CREATING & USING A LIVE, LOCALIZED EXTRACT OF OPENSTREETMAP

OVERVIEW

- Introduction
- OSM Data
- Import Scenarios
- Keeping it Up to Date
- Using it
- Questions

STEAL THIS PRESENTATION!

OPENSTREETMAP DATA

WHAT IS OSM?

"The Wikipedia of Maps"

OSM is a **dataset**, which is...

- global
- open
- used to map nearly anything
- one of my favorite things

IS ANYONE ACTUALLY USING IT?

- Amazon Logistics
- Tesla Smart Summon feature
- Esri (basemaps, feature services)
- Pokemon Go
- Red Cross
- Kendall County!

OPENSTREETMAP DATA: *IT'S WEIRD!*

The OSM dataset has:

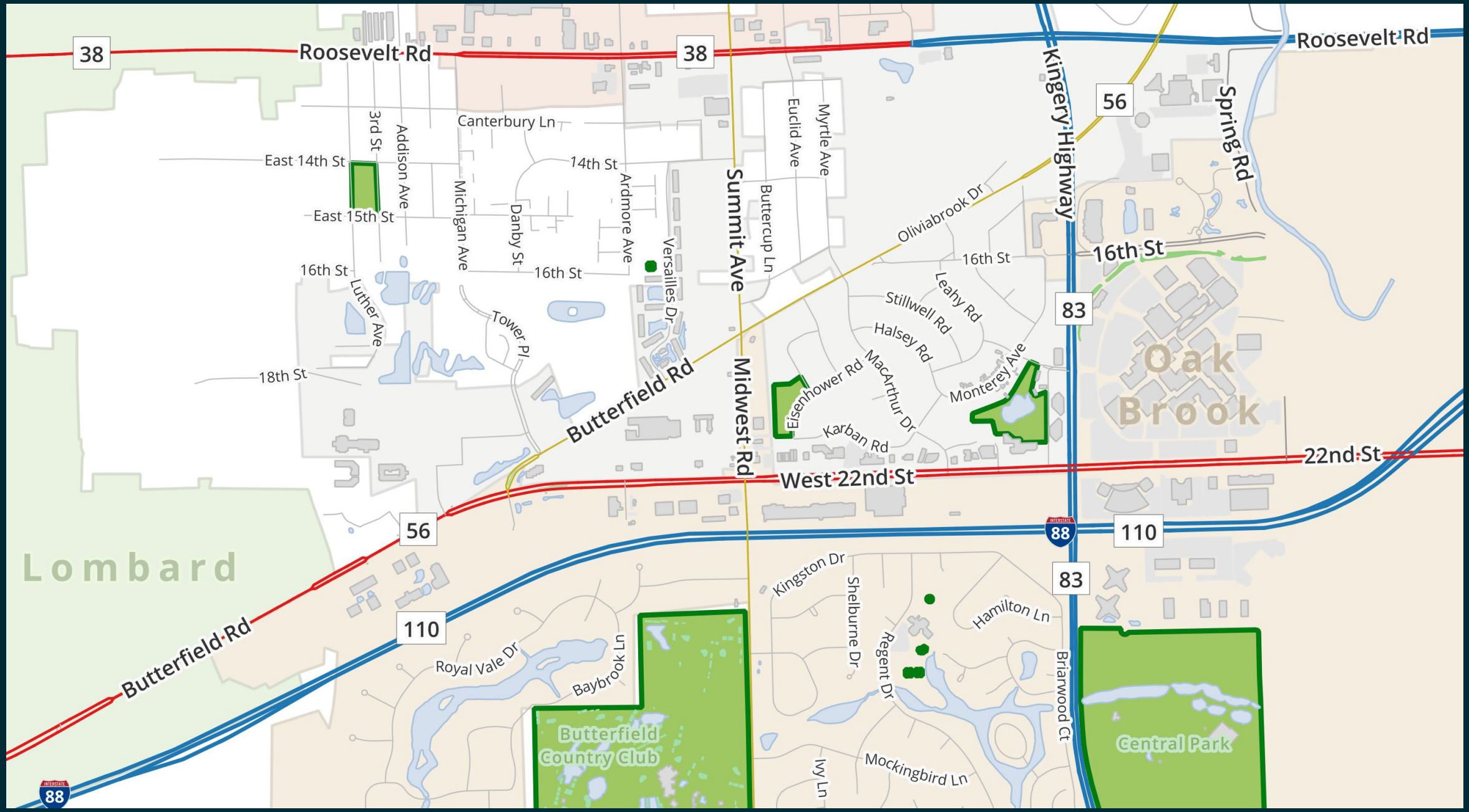
- NO layers
- NO schema
- NO polygons

To be more precise, OSM is an **object-oriented data model**.

Every element in the dataset can have an arbitrary number of key-value pairs

In OSM, we call those "tags".

```
{
  "elements": [
    {
      "type": "node",
      "id": 1,
      "tags": {
        "amenity": "bench"
      }
    },
    {
      "type": "way",
      "id" 2,
      "tags": {
        "natural": "water",
        "water": "lake",
```


38
Roosevelt Rd
38
Roosevelt Rd
Kingery Highway
56
Spring Rd
Canterbury Ln
3rd St
Addison Ave
East 14th St
14th St
Ardmore Ave
Euclid Ave
Myrtle Ave
Summit Ave
Buttercup Ln
Oliviabrook Dr
East 15th St
Michigan Ave
Danby St
16th St
16th St
16th St
Versailles Dr
Luther Ave
Tower Pl
Stillwell Rd
Leahy Rd
83
Halsey Rd
Eisenhower Rd
MacArthur Dr
Monterey Ave
18th St
Butterfield Rd
Midwest Rd
Karban Rd
Oak Brook
22nd St
West 22nd St
56
88
110
Lombard
Kingston Dr
Shelburne Dr
Hamilton Ln
83
110
Butterfield Rd
Royal Vale Dr
Baybrook Ln
Regent Dr
Briarwood Ct
Butterfield Country Club
Central Park
Mockingbird Ln
Ivy Ln
88

GETTING THE DATA

SOURCES

