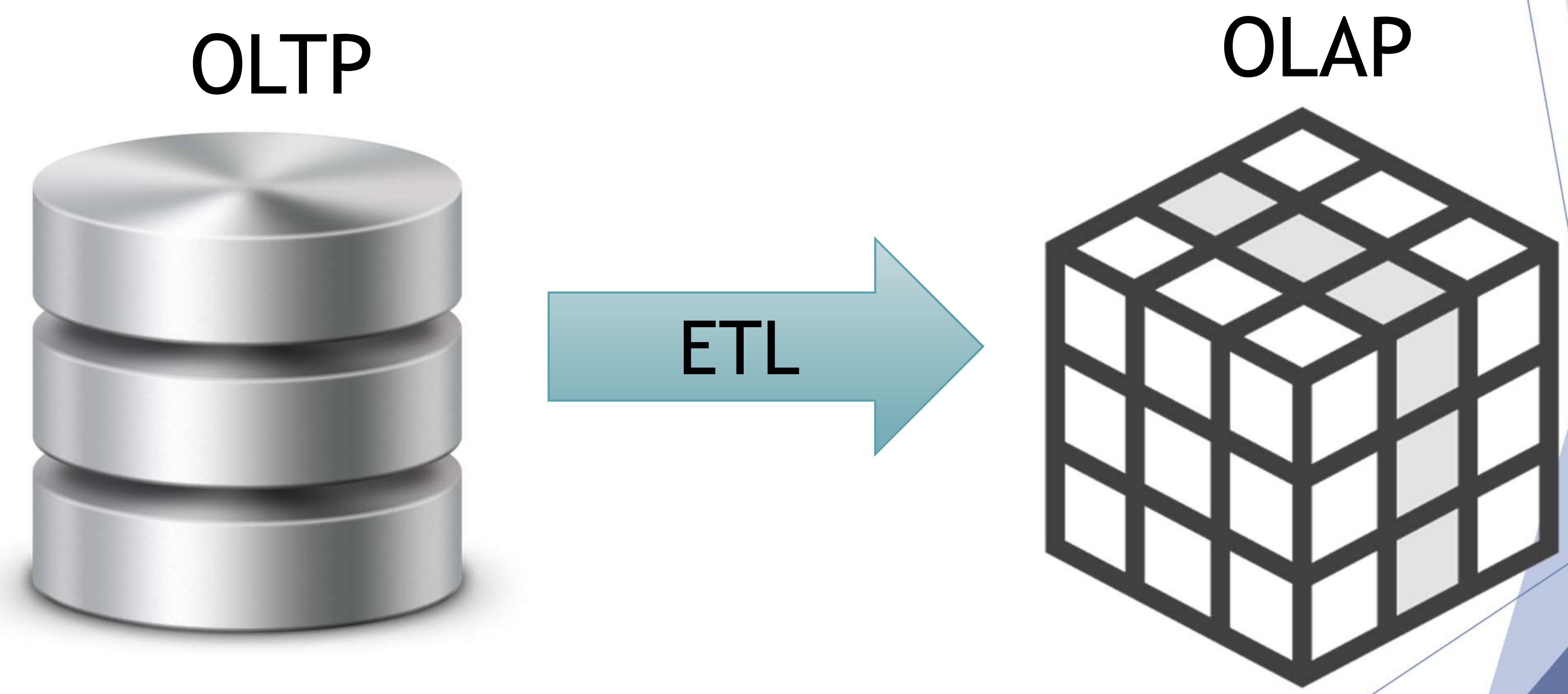
Using spatial data: Thematic

- Density of drivers around the northeast United States
- Regional crime rates
- Regional weather maps

Using Spatial Data: Thematic

- Visualize trends over an area

Transactional vs. Reporting



All data is dirty!

