# Large Scale Databases



## Spatial Data Models

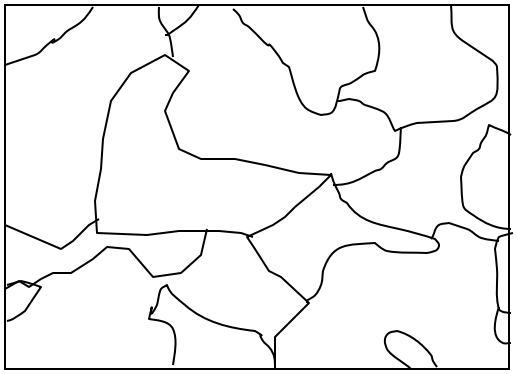


Chris Jones

# Spatial Data Models

2 main types of spatial data model:

- Field-based or location-based
  'raster model'
- object-based
  'vector model'



# Raster Model

- Divide ground surface into regular grid cells
- record what is present in each cell
  - class of landcover or value of measured property
- individual raster is usually theme-based
  - i.e. represents particular class of data
- may be several raster 'layers' in a database
- store each raster as matrix (array) of numeric data values

# Raster model

- Often derived/interpreted from scanning device images

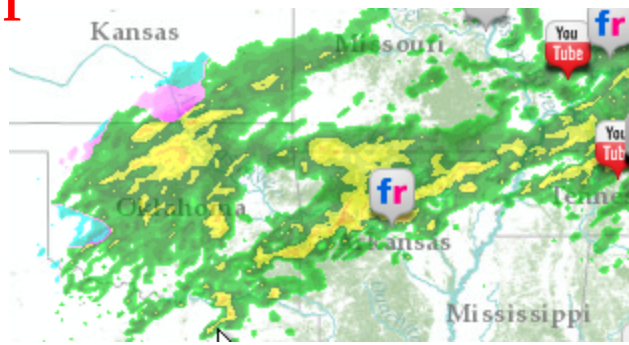
  e.g. multispectral scanners

  or *rasterised* from vector data



- Good for:
  - representing gradual change and imprecise phenomena
  - fast computation (especially overlay operations, to find where multiple classes of phenomena coincide in space)






- Poor for
  - precisely surveyed objects

    As it has fixed resolution

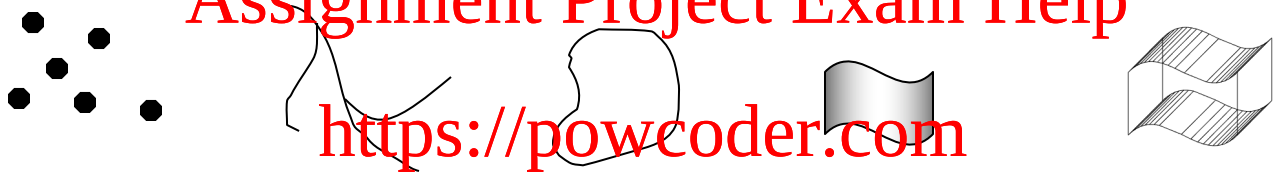
    And lacks explicit object structure

# Vector Model

- Describes the location of discrete objects using geometry
- precise locations
- can record boundaries of complete objects
- good for determining spatial relationships between whole objects
- requires more complex geometric computation to determine spatial relations

# Geometric Representation of Locations (with coordinates) in Vector Model

- Primitive geometric objects
  - points, lines, areas, surfaces, volumes



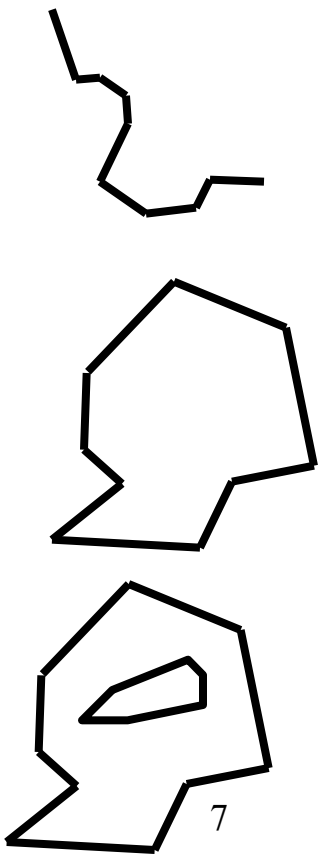


  - Facilitate measurements (e.g. distance, length, height, area, volume, orientation)

