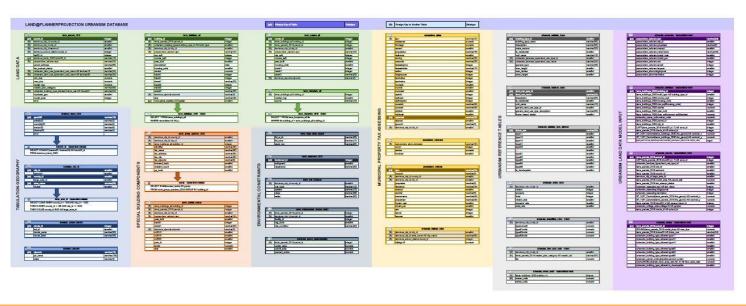
# PostGIS and GISquirrel and PostgreSQL...

Oh My!



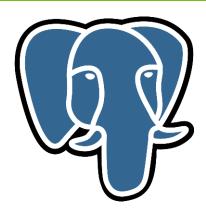
### The Why and How we got started

- 2050 Forecast data was scattered
- Needed better analysis and performance
- SQL Server not performing for us
- Reliable/low cost
- Centralize forecast data



#### What is?

- PostgreSQL
  - Open source
  - Similar to Microsoft SQL Server
  - Uses common data types (String, Integer, etc. )
  - Allows for powerful indexing GiST
  - Easy to extend into other tools like python
- PostGIS
  - Open source extension for PostgreSQL
  - Allows for spatial data to be stored and queried
  - Functions
  - Supports common spatial references we use
     NAD\_1983\_HARN\_StatePlane\_Michigan\_South\_FIPS\_2113\_Feet\_Intl
- GISquirrel
  - ArcMap add on
    - Use and edit layers from the database in a more relatable way.
  - Has free trial







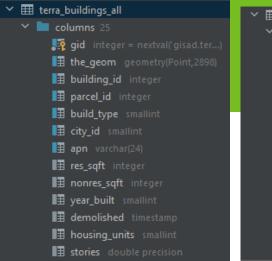
```
⇒SELECT p.*

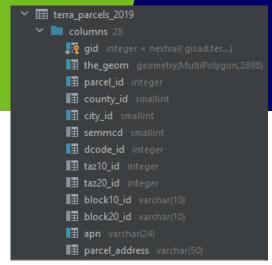
FROM terra_parcels_2019 p

join terra_buildings_2020 b

on p.parcel_id = b.parcel_id

⇒where b.build_type in (81, 82, 83, 84)
```