OpenStreetMap Data

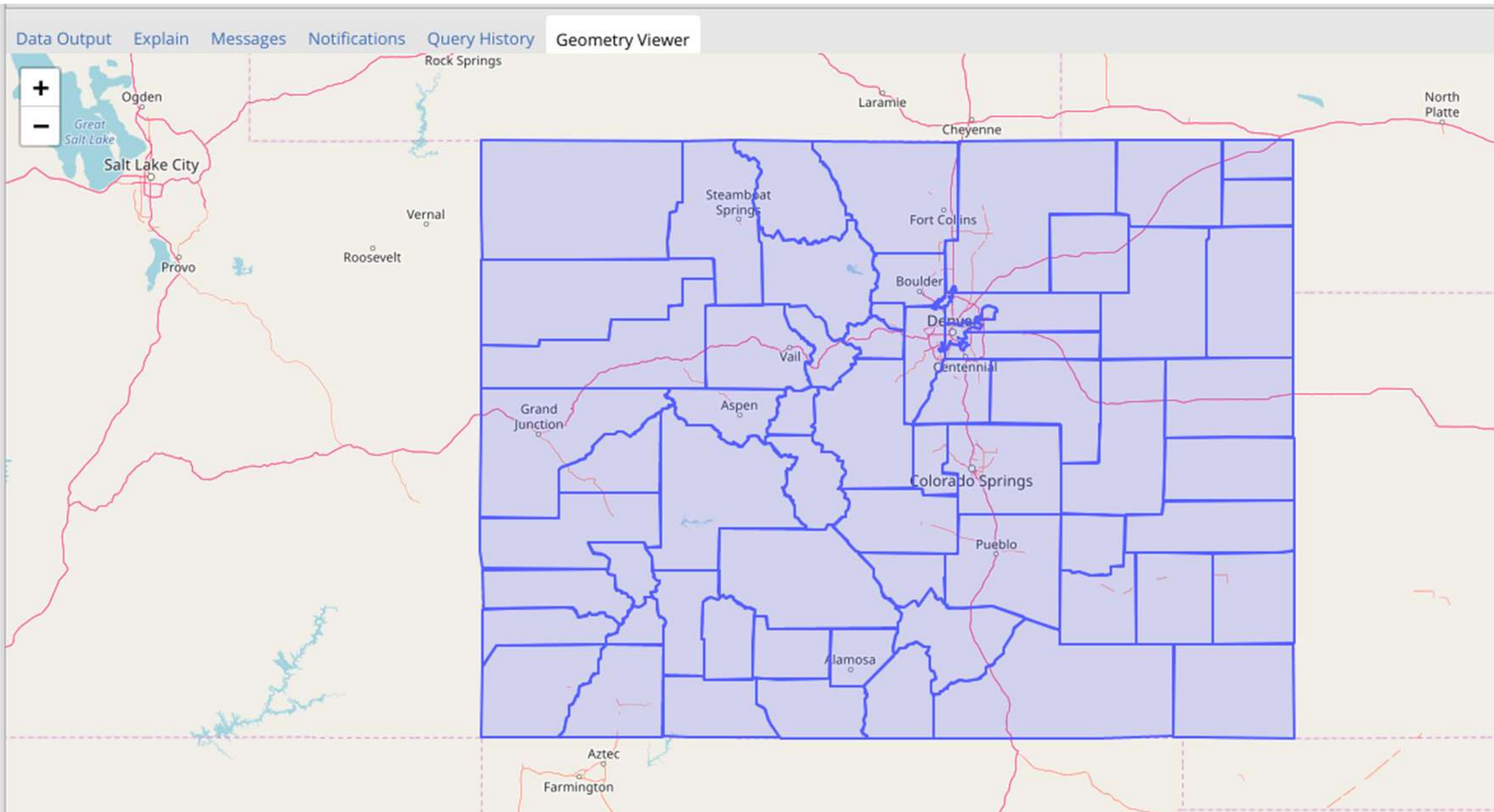
- Data entry
  - Some professionals
  - Other unpaid, untrained volunteers
  - Everything between
- Variable quality / formatting

Thematic GIS

Large-area polygons

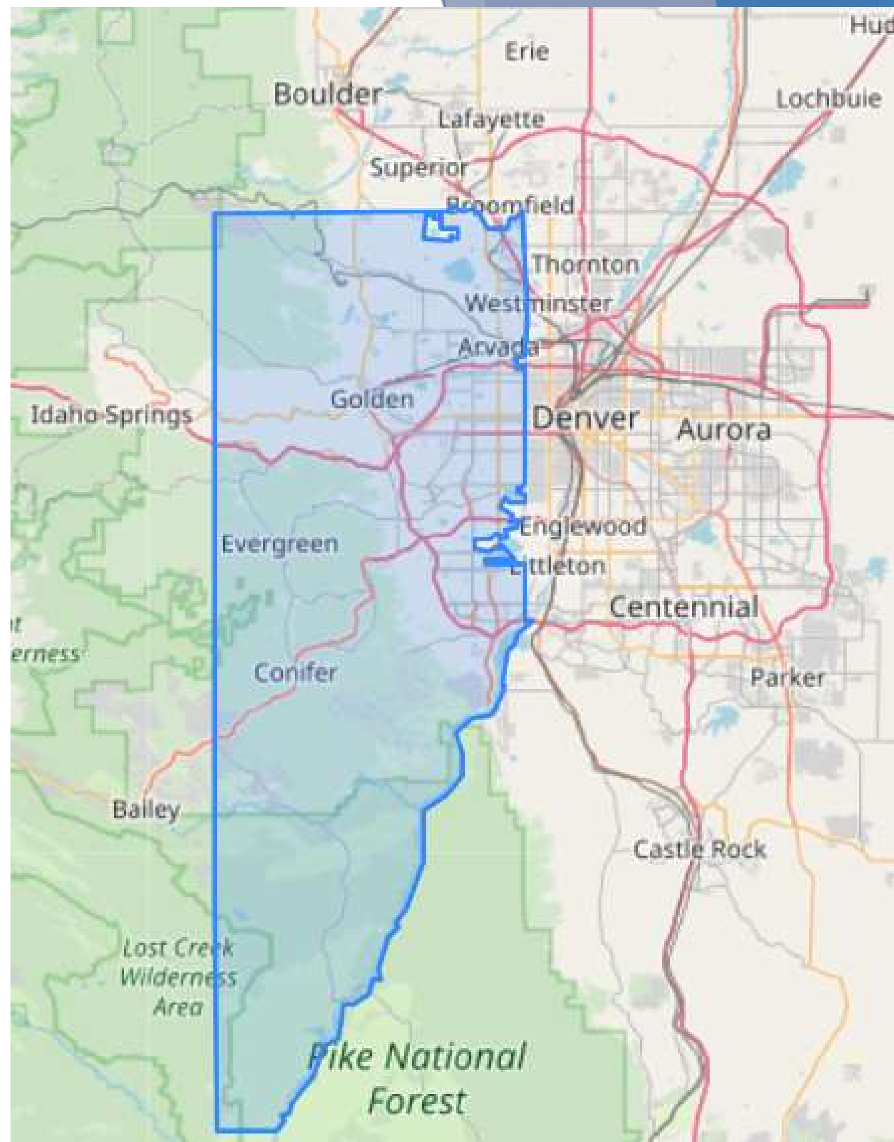
- Counties
- Zip Codes
- States
- Countries
- Lakes