https://wiki.openstreetmap.org/wiki/Downloading_data

For very small areas, it may be possible to query the **Overpass API** directly.

State-based extract from GeoFabrik:
https://download.geofabrik.de/north-america/us/illinois.html

OSMIUM

https://osmcode.org/osmium-tool/

A "swiss army knife" for OSM data.

- **Extracting geographic subset**
- Getting metadata about objects
- Extract by attribute
- Generate / apply change files
- and more!

AUDIENCE PARTICIPATION!

http://geojson.io/#map=6.2/39.976/-89.164

```
osmium extract \
  -p city.geojson \
  ./data/illinois-latest.osm.pbf \
  -o ./data/city-extract.osm.pbf --overwrite \
  -s smart -S types=any
```

IMPORTING

imposm

Pros	Cons
✅ Easy to use and configure	❌ Relations are harder to work with, and require own tables
✅ Built in tag "cleaning"	❌ Lacks advanced geometry processing or configuration
✅ Table generalization	❌ Area of interest does not apply to relations
✅ Area of interest filtering for import *and* updating	❌ Development has plateaued

Mapping OSM data to tables is done using a YAML config file.

```yaml
tables:
  roads:
    columns:
    - name: osm_id
      type: id
    - name: the_geom
      type: geometry
    - name: type
      type: mapping_value
    - key: name
      name: name
      type: string
    filters:
      reject:
        area: ["yes"]
    mapping:
```

IMPORTING: SINGLE TABLE

```
imposm import \
  -config /app/imposm-scenarios/highways-config.json \
  -read /app/data/city-extract.osm.pbf \
  -deployproduction -optimize -write -overwritecache
```

```
{
    "cachedir": "/app/data/cache/highways",
    "mapping": "/app/imposm-scenarios/highways-mapping.yml",
    "connection": "postgis://gis:gis@database:5432/ilgisa2023?prefix=state_highw
}
```

