# Spatial Databases

Dr Claire Ellul
c.ellul@ucl.ac.uk

## Assignment Progress

- By now you should have:
  - Created your system specification
  - Created your conceptual and logical diagrams and written the documentation
  - Written the DDL, DML and the non-spatial queries
  - Made good progress on your 500 word assignment
- You will refine the above in the next 2 weeks to add spatial information (this and next week) and 3D (next week) after which you can complete the assignment

## Overview



- What is spatial data
  - Georeferencing
- Modelling spatial data in a database
- Storing spatial data in PostgreSQL/PostGIS
  - DDL - adding a spatial column
  - DML - inserting data
- Visualising the Data

## Spatial Data

- CEGE0052 is a *spatial* databases module
- In this case *spatial* refers to any data that can be located somewhere on the earth's surface (or above or below the surface)

** See Week 1 slides for more detail **

## Spatial Data

- The ability to create "maps" using spatial data can be found in:
  - Geographical Information Systems (GIS)
    - E.g. QGIS, ArcMap
  - Building Information Modelling (BIM)
    - E.g. Revit, Bentley Architecture, ArchiCAD
- In both cases, the maps can be 2D or 3D

## Spatial Data

- We will be using GIS for mapping during this module as GIS software currently works best with databases -
  - GIS are also extensively used in Asset Management
- … BIM software is slowly becoming better at working with databases but isn't quite there yet ..

# Everything Happens Somewhere

http://www.iotphils.com/wp-content/uploads/2014/07/Smart-Cities.png

# Spatial Data

- Used to understanding WHERE and WHEN things happen
  - And thus solve problems and provide useful services to people

http://www.directionsmag.com/images/newsletter/2011/01_week1/world_lg.jpg



# Spatial Data

- How do we model the world using spatial data?
  - Using four types of geometric representation
    - Point
    - Line
    - Polygon
    - Polyhedron (3D)
  - (Also other types of representations e.g. for continuous surfaces – not part of this module)

# Spatial Data – Points and Lines

- Spatial Types:
  - Points (also called nodes) are used for single point objects such as a well or a street light or traffic lights or – depending on the scale – a city or even a country.
    - Properties of a point include its location and its centroid (geometric centre)
  - Lines (also called polylines, arcs, edges) are used for interlinked objects such as rivers, water pipes or roads or for objects that appear linear from the air e.g. walls, fences, hedges.
    - Properties of a line include location, length, centroid and end-points.

# Spatial Data – Polygons and Polyhedra

- Spatial Types:
  - Polygons are used for objects having an actual area – e.g. buildings, parks, gardens. Polygons are also used for administrative boundaries (counties, city boundaries, school catchment areas, country boundaries).
    - Polygons have associated area, perimeter and centroid measures and are two-dimensional.
  - Polyhedra (or volumes) are three dimensional objects and provide the most realistic representation of real-world objects (e.g. buildings, geological rock strata).
    - They have associated measures of surface area and volume (the measure of the total enclosed space). Polyhedra are three-dimensional.

# Grouping Spatial Data

- Spatial data is grouped into themes (often known as layers)
  - A layer in a GIS is a way of collecting all the information relating to a particular object type type into one group.
    - E.g. – Rivers, Countries, Buildings, roads, rubbish bins, noise measurements
  - A layer can have any name you like
  - Layers are ‘stacked’ in the map to show all the data in one place
    - Maps are usually 2D but 3D is emerging (see later on in this module)

# Grouping Spatial Data

- For the purposes of this module:
  - a layer is a table in the database that has a spatial column
    - So the entities in your ERD become layers of spatial data if they are entities that can be mapped
  - see later on in this lecture for more information about making entities mappable using spatial columns

# Grouping Spatial Data

- In theory, in GIS you can name your entities anything you like and structure them how you like
- However, if you want to share data, you probably want to use a standard
  - Standards tell you exactly what to model
  - (For your assignment you should NOT use any standards – it needs to be your own work)