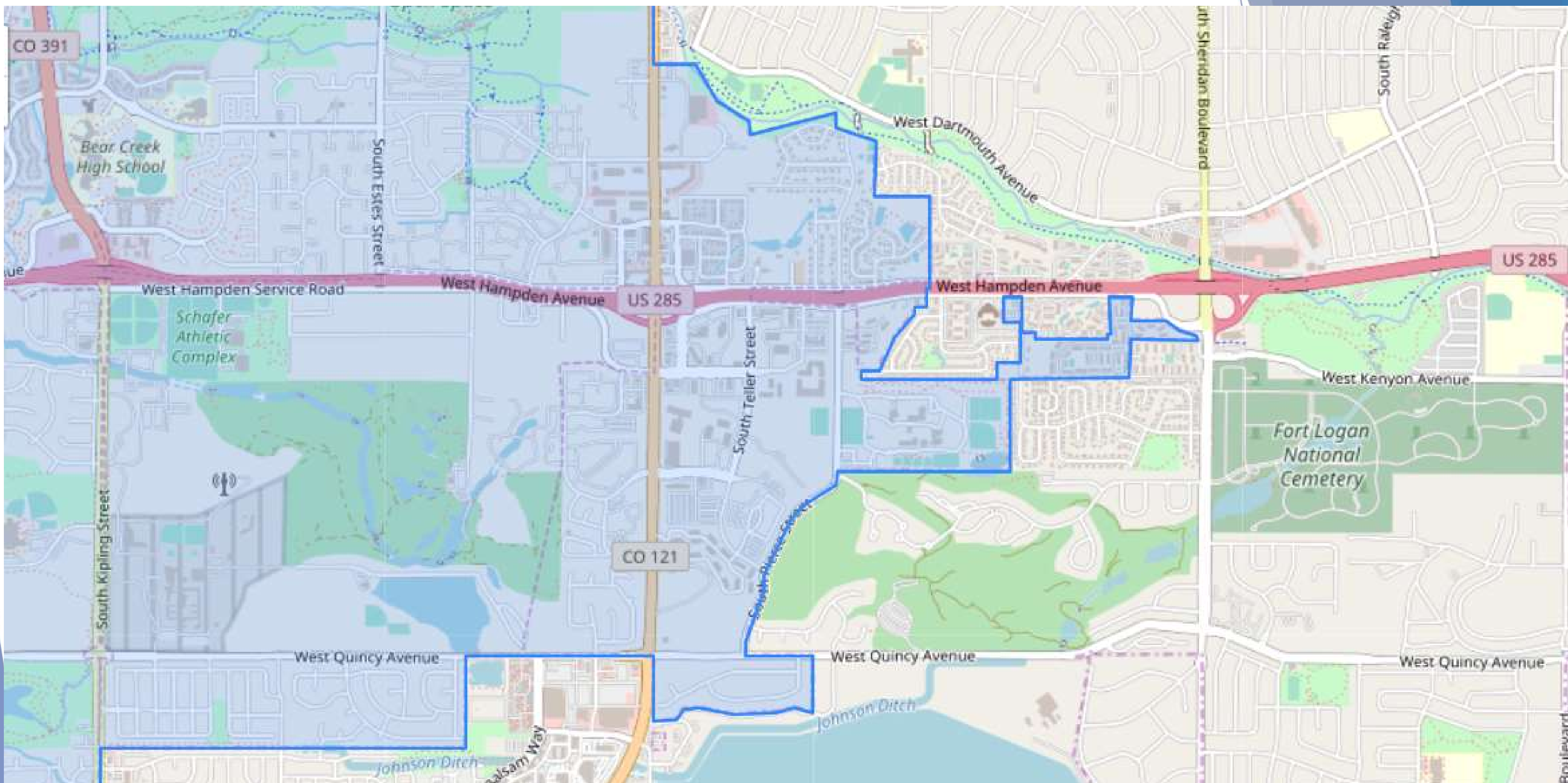
Counties of Colorado (64)



Large Polygon

- Jefferson County, CO
- ST_Npoints(way) = 780
- ST_MemSize(way) = 12.3 kb





CO 391
Bear Creek High School
South Estes Street
West Hampden Service Road
West Hampden Avenue
US 285
Schafer Athletic Complex
South Teller Street
West Dartmouth Avenue
South Sheridan Boulevard
South Raleigh
West Kenyon Avenue
Fort Logan National Cemetery
South Kipling Street
CO 121
South Pierce Street
West Quincy Avenue
Johnson Ditch
Balsam Way
Boulevard

