Table of Contents

To run SQL queries in the *pgAdmin* interface, you use the *TOOLS* > *QUERY TOOL* option, type the SQL and press the lighting 'EXECUTE' button to run the query. Use these steps to run all the queries listed in Exercises 1, 2 and 3 below.

Keep the lecture slides handy when you're working through this practical – the queries you're asked to write are presented in the order we covered them in class.

Part 1 – Queries on Single Tables

For this exercise we will be using 3 tables that already exist in the PUBLIC schema in your database – public.london_counties (constituency boundaries for the London area), public.london_highway (the road network) and public.london_poi (points of interest)

NB: PostGreSQL is case sensitive, and if your column names or table names contain capital letters you will need to use quotation marks around them in your SQL. Some installations of PostGreSQL will force you to use lower case – if this is the case, you will need to adapt the queries below.

Note: In the following exercises your Column Names may be different – e.g. geom instead of the_geom, id instead of gid. Change the SQL as appropriate!

** SOME OF THESE QUERIES TAKE A WHILE TO RUN – BE PATIENT **

Part 1A - SIMPLE SELECT QUERY

1. Write a query to select all the rows and columns from the public.london_counties dataset. Note that we use the full name of the table here – i.e. *public*.london_counties. "p*ublic*" refers to the schema (or owner) of the dataset – i.e. the sub-area of the database where the table is stored. Schemas are used to allow you to keep datasets separate, perhaps for different projects.

(NB: this is a lot of complex geometry – see later in the module – so this query may be slow)

Part 1B - SELECTING INDIVIDUAL FIELDS

2. As you can see, there are a number of columns in the london_counties. Write a query to only select the name and file_name columns

Part 1C - WHERE CLAUSE

3. Generate a list of all the Points of Interest (public.london_poi) whose name is *Parking*. Remember the quote marks around *Parking* as it is a string.

4. List all the counties whose area is greater than 3000 hectares (you can use the hectares column

5. Find all the counties whose name begins with the letter B - use LIKE for this query.

6. Does it make a difference if you use a lowercase b in the above query?

7. Find all the counties that begin with the letter H and have an area of over 1500 hectares

Part 1D - COUNT(*) and WHERE

8. Find the number of roads in the highways dataset (public.london_highway).

9. As you can see, the column that was returned is called *COUNT*. As this is not a particularly useful name, we can use an *alias* to give the column a more reasonable name as follows (the new column name is enclosed in straight double quotes):

10. How many counties begin with the letter B? Write a query to work this out using:
    WHERE NAME like 'B%' and COUNT(*)

11. How many counties have the letter *a* in their name? Write a query to work this out using:
    like '%a%'

12. Write a query to find the number of roads from the *london_highway* dataset where the *TYPE* is a *path* or a *track*. (This query may take some time to run).

You will need to use the word OR in the WHERE clause, to list the types of road we want to match. This means that if the highway matches at least one of the road types, then it will be selected.

13. Adapt the above query to find *path* or *track* or *road*.

14. Now find the number of *secondary* roads (type is secondary) in the highway dataset which have a name associated (name is not null) with them and are also one way (oneway='true').

    In this case, instead of OR we need to use AND. In this case, the highway must match ALL the criteria listed, otherwise it won't be added to the selection. The phrase *is not null* finds any rows that actually contain a name. In other words, the query looks for all the secondary roads that are named and are also oneway.

15. Looking through the data you will find that ONEWAY is sometimes written as *true* and sometimes written as *yes.* Write the query that accounts for this – you will need to use OR and some brackets – i.e. (oneway = 'true' or oneway = 'yes')

16. Write a constraint to force all the oneway values to be 'true' or 'false'. What happens when you try to apply this rule retrospectively?

17. Write a query to find the population of all the counties dataset where the word *ham* is included in the locality name (*ham* is old English for village).

Part 1e - AGGREGATES, GROUP BY, ORDER BY AND DISTINCT

18. Write a query to find the maximum size county (size is stored in the hectares column)

19. Write a query to find the average size of the counties

20. Find the number of highways of each type, using a GROUP BY query

21. Create a GROUP BY query to find the number of points in each category in the points of interest data.

22. Write a query that lists all the different values in the oneway column in the highways table, and gives a count of the number of times each one occurs

23. Write a query that counts the number of different highway names and lists these in alphabetical order

24. Find the street name that occurs most commonly in the highways data, by sorting the data using a GROUP BY and ORDER BY query with the DESC option.

25. Find the different *names* associated with the Automotive category (category = 'Automotive') for the points of interest. Run this query without DISTINCT and with DISTINCT and compare the results.

Part 2 – Queries on Multiple Tables

Part 2a - SUB QUERIES

26. Find the highways that share a type with points of interest. First write the sub query

    SELECT DISTINCT type FROM public.london_poi;

Then nest it

    SELECT * from public.london_highway where type in (select distinct type from public.london_poi);

27. Nested queries and union: Find the details of the largest and smallest counties using a UNION ALL statement. You will first need to use a nested query to find which county has the max and min area.

Part 2b - SET QUERIES

28. Use a UNION query to create a list of all the bus-related infrastructure in LONDON_POI (name like '%bus%) and LONDON_HIGHWAY (type like '%bus%).

29. Use a union query to generate a complete list of all the names and geometries (i.e. the location of each feature) – this could then be used for a gazetteer for London

```
select name from public.london_highway
union all
select name from public.london_poi
union all
select name from public.london_counties;
```

30. Use group by and a nested query to list all the streets where the name only occurs once.

31. Write a query to find the ID of the largest county. Use this query as a sub query (nested query) to list all the highways in that county.

32. Make a list of the number of highways in each county

33. Use a DIFFERENCE query to create a list of all the highways that are not in a county with the letter b or B in its name (query might take a few minutes to run).

    select * from public.london_highway
    except
     (select * from public.london_highway where county_id not in
    (select id from public.london_counties where name like '%b%' or name like '%B%'));

34. Wrap another query around the DIFFERENCE query to find out how many highways there are in this list (i.e. use the DIFFERENCE query as a sub query)

35. Use an INTERSECT query to find out all the highways that are close to at least one POI names that are shared with highway names (if there are any).

    select name from public.london_highway
    intersect
    select name from public.london_poi;

Part 2c - JOINS

36. Use an INNER JOIN query to list all the names of highways in the counties whose name begins with R.

37. Use a LEFT JOIN to work out (list) if there are any highways not in a county. NB: As there are duplicate columns in both tables – id, name – then you need to start with something like:

    select a.name as highway_name, b.name as county_name, a.id as highway_id, b.id as county_id …

    Wrap that query to find a count of highways which don't have a county_id (i.e. where county_id is null)

NB: if you get the error: subquery in FROM must have an alias this means you need to assign a letter (alias) to your sub-query. You do this by putting an unused letter (e.g. c, d) to the right of the bracket that closes the sub query.

38. Use a FULL JOIN to list all the highways and their counties.

39. Wrap the FULL OUTER JOIN with a GROUP BY statement to work out how many highways are in each of the counties.