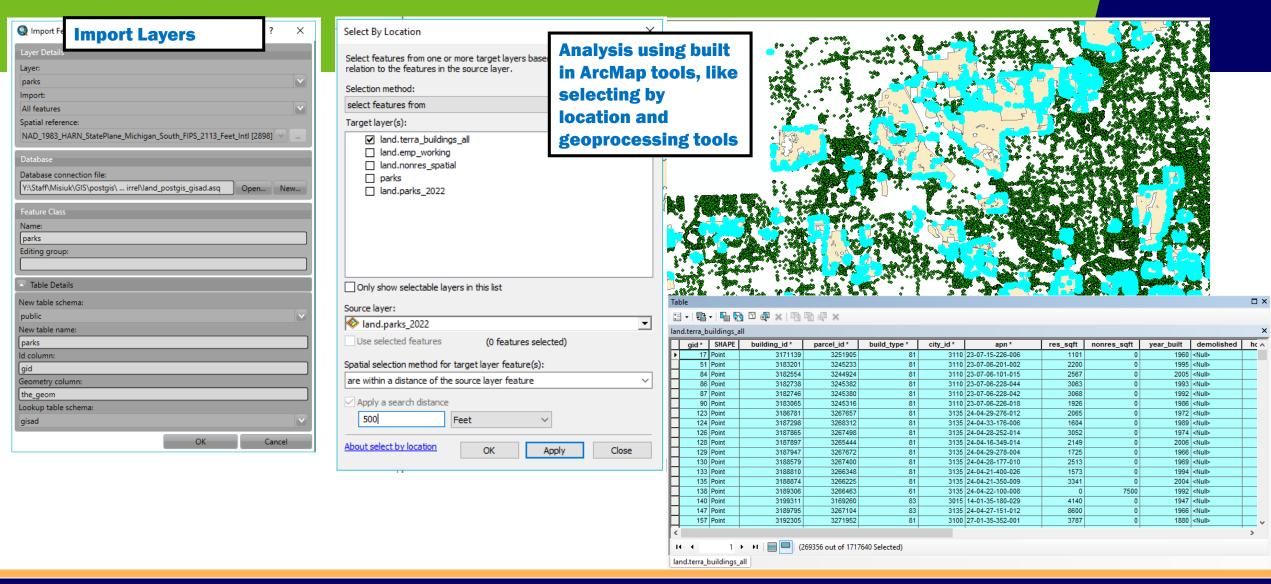




	驔 gid 🕏	I⊞ the_geom ÷	■ parcel_id ÷	<b>■</b> county_id ÷	<b>I</b> ≣ city_id ÷	■■ semmcd ÷	I≣ dcode_id ÷	∎≣ taz10_id ÷	∎≣ taz20_id ≎	■ block10_id :	■ block20_id	‡ <b>I∃</b> apn
1	1275502	0106000020520B0000010000000103000000010000000000000	1345553	163	1020	1020	82160	19106		5649001010	5649001003	71 142 99 0021
2	1820491	0106000020520B0000010000000103000000010000001500000090F4	1438527	163	1170	1170	82160	16311		5858001021	5858001007	80 023 99 0002
3	1820491	0106000020520B0000010000000103000000010000001500000090F4	1438527	163	1170	1170	82160	16311		5858001021	5858001007	80 023 99 0002
4	1820491	0106000020520B000001000000010300000010000001500000090F4	1438527	163	1170	1170	82160	16311		5858001021	5858001007	80 023 99 0002
5	1496668	0106000020520B000001000000010300000010000001200000010B4	1412741	163	1180	1180	81070	15007		5894002001	5894021001	81 071 99 0005
6	65282	0106000020520B000001000000010300000010000001D000000A0D1	5006374	115	5010	5010	58030	53202		8335003018	8335004017	02 102 002 15
7	65282	0106000020520B000001000000010300000010000001D000000A0D1	5006374	115	5010	5010	58030	53202		8335003018	8335004017	02 102 002 15
8	59753	0106000020520B000001000000010300000010000000500000048F9	5006395	115	5010	5010	58030	53202		8335003041	8335003013	02 102 027 00

#### create index on table USING GIST (the\_geom);







#### How do we use each?

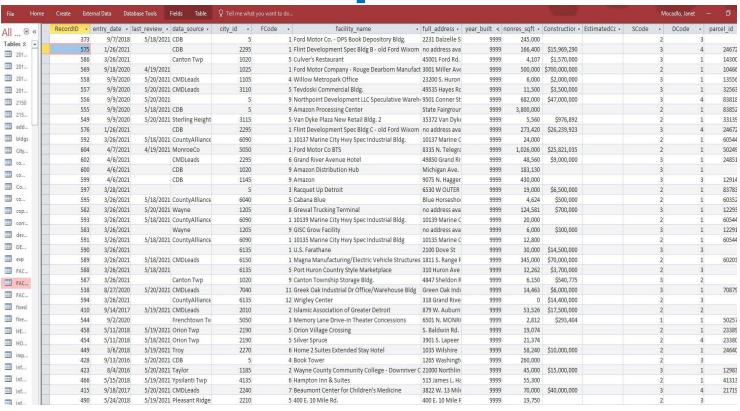
- Setup
  - Linux server
  - PostgreSQL v13
  - PostGIS v3
  - ArcMap 10.8.1
- Data Structure
  - One database with one schema
- Connections
- Use in Enterprise

#### **Demos**

- GISquirrel
  - Import data
  - Editing
- Postgres and PostGIS overview in PyCharm
- Publish query layer in ArcGIS Pro

#### Workflow Example - Nonresidential Development Data

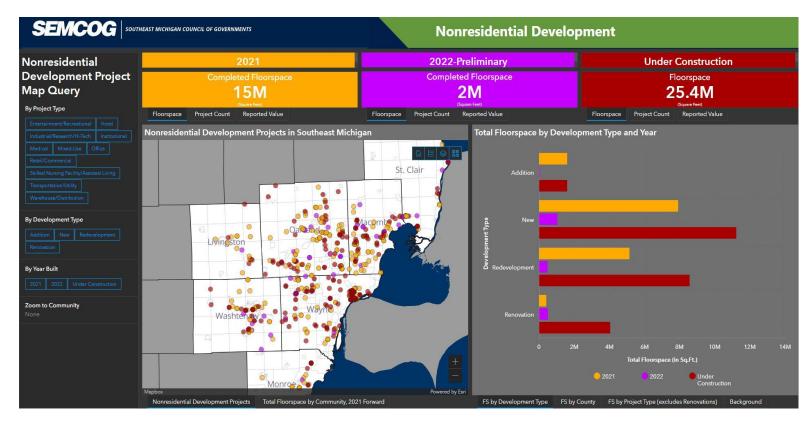
- Quick history
- Data originally housed in MS Access going back 20 years
- Annual development reports were always static pdfs
- Provided input to forecast using exports



#### Workflow Example - Nonresidential Development Data

#### "Perfect Timing"

- Desire to present data in an interactive, "real-time" way using ArcGIS Dashboards
- Forecast data now in PostgreSql. Made perfect sense to transition



### **Analysis**

- Query demo
- Undevelopable

```
SELECT r.gid, r.fename, AVG(b.year_built)
FROM terminus_roads_v17 r,
    terra_buildings_all b
WHERE st_intersects(st_buffer(r.the_geom, 1000),
b.the_geom)
GROUP BY r.gid, r.fename
```

#### **Extras**

- Ogr2ogr
- Pg\_tileserv

#### **Key Takeaway**

For very low cost, using these three tools in conjunction with ArcGIS Desktop and Enterprise, you can serve your internal and external stakeholders with a powerful back end that is <u>fast</u>, <u>flexible</u>, and <u>easy</u> to maintain.

#### **Q** and **A**

#### **Contact Information**



Chad Misiuk – GIS Coordinator SEMCOG 313-407-9996 misiuk@semcog.org



Janet Mocadio – Planner III SEMCOG 313-324-3434 mocadio@semcog.org