Counties in the western US

```
SELECT COUNT(*) AS county_count,
    AVG(ST_NPoints(way)) AS points_avg,
    SUM(ST_NPoints(way)) AS points_total,
    AVG(ST_MemSize(way)) / 1024 AS kb_avg,
    SUM(ST_MemSize(way)) / 1024 AS kb_total
  FROM osm.boundary_polygon
  WHERE admin_level = '6'
```

Counties in the western US

- # of Counties: 460
- Total # of Nodes: 623k

Counties in the western US

- # of Counties: 460
- Total # of Nodes: 623k
- Average # of Nodes: 1,355
- Max # of Nodes: 13,832

Counties in the western US

- # of Counties: 460
- Total # of Nodes: 623k
- Average # of Nodes: 1,355
- Max # of Nodes: 13,832
- Average size per polygon: 21.2 kB
- Max size of polygon: 216 kB

Size Matters

- Average of 16 bytes / node

What can we do?

PostGIS: ST_Simplify()

- “Returns a "simplified" version of the given geometry…”

```
ST_Simplify(geometry, tolerance)
```

- Higher tolerance == More simplification

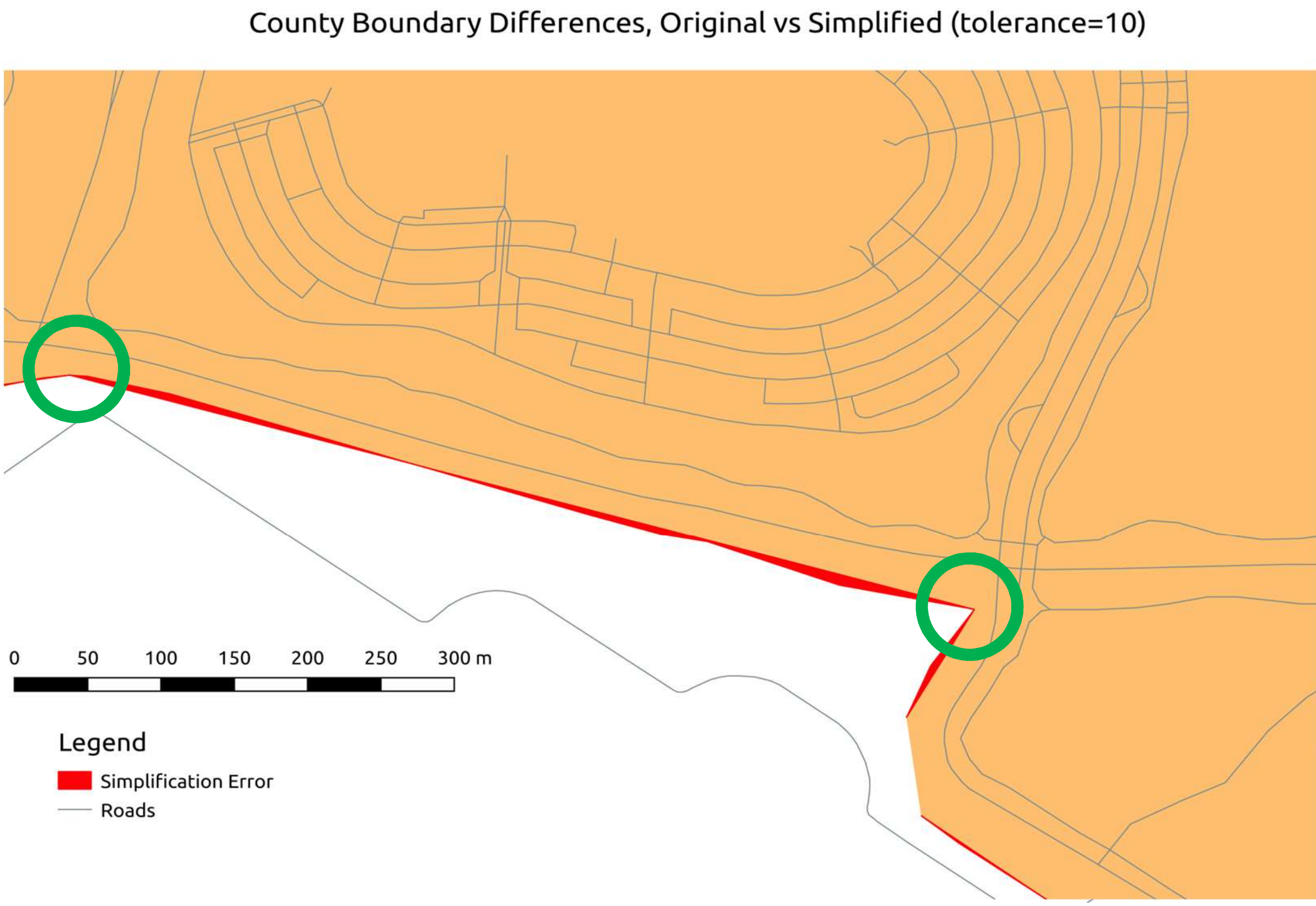
In non-spatial terms…

$$3.14159 \approx 3.14$$

PostGIS: ST_Simplify()