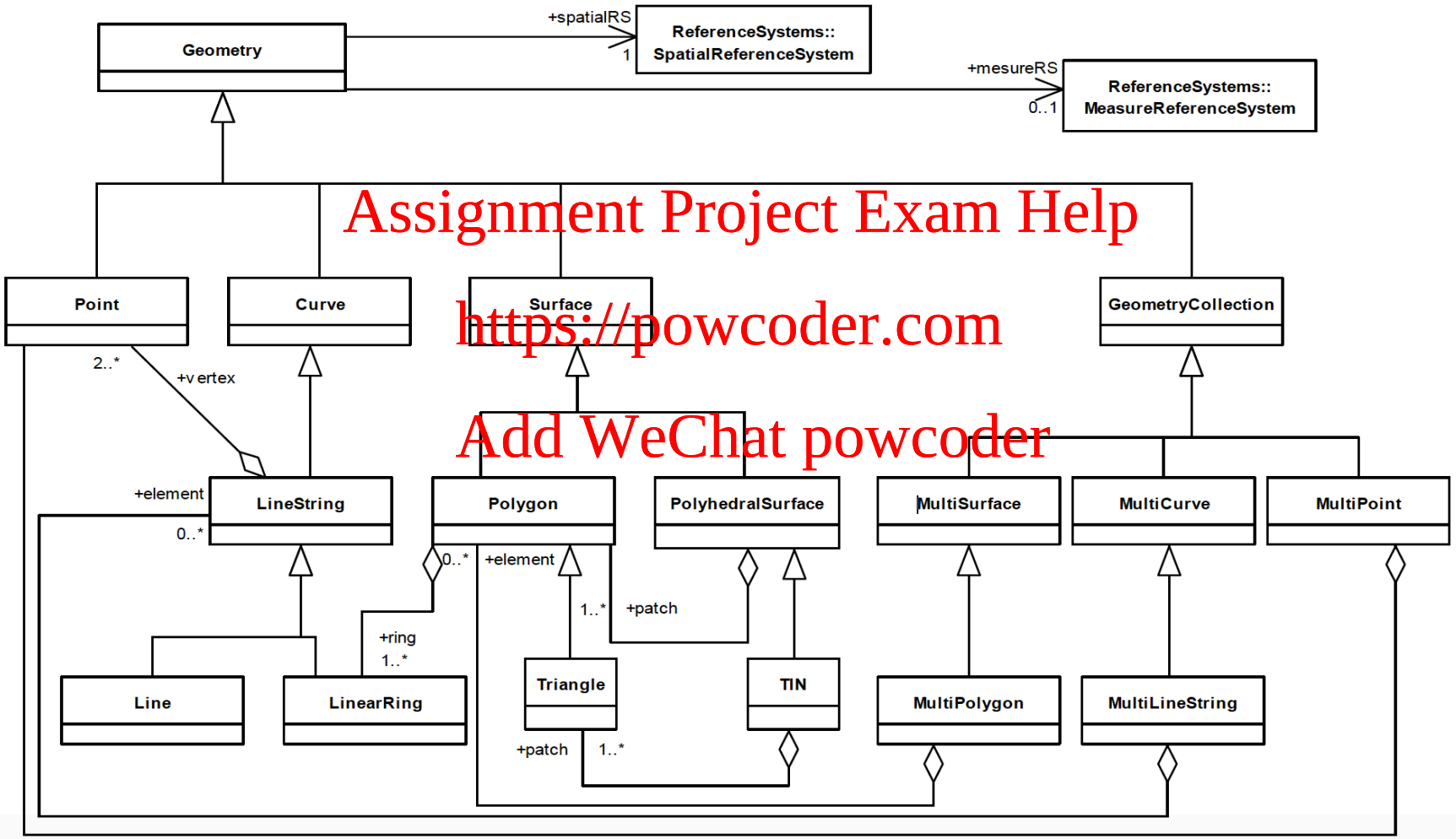
# 2D Geometry Data Types

- *point*
  - defined by X, Y, (Z) coordinates
  - e.g. telegraph post, town at small scale
- *line (polyline / line string)*
  - defined by list of points (x1,y1…..xn,yn)
  - e.g. roads and rivers
- *simple polygon*
  - defined by closed list of points
  - e.g. building, administrative district, lake
- *complex polygon*
  - defined by exterior polygon and one or more interior polygons (holes)



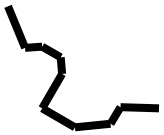


# Simple Feature Geometry Class Hierarchy from Open Geospatial Consortium

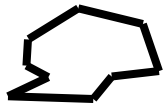


# Open Geospatial Consortium (OGC) Geometry Types

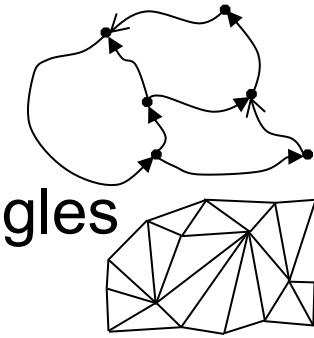
- Point, Curve, Surface are types of Geometry
- LineString is a Curve = sequence of points with linear interpolation between the points