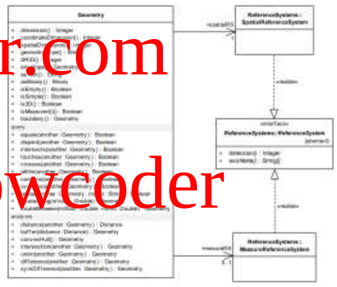
# Grouping Spatial Data

- Standards
  - An EU directive called INSPIRE has created 100s of standards for spatial data sharing
  - We may also see CityGML, for 3D data sharing, as part of the advanced topics work

INSPIRE 3D BUILDINGS

https://inspire.ec.europa.eu/data-model/approved/r4618-ir/html/index.htm?goto=2:2:1:3:7693

# OGC Standard – Simple Features



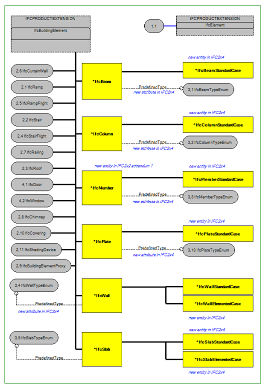
http://www.opengeospatial.org/standards/sfa

# Grouping Spatial Data - BIM

- In BIM objects are always represented in 3D
  - Or at least that is the aim of Level 2 and Level 3 BIM
- In BIM information is grouped by construction object type
  - E.g. concrete slab, window, door, wall, duct
- (For information only, not required for your assignment)

# Grouping Spatial Data - BIM

- In BIM the entity names are defined through a standard called Industry Foundation Classes



- (For information only, not required for your assignment)

## Spatial Data

- Most important thing for this module:
  - You can store spatial data in the database just like any other type of data
  - When you map the data, you don't only get the points/lines/polygons/polyhedral
    - YOU ALSO GET THE OTHER INFORMATION (attributes/columns) FOR THAT DATA

## Overview

- What is spatial data
  - **Georeferencing**
- Modelling spatial data in a database
- Storing spatial data in PostgreSQL/PostGIS
  - DDL - adding a spatial column
  - DML - inserting data
- Visualising the Data



## Spatial Data

- Spatial data includes *anything* that can be modelled using some form of *location* information!
  - i.e. where something is, referenced to a shared framework (could be a coordinate system, a map of London Boroughs, countries of the world, UK counties and many more)
- This referencing is called *geo-referencing*



## Geo-Referencing

- Can be direct:
  - E.g. a map that shows a building or another object, x/y coordinates, GPS coordinates
- Or indirect
  - For example, a Post Code or a Street Address is an indirect geo-reference that can be used to link non-spatial data to a position on the map. A PDF file containing the specification of a water pipe can be linked to the location of that pipe.
- See later on in the module for more details about georeferencing

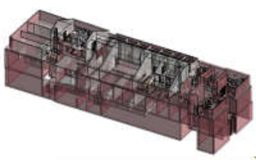


## Direct - Coordinate Systems

- Direct referencing works by mapping objects using their real coordinates (e.g. the coordinates that a GPS captures)
- Depending on where you are in the world, and what system you are using these coordinates may be referenced to different 'origin' points ..

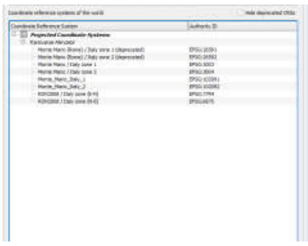
## Direct – Local Coordinate Systems

- Used in CAD/BIM
- Have a local reference point as the 0,0 point
  - Usually the edge of a construction site
- All distances and angles measured from this local reference point
- Also Cartesian (flat surface)



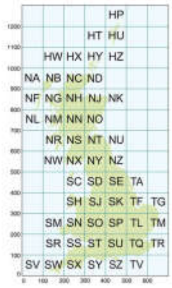
## Direct – National Coordinate Systems

- National coordinate systems are usually created by an organisation such as a National Mapping Agency
- Some countries have more than one