- SRID 900913
- tolerance=10

Polygon simplification



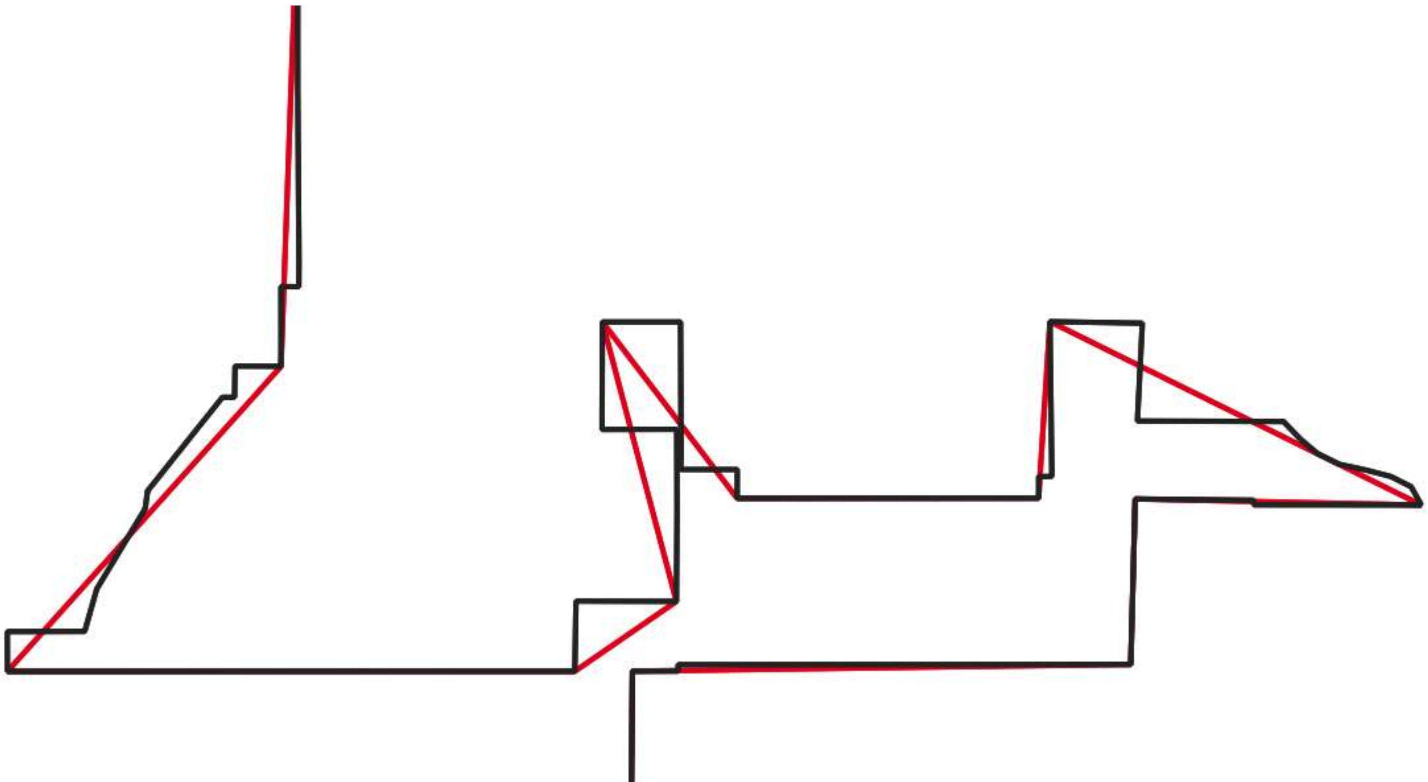
Benefits of Simplified Polygons

- Reduced # of nodes by 45-50%
- Reduced size on disk by 45-50%
- Improved query performance by ~ 40%

Side effects

- Reduced accuracy
- Potential errors
- Before/After
  - Error rate in testing: < 0.5%

Too much simplification?



