Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Lab 19: Advanced SQL

In this lab students will use SQL to perform some higher-order Geoprocessing functions.

## Part I: Connect to rpi schema in atlas\_rpi database

1. Open up QGIS and Go to Database🡪DB Manager🡪DB Manager
2. Click the carat (>) next to ‘atlas\_rpi’ in the tree to connect to the database
3. Expand the schema named ‘rpi’ and right-click a layer, e.g. ‘renscotowns’ layer and select ‘add to canvas’. You should see the layer appear in the QGIS map. Note: all tables in the rpi schema are in the UNM18N projection (26918).
4. Back in the DB Manager, click the ‘SQL Window’ button . This is how you will create your queries and load them as new layers into your map to check your work.

**ASSIGNMENT: Due Tuesday 4/17**

Working on your own or with others, create SQL statements that accomplish the tasks listed on the next page

**TIP:** There always needs to be a unique identifier in the query in order to Load as new layer to the map. If you query runs but the layer doesn’t load on the map, this is probably why. To help this avoid this problem, try these techniques:

1. **If a table has a unique ID, be sure to include it in your query.**
2. **always use SELECT DISTINCT instead of SELECT**
3. **If there is no unique identifier, you can always generate a new field using this code: ROW\_NUMBER() OVER() AS gid**

**Here are some functions you will likely need to use:**

**Geometry Constructors:**

ST\_Buffer(geom,distance in CRS units)

ST\_Intersection(a.geom,b.geom):

**Comparison Functions:**

ST\_Dwithin(a.geom,b.geom, distance in CRS units): find features from table a that are within a specified distance from table b

ST\_Intersects(a.geom,b.geom): determine whether features overlap

**Aggregate functions**—need to be accompanied by a GROUP BY statement

Dissolve geometries: ST\_Union(a.geom)

Count features: COUNT(b.geom)

**YOUR TASKS:**

Supply (e.g. cut/past) the SQL code used to complete each task

1. **Dissolve the Towns layer (renscotowns) by muni\_type** (city vs. town). You will need a GROUP BY statement at the end of the SQL so that it knows what to dissolve on.
2. **Find pollution sites within 100m of waterbodies**. Note: pollutant\_discharge\_sites may be within 100m of more than one waterbody. Avoid repeats with a **SELECT DISTINCT** statement.
3. **Count pollution sites by township**: Create a layer that has the gid, name and geom from the renscotowns layer, along with a count of how many pollutant\_discharge\_sites located within each town.
4. **Find Street-Flowline intersections**: find all the places where flowlines\_merged and renscostreets intersect. Include the stream name and street name along with the geometry. There will be no unique id for these so add a field that generates one on the fly with **SELECT ROW\_NUMBER() OVER() AS gid**
5. **Select all waterbodies that intersect Rensselaer County**. Use the waterbodies\_merged and renscotowns layers.
6. **Create 500m buffers around all waterbodies intersecting Rensselaer County.** Modify your code from #5 to create buffers instead of the original geometry.