## Direct – National Coordinate Systems

- In Great Britain, our mapping system uses “British National Grid” which has its 0,0 of the south west coast
- Cartesian system




https://timetrail.warwickshire.gov.uk/assets/exhibition/Image/Toolkit/gb-map.jpg



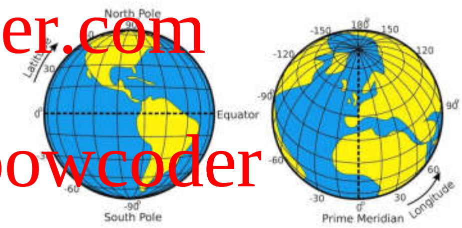
## Direct – Global Coordinate Systems

- As satellite systems such as GPS don’t only map Great Britain, they use a reference system that covers the world
  - Coordinates are latitude/longitude
    - Longitude ranges from 0 at the Prime Meridian passing through Greenwich, England, to +180 toward the east and 0 to -180 toward the west.
    - Latitude ranges from 0 at the equator to +90 at the North Pole and 0 to -90 at the South Pole. For example, Denver's position shown in the figure is -104.9 degrees longitude (west) and +39.8 degrees latitude (north).

http://www.innovativegis.com/basis/pfprimer/topic7/topic7.html



## Direct – Global Coordinate Systems




https://thumbs-prod.si-cdn.com/0rQSHAWkucV1O0dmyJVN8Ml0sS4=/800x600/filters:no_upscale()/https://public-media.smithsonianmag.com/filer/5c/ea/5cea567c-050b-432a-834f-fc94dcb1b49e/coordinates.jpg



## Coordinate Reference Systems – Standard Codes

- Local coordinate reference systems are not set by any authority but are just defined by whoever is working on a project
- However, national and international systems are public and are assigned a code by the European Petroleum Standards Group
  - This is called an EPSG code
- In the UK
  - EPSG 27700 – British National Grid
  - EPSG 4326 – the WGS84 system used by GPS

## Coordinate Reference Systems – Standard Codes

## Coordinate Reference Systems – Linking Local and National Data

- If you have a locally referenced dataset, you can transform the data to a national reference system
- At a very basic level, if you know the real world x/y of one point (e.g. a corner of the building) in national units, then you can use this to shift all the coordinates
  - Tools such as Revit (for BIM) allow you to do this
  - Might also need to change the units from mm to m
- More sophisticated methods also exist (the geospatial students might learn some of these over the coming year)

## Overview

- What is spatial data
  - Georeferencing
- **Modelling spatial data in a database**
- Storing spatial data in PostgreSQL/PostGIS
  - DDL - adding a spatial column
  - DML - inserting data
- Visualising the Data



## Databases – Storing Spatial Data

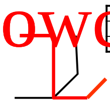
- Data Types
  - So far, we have seen the following data types in our databases:
    - VARCHAR
    - NUMBER (m, n)
    - DATE
  - Most modern Databases also have a special data type for storing vector spatial data
  - In Oracle, this data type is called SDO_GEOMETRY
  - SDO stands for Spatial Data Object
  - In PostGIS this is a GeometryColumn

## Spatial Data – Basic Representation

- Points represented as two coordinates – x,y

| ID | X | Y | Attribute |
|---|---|---|---|
| 1 | 100 | **100** | **Well1** |
| 2 | 400 | **400** | **Well2** |

- Lines are represented as a list of coordinate pairs

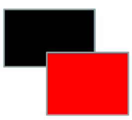


| ID | X1 | Y1 | X2 | Y2 | X3 | Y3 | x4 | Y4 | X5 | Y5 | Attribute |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 125 | [illegible] | [illegible] | [illegible] | 450 | 375 | 450 | 500 | | | **Road 1** |
| [illegible] | [illegible] | [illegible] | [illegible] | [illegible] | 335 | 150 | 500 | 150 | 600 | 250 | **Road 2** |

How do you model lines with more than 5 nodes?