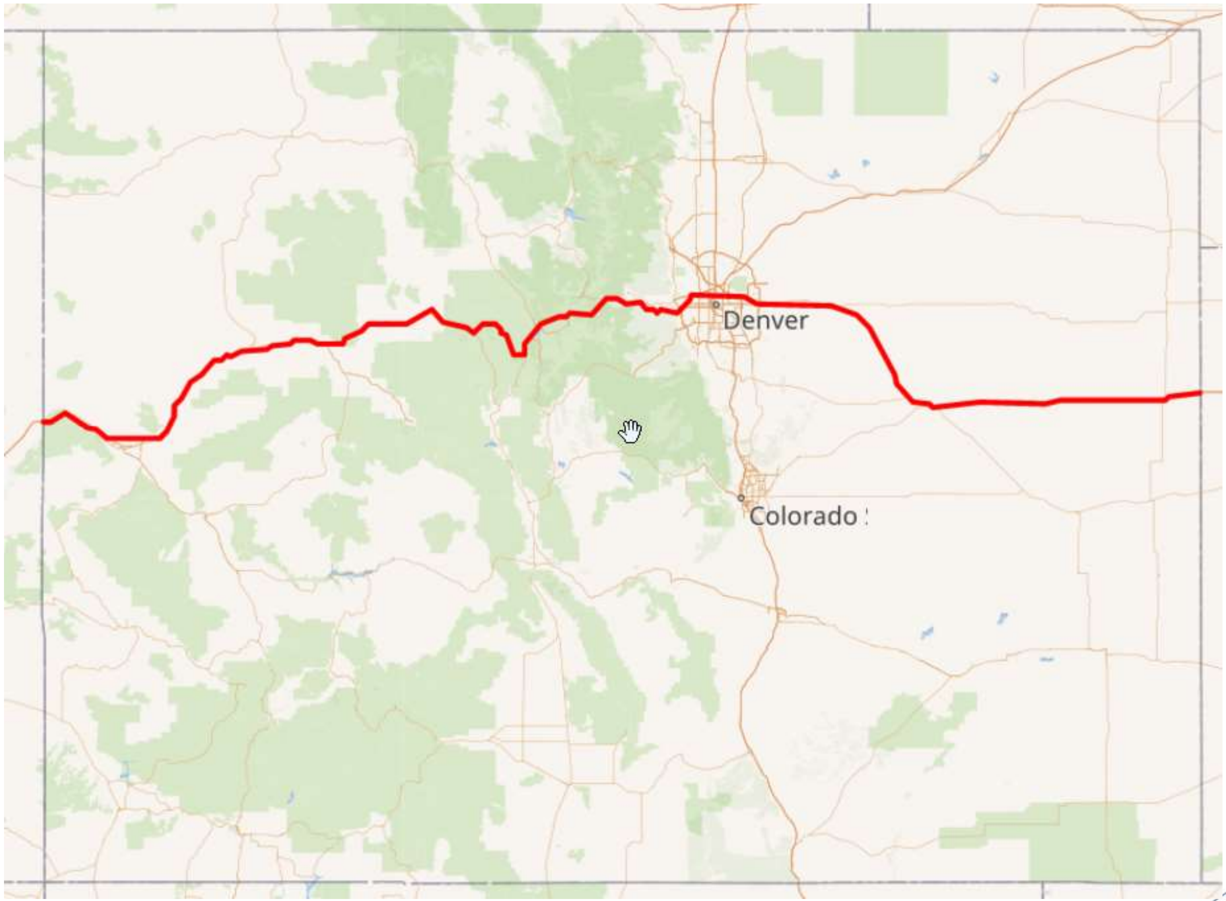
Questions so far?

Large number of small lines

Large number of small lines

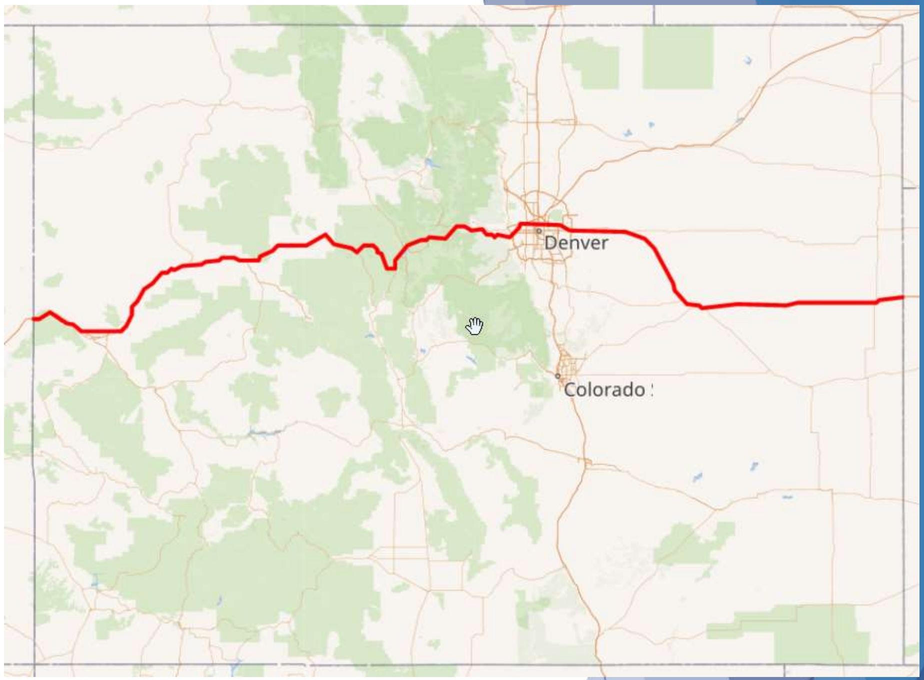
- OpenStreetMap roads
  - Interstate
  - Major highway
  - Minor highway
  - Residential roads
  - Sidewalks
  - Parking aisles
  - Hiking trails

I-70 in Colorado



I-70 in Colorado

- 450 miles
- 1,647 rows of data
- 19,281 nodes
  - MIN(ST_NPoints(way)): 2
  - MAX(ST_NPoints(way)): 217