- Line is an open LineString *with only 2 points*
- LinearRing is closed LineString



- Polygon is a planar Surface bounded by 1 or more Linear Rings
- Triangle is a Polygon



- Polyhedral surface is polygons that share boundaries
- Triangulated Irregular Network (TIN) consists of Triangles
- TIN is a PolyhedralSurface
- MultiPolygon is a Multisurface is a GeometryCollection
- MultiLineString is a MultiCurve is a GeometryCollection

# Large Scale Databases



# Locations and Coordinate Systems



Chris Jones

# Spatial Information

- Describing Location : qualitative vs quantitative
- Spatial Data Models (conceptual models)
- Geometry Data Types
- Spatial Relationships
  - Topological relations
  - Orientation relations
  - Proximal relations
- *Shape and Pattern*
- *Correlations*
- *Interactions*
- *Routes*



# Location

## Qualitative *vs* Quantitative

- Qualitative: Place names, post codes / zip codes

  e.g. Cardiff, Queens Street, CF24 3AA

- Quantitative : Coordinates (lat/long *vs* map grid)

  e.g. (52.3, -2.3)  (248667, 775925)

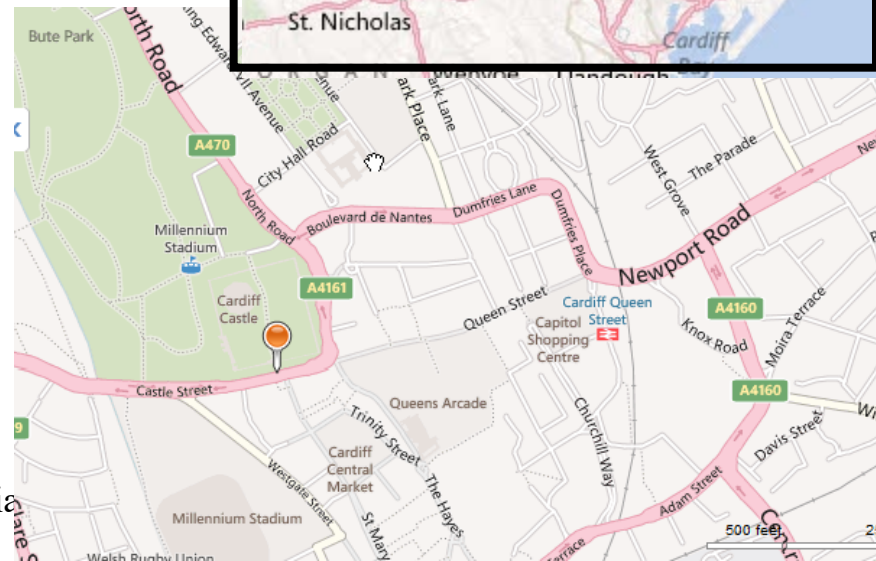
For GIS the two are usually combined



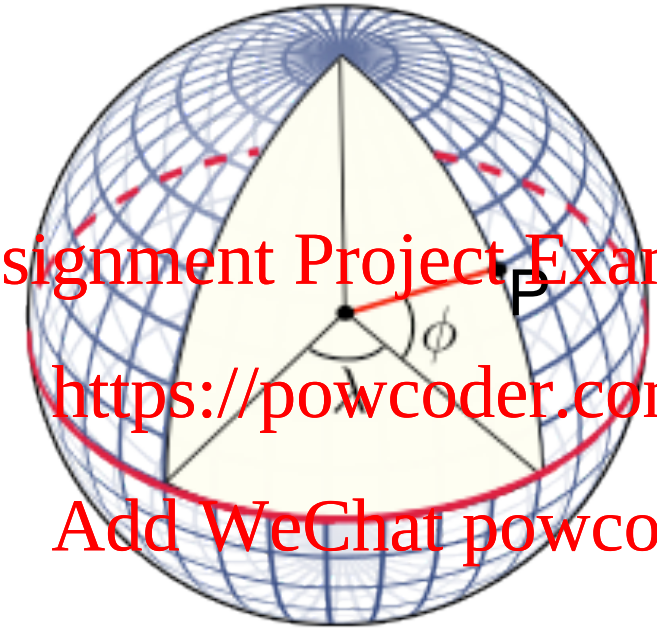

## Absolute *vs