## Spatial Data – Basic Representation

- Polygons are also represented as a series of x, y points – but the first and last point must be the same to close the loop

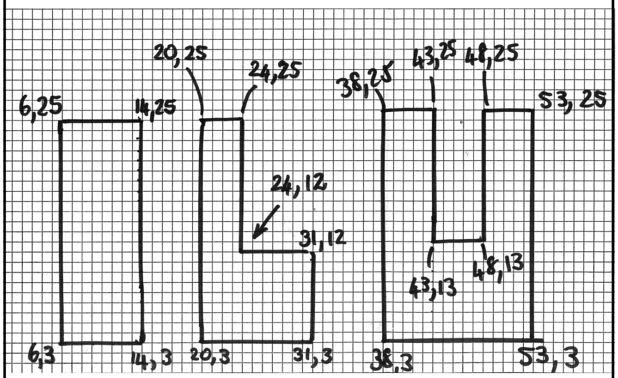


| ID | X1 | Y1 | X2 | Y2 | X3 | Y3 | x4 | Y4 | X5 | Y5 | Attribute |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 125 | 150 | 300 | 150 | 300 | 450 | 125 | 450 | 125 | 150 | **House 1** |
| 2 | 225 | 50 | 450 | 50 | 450 | 200 | 225 | 200 | 225 | 50 | **House 2** |

How do you model polygons with more than 4 nodes (coordinate pairs)?

## Exercise

- Draw the table that you would need to store these polygons in a database




## Modelling more complex objects

- Option 1
  - Keep adding columns
    - But you could have 1000s of nodes!
    - Also could have lots of empty space
- Option 2
  - Use an ‘object relational’ approach – i.e. create a primitive type to store all the required information – this is called a ‘geometry’ data type

## Modelling more complex objects



- Object-Relational Approach
  - Geometry is just another column in the table

| Attribute | Attribute | Attribute | Geometry |
|---|---|---|---|
| | | | << all the geometry information stored in one column as a single object >> |

- (Some additional information usually stored separately)

## Databases – Storing Spatial Data

- Knowing the x,y values is not enough!
- What do the following numbers represent? Draw as many possible representations of these numbers (single points, lines, polygons, combinations of these etc).

2,4, 4,3, 10,3, 13,5, 13,9, 11,13, 5,13, 2,11, 2,4, 7,5, 7,10, 10,10, 10,5, 7,5

GEOMETRY primitive type

- Overall GEOMETRY TYPE → Can be a geometry or collection of geometries or can constraint to one type e.g. only points
- CRS → The coordinate reference system (also known as an SRID)
- ELEMENT INFORMATION → The different parts of the geometry PostGIS uses brackets and geometry type names
- COORDINATE POINTS → List of x,y points In PostGIS each pair has a space between the x and y Each x, y pair is then separated from the next pair by a comma Brackets also used

## Overview

- What is spatial data
  - Georeferencing
- Modelling spatial data in a database
- Storing spatial data in PostgreSQL/PostGIS
  - **DDL – adding a spatial column**
  - DML – inserting data

## Adding a Geometry (Spatial) Column

- Create some test tables first

```
create table assetsclass.london_poi(id serial);

create table assetsclass.london_highway(id serial);

create table assetsclass.london_counties(id serial);
```

## Adding a Geometry (Spatial) Column

- Generic column can take any geometry type

```
select
AddGeometryColumn(<<schema>>,<<tablename>>,<<column
name>>,SRID, <<type of geometry>>,<<number of dimensions>>);

alter table assetsclass.building drop column if exists location;

select AddGeometryColumn('assetsclass','buildings','location',0,
'geometry',3);
```



## Adding a Geometry (Spatial) Column

- Column for a specific geometry type
  - This is a form of constraint



```
alter table assetsclass.rooms drop column location;

select AddGeometryColumn('assetsclass','rooms','location',0, 'polygon',2);
--        (2 dimensions, polygons)

alter table assetsclass.buildings drop column if exists location;

select AddGeometryColumn('assetsclass','buildings','location',0,
'polyhedralsurface',3);
          (3 dimensions, polyhedral surfaces)
```



## Adding a Geometry (Spatial) Column

- Coordinate reference systems and constraints - local reference system

```
alter table assetsclass.buildings drop column if exists location;

select AddGeometryColumn('assetsclass','buildings','location',0,
'polyhedralsurface',3);
```

## Adding a Geometry (Spatial) Column

- Coordinate reference systems and constraints – British National Grid

```
alter table assetsclass.London_counties drop column if exists
location;

select AddGeometryColumn
('assetsclass','london_counties','location',27700, 'polygon',2); --
British National Grid
```