Aggregate and Simplify

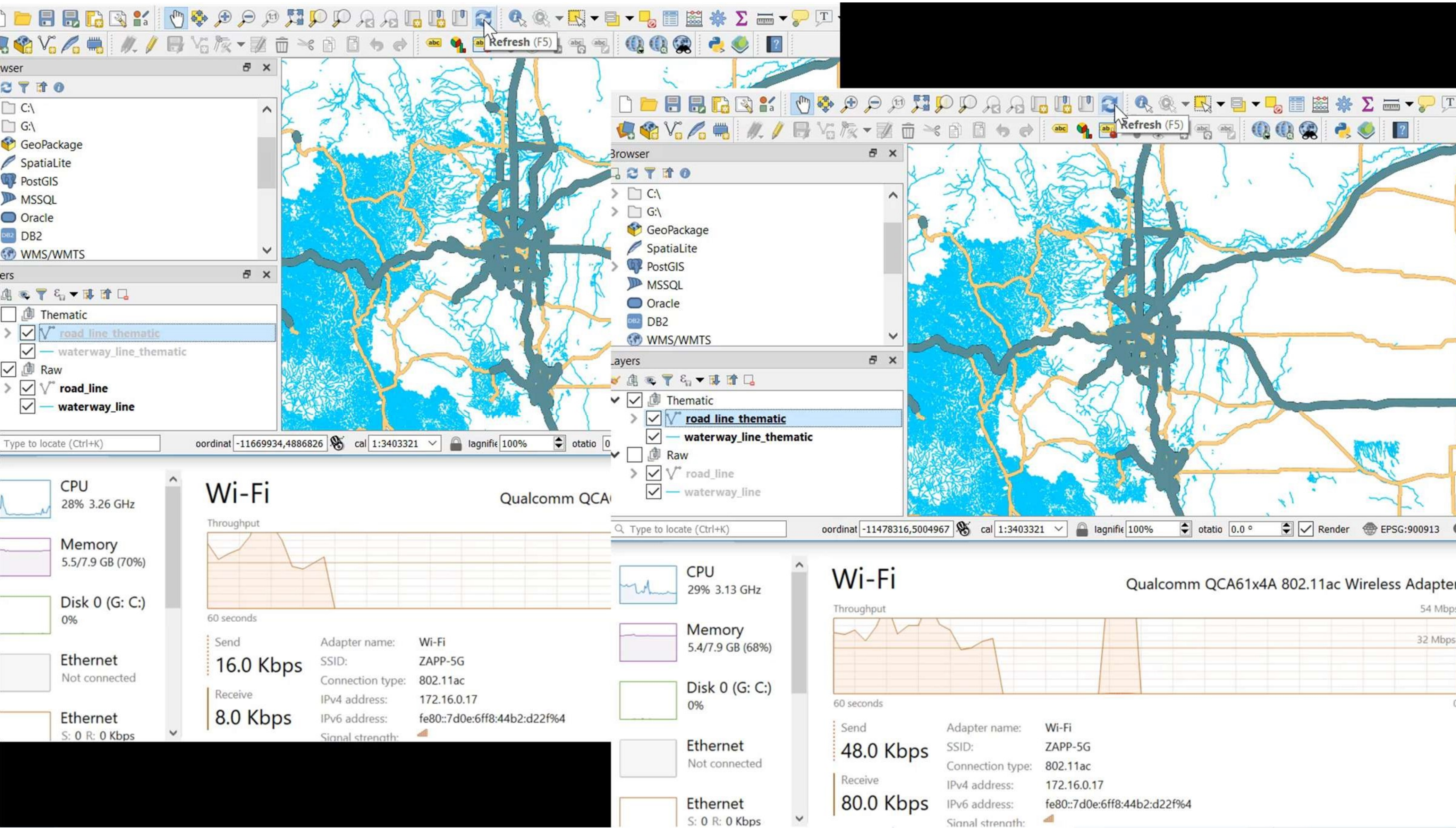
- ST_Collect()
- ST_Simplify()

ST_Collect()

- Aggregate function
- Think SUM() for spatial
- GROUP BY
  - ref
  - name
  - level
  - city

Demo (video)

- Render roads and waterways in QGIS
- Two windows
  - Raw data: Upper-left
  - Thematic: Lower-right


Refresh (F5)
Browser
C:\
G:\
GeoPackage
SpatiaLite
PostGIS
MSSQL
Oracle
DB2
WMS/WMTS
Layers
Thematic
road_line_thematic
waterway_line_thematic
Raw
road_line
waterway_line
Type to locate (Ctrl+K)
-11669934,4886826
1:3403321
100%
CPU
28% 3.26 GHz
Memory
5.5/7.9 GB (70%)
Disk 0 (G: C:)
0%
Ethernet
Not connected
Ethernet
S: 0 R: 0 Kbps
Wi-Fi
Throughput
60 seconds
Send
16.0 Kbps
Receive
8.0 Kbps
Adapter name: Wi-Fi
SSID: ZAPP-5G
Connection type: 802.11ac
IPv4 address: 172.16.0.17
IPv6 address: fe80::7d0e:6ff8:44b2:d22f%4
-11478316,5004967
1:3403321
100%
0.0 °
Render
EPSG:900913
CPU
29% 3.13 GHz
Memory
5.4/7.9 GB (68%)
Disk 0 (G: C:)
0%
Ethernet
Not connected
Ethernet
S: 0 R: 0 Kbps
Wi-Fi
Qualcomm QCA61x4A 802.11ac Wireless Adapter
Throughput
54 Mbps
32 Mbps
60 seconds
0
Send
48.0 Kbps
Receive
80.0 Kbps
Adapter name: Wi-Fi
SSID: ZAPP-5G
Connection type: 802.11ac
IPv4 address: 172.16.0.17
IPv6 address: fe80::7d0e:6ff8:44b2:d22f%4

PostGIS to QGIS Rendered

Source	Time (s)	Reduction in TTR
Original	50	0%
Original w/ Filter	8	-84%
ST_Collect() (table or mat. view)	2.5	-95%

https://blog.rustprooflabs.com/2018/12/postgis-tame-your-data-2

PostGIS to QGIS Rendered

- 40% faster query in Postgres
- 80-95% faster in QGIS

- QGIS pulls 2k rows at a time
- QGIS has to load, process, apply rules, and render

PostGIS to Rendered

- 80 - 95% improvement!