ADDITIONAL SCENARIOS

MULTIPLE TABLES

```
imposm import \
  -config /app/imposm-scenarios/city-parks-config.json \
  -read /app/data/city-extract.osm.pbf \
  -deployproduction -optimize -write -overwritecache
```

IMPORTING *EVERYTHING*

And prepping for future updates!

```
1 imposm import \
2   -config /app/imposm-scenarios/city-all-config.json \
3   -read /app/data/city-extract.osm.pbf \
4   -deployproduction -optimize -write -overwritecache \
5   -diff
```

osm2pgsql

Pros	Cons
✅ Well established, continued development	❌ Updating requires a replication url, cannot easily be limited to an area of interest
✅ Extremely configurable	❌ Custom configuration can be harder to adjust / understand
✅ Allows mid-stream geometry operations	❌ Built-in generalization is still only experimental
✅ Exports to many coordinate systems	

Mapping to tables is done with a **Lua** file.

```lua
local nodes = osm2pgsql.define_table({
    name = 'osm2pgsql_nodes',
    ids = { type = 'any', id_column = 'osm_id' },
    columns = {
        { column = 'id', sql_type = 'serial', create_only = true },
        { column = 'tags', type = 'hstore' },
        { column = 'the_geom', type = 'point', not_null = true }
    }
})

function osm2pgsql.process_node(object)
    nodes:insert({
        tags = object.tags,
        the_geom = object:as_point()
    })
end
```

IMPORTING

POINTS OF INTEREST

```
1 osm2pgsql \
2   -j /app/data/city-extract.osm.pbf \
3   -d postgres://gis:gis@database:5432/ilgisa2023 \
4   -O flex -S /app/osm2pgsql-scenarios/pois.lua
```

EVERYTHING

```
osm2pgsql \
  -j /app/data/city-extract.osm.pbf \
  -d postgres://gis:gis@database:5432/ilgisa2023 \
  -O flex -S /app/osm2pgsql-scenarios/city-all.lua
```

UPDATING

```
imposm run -config /app/imposm-scenarios/city-all-config.json
```

Updating with `osm2pgsql` will only pull apply updates from the original download, i.e., all changes in the GeoFabrik IL extract, or all minutely changes for the entire planet.

USING IT

THOSE "EVERYTHING" TABLES

Rather than filter the data on import, it is possible to import all elements.

The tags can all be put into a single `jsonb` column. Attributes stored in the following format:

```
{"key_1":"value_1", "key_2":"value_2", ... "key_n":"value_n"}
```

Using the `column -> 'key'` syntax returns the value for the specified key.

```
1 SELECT
2   osm_id,
3   the_geom,
4   tags -> 'cuisine' as cuisine
```

WHY WOULD I WANT THAT?

- Schema remains as flexible as OSM itself
- No re-importing / re-indexing
- As use cases change, only the *queries* need to be modified, not the data
- Query output can be used identically to any SQL table
- QGIS has built-in support to parse *and edit* `jsonb` fields

SOME QUERIES

TRAFFIC ROADS

```
SELECT
  osm_id,
  the_geom,
  tags -> 'name' name,
  tags -> 'highway' class
FROM osm2pgsql_ways
WHERE tags -> 'highway' IN (
  'primary',
  'secondary',
  'tertiary',
  'motorway',
  'trunk',
  'unclassified',
  'residential',
  'service'
)
```

NATURAL AREAS / LAND COVER

```
SELECT
    osm_id,
    the_geom,
    COALESCE(tags -> 'natural', tags->'landuse') type,
    tags
FROM kendall_areas
WHERE COALESCE(tags -> 'natural', tags->'landuse') IS NOT NULL
```

WHAT TO DO

Honestly, once you write the query, you can use the results for **anything** that you would a normal table, including:

- Publishing a feature service to the web
- Generating vector tiles
- Using as input in geoprocessing
- Create an ArcGIS Locator dataset
- Create a routable graph network

QUESTIONS?