## Adding a Geometry (Spatial) Column

- Coordinate reference systems and constraints – WGS84 (world wide)

```
alter table assetsclass.london_highway drop column if exists
location;
select AddGeometryColumn
('assetsclass','london_highway','location',4326, 'linestring',2);

alter table assetsclass.london_highway drop column if exists
location;
Select addGeometryColumn
('assetsclass','london_poi','location',4326, 'point',2);
```

## Overview

- What is spatial data
  - Georeferencing
- Modelling spatial data in a database
- Storing spatial data in PostgreSQL/PostGIS
  - DDL - adding a spatial column
  - **DML - inserting data**
- Visualising the Data

## Storing Spatial Data - PostGIS

- Well-Known Text
  - WKT is a human-readable format for representing geometry, and is therefore often used when populating databases using SQL.
  - Used by PostGIS for spatial data creation
  - Readable to the human eye
  - Not very compact

## Storing Spatial Data - Storing Spatial Data - PostGIS



- Well-Known Text
  - POINT(0 0)
  - LINESTRING(0 0,1 1,1 2)
  - POLYGON((0 0,4 0,4 4,0 4,0 0),(1 1, 2 1, 2 2, 1 2,1 1))

## Storing Spatial Data - Storing Spatial Data - PostGIS

- Well-Known Text
  - MULTIPOINT(0 0,1 2)
  - MULTILINESTRING((0 0,1 1,1 2),(2 3,3 2,5 4))
  - MULTIPOLYGON(((0 0,4 0,4 4,0 4,0 0),(1 1,2 1,2 2,1 2,1 1)), ((-1 -1,-1 -2,-2 -2,-2 -1,-1 -1)))
  - GEOMETRYCOLLECTION(POINT(2 3),LINESTRING(2 3,3 4))

## Storing Spatial Data - Storing Spatial Data - PostGIS

- Well-Known Binary
  - "The Well-known Binary Representation for Geometry (WKBGeometry) provides a portable representation of a geometric object as a contiguous stream of bytes.
  - It permits geometric object to be exchanged between an SQL/CLI client and an SQL-implementation in binary form" (OGC Simple Features Specification, 2006).

## Storing Spatial Data - Storing Spatial Data - PostGIS

- Well-Known Binary
  - Unlike WKT, WKB is not readable to the human eye, and is a more compact format for storing geometry objects.
    - It is therefore used in particular for data exchange and transferring data between one platform and another. It makes use of Binary Large Objects inside the database to store the geometry as a stream of bytes.

## Storing Spatial Data

- WKT and WKB
  - The WKT and WKB formats handle many of the items on the list of information required to be stored in a database to fully represent spatial data.
  - In particular, information describing how to use the coordinates (do they represent a point or set of points, a line, multiple disjoint lines, a simple polygon, a polygon with holes or multiple disjoint polygons, with or without holes) is present, as is information describing which coordinates in the list correspond to these individual object parts.

## Storing Spatial Data

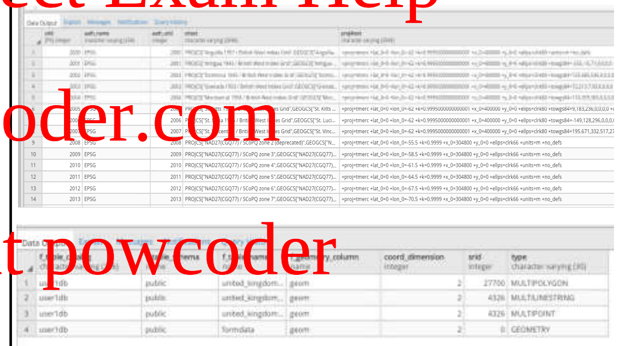
- WKT or WKB on their own are not enough
- You also need additional metadata to describe your spatial objects
- In PostGIS, the remaining information is stored separately from the main data, a metadata table (view) called *geometry_column*. This contains the following information about each spatial dataset:
  - schema_name - which user owns the dataset
  - table_name - the name of the spatially enabled table

## Storing Spatial Data - Storing Spatial Data - PostGIS



- In PostGIS, the remaining information is stored separately from the main data, a metadata table called *geometry_column*. This contains the following information about each spatial dataset:
  - column_name - the column containing the spatial information
  - srid - the spatial reference system (this cross-references a table called SPATIAL_REF_SYS, which contains information about over 3000 coordinate systems)
  - type- used to restrict the column to a single type of spatial object (point, line or polygon etc) if required
  - dimension - whether the data is 2 or 3D.







## Inserting Spatial Data

- Geometry is treated the same as the other columns
  - Use ST_GEOMFROMTEXT and WKT
  - Note the quotation marks

```
insert into assetsclass.rooms (floor, room_number, building_id, location, last_repainted, room_use)
values
(1, '1.01',(select building_id from assetsclass.buildings where building_name = 'Chadwick'),
st_geomfromtext('POLYGON((3 2, 8 2, 8 12, 3 12, 3 2))', 0),'12-Jan-1950','classroom');
```

## Inserting Spatial Data

- …. (note the quotation marks)

st_geomfromtext('POLYGON((3 2, 8 2, 8 12, 3 12, 3 2))', 0)

## Inserting Spatial Data – National CRS

insert into assetsclass.london_counties (location) values (st_geomfromtext('POLYGON((328103 186492, 328108 186492, 328108 186502, 328103 186502, 328103 186492))', 27700));

## Inserting Spatial Data – Global CRS

insert into assetsclass.london_poi (location)
values
(st_geomfromtext('POINT(-5.4233444 50.1876552)',4326));

insert into assetsclass.london_highway (location)
Values
(st_geomfromtext('LINESTRING((-4.1997314 50.4060347,-4.1998784 50.4061017,-4.1999345 50.406374,-4.2000396 50.4066819,-4.2002497 50.4069987,-4.2004388 50.4071015,-4.2006227 50.4070739))',4326);



## Inserting Spatial Data

- Violating a constraint …

insert into assetsclass.london_poi (location)
values
(st_geomfromtext('MULTIPOINT(-5.4233444 50.1876552)',4326));

-- Geometry type (MultiPoint) does not match column type (Point)

## Inserting Spatial Data – Using an UPDATE statement

update assetsclass.buildings set location = (st_geomfromtext('POLYHEDRALSURFACE(((480501.5 131048.4 0,480501.5 175767.7 0,543813.3 175767.7 0,543813.3 131048.4 0,480501.5 131048.4 0)))', 0)) where building_id =(select building_id from assetsclass.buildings where building_name = 'Chadwick');

## Overview

- What is spatial data
  - Georeferencing
- Modelling spatial data in a database
- Storing spatial data in PostgreSQL/PostGIS
  - DDL – adding a spatial column
  - DML – inserting data
- **Visualising the Data**

## Visualising the Data

- As WKT (well known text) is a standard, then most GIS software packages can easily connect onto a database that uses this standard and visualise the data
  - As a reminder – GIS = geographical information system – the software that stores, edits, analyses and visualises spatial data
  - Map creation – and the spatial SQL we will see next week – is just a very small part of what a GIS can do

## Visualising the Data

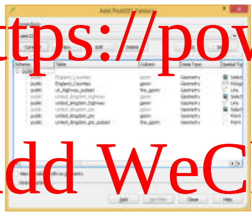
- For this module, we will be using two GIS software packages to visualise the data
  - QGIS (NB: Version 2.18)
  - FME

## Step 1 – connect to the database

- Connect to the database (use your databases username and password for this)

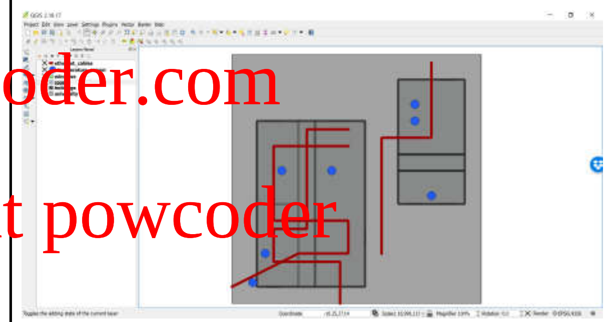
## Step 2 – select the layer(s) you want to view

- From data that already exists in your database
- from the spatial tables you have created (once you have done this)





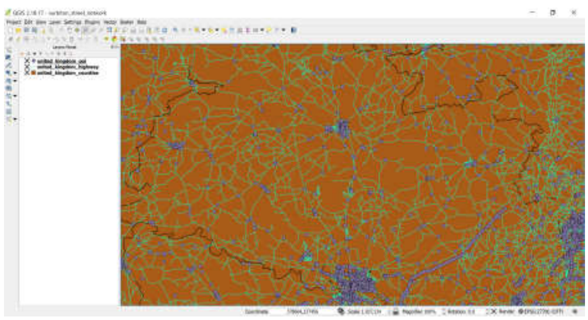
## Step 3 – View the Data



## Step 4 – map styling

- In a GIS you can – in theory – make your layers use any colours you like
  - Cartographers - map makers – do have general guidelines e.g. motorways are orange
  - Large mapping projects – e.g. HS2 or Crossrail – will have their own styles and templates
  - Some disciplines – e.g. space syntax – also have their own conventions

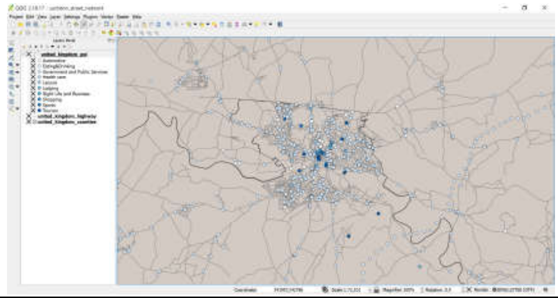
## Step 4 – choose colours (random)



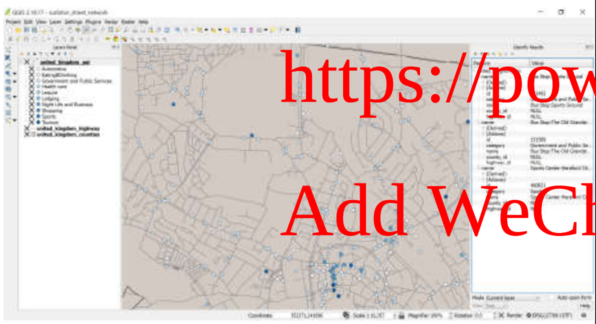
# Map Styling

- http://proceedings.esri.com/library/userconf/fed16/papers/fed_86.pdf
- ColorBrewer provides some help on colour choices http://colorbrewer2.org/#type=sequential&scheme=BuGn&n=3
- Production maps also need a legend, scale bar and north arrow
  - (Geospatial students – you will learn about this in more detail)
- For your assignment – screenshots are sufficient, professional maps not required

# Step 4 – choose colours (by category)

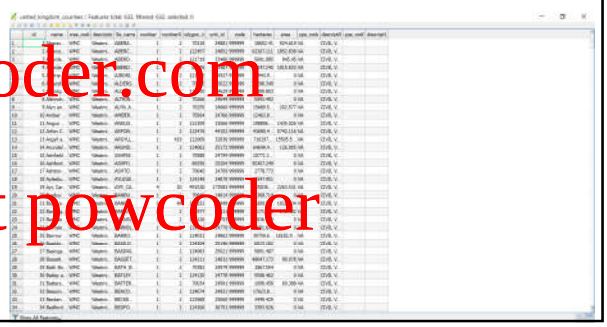


# Step 5 – see the other attributes





# Step 5 – see the other attributes



# Connecting via ArcMap

- For the MSc Geospatial Science, MSc Spatio-Temporal Analytics, MSc Civil Engineering with GIS ..
  - You can also connect to PostGIS from R (apparently)
  - ArcMap – use ArcCatalog Database Connections
- (not required for your assignment)


https://www.reddit.com/r/dataisbeautiful/comments/9oxlct/a_mosaic_of_world_countries_retaining_their/
http://www.freeusandworldmaps.com/images/WorldProjections_Maps/WorldProjectionCountriesColorA.jpg