Faster in-DB spatial operations

Trees (Point) per county (Polygon)

```
EXPLAIN (ANALYZE, BUFFERS, COSTS)
SELECT  c.osm_id, c.name, c.way,
   COUNT(n.osm_id) AS trees
  FROM osm.county_polygon c
  INNER JOIN osm.natural_point n
    ON ST_Contains(c.way, n.way)
  WHERE n."natural" = 'tree’
    AND c.name = 'Jefferson County’
  GROUP BY c.osm_id, c.name, c.way
;
```

Trees per county

Raw

```
→ Sort (cost=968.13..968.35 rows=86 width=6,085) (actual time=6,098.885..8,969.786 rows=36,757 loops=1)
    Sort Key: c.osm_id, c.way
    Sort Method: external merge Disk: 280344kB
    Buffers: shared hit=1189, temp read=74553 written=74627
```

274 MB

Trees per county

Raw

```
→ Sort (cost=968.13..968.35 rows=86 width=6,085) (actual time=6,098.885..8,969.786 rows=36,757 loops=1)
    Sort Key: c.osm_id, c.way
    Sort Method: external merge Disk: 280344kB
    Buffers: shared hit=1189, temp read=74553 written=74627
```

274 MB

Thematic

```
→ Sort (cost=863.61..863.82 rows=86 width=6,536) (actual time=1,780.696..2,796.884 rows=36,757 loops=1
    Sort Key: c.osm_id, c.way
    Sort Method: external merge Disk: 137128kB
    Buffers: shared hit=1091, temp read=28609 written=28646
```

134 MB (-51% diff in spillage)

Trees per county

Raw

node type	count	sum of times	% of query
Bitmap Heap Scan	1	1,262.972 ms	12.1 %
Bitmap Index Scan	1	7.944 ms	0.1 %
GroupAggregate	1	1,446.367 ms	13.9 %
Nested Loop	1	5.078 ms	0.0 %
Seq Scan	1	0.276 ms	0.0 %
Sort	1	7,693.516 ms	73.9 %

Thematic

node type	count	sum of times	% of query
Bitmap Heap Scan	1	490.001 ms	14.7 %
Bitmap Index Scan	1	15.958 ms	0.5 %
GroupAggregate	1	528.265 ms	15.9 %
Nested Loop	1	4.762 ms	0.1 %
Seq Scan	1	0.063 ms	0.0 %
Sort	1	2,286.100 ms	68.8 %

Latencies at Human Scale

System Event	Actual Latency	Scaled Latency
One CPU cycle	0.4 ns	1 s
Level 1 cache access	0.9 ns	2 s
Level 2 cache access	2.8 ns	7 s
Level 3 cache access	28 ns	1 min
Main memory access (DDR DIMM)	~100 ns	4 min
SSD I/O	50–150 μs	1.5–4 days
Rotational disk I/O	1–10 ms	1–9 months
Internet call: San Francisco to New York City	65 ms[3]	5 years
Internet call: San Francisco to Hong Kong	141 ms3	11 years

https://www.prowesscorp.com/computer-latency-at-a-human-scale/

When to optimize?

- ETL
- Views / Materialized views
- Ad-hoc queries

ETL: PgOSM Project

- Started in 2015
- Transforms `osm2pgsql` structure to “Layers”
- MIT License
- https://github.com/rustprooflabs/pgosm

Final Thoughts

Postgres v11…

- Covering indexes!

Coming in Postgres v12

- Covering GIST indexes

```
CREATE INDEX gix_road_line
  ON osm.road_line
  USING GIST (way)
  INCLUDE (highway, ref);
```

- https://commitfest.postgresql.org/21/1615/
- https://github.com/postgres/postgres/commit/f2e403803fe6deb8cff59ea09dff42c6163b2110

Resources

PostGIS Docs

Chapter 8. PostGIS Reference

Table of Contents

https://postgis.net/docs/reference.html
https://postgis.net/workshops/postgis-intro/

RustProof Labs Blog

- PostGIS: Tame your spatial data (Part 1)
- PostGIS: Tame your spatial data (Part 2)
- Load OpenStreetMap data to PostGIS
- osm2pgsql on a Raspberry Pi
- PgOSM: Transform OpenStreetMap data in PostGIS
- PgOSM Transformations explained

Versions used

- SELECT version();
- PostgreSQL 11.1 (Ubuntu 11.1-1.pgdg16.04+1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 5.4.0-6ubuntu1~16.04.10) 5.4.0 20160609, 64-bit

- SELECT PostGIS_Full_version();
- POSTGIS="2.5.1 r17027" [EXTENSION] PGSQL="95" (procs need upgrade for use with "110") GEOS="3.5.0-CAPI-1.9.0 r4084" PROJ="Rel. 4.9.2, 08 September 2015" GDAL="GDAL 1.11.3, released 2015/09/16" LIBXML="2.9.3" LIBJSON="0.11.99" LIBPROTOBUF="1.2.1" RASTER

Thank you!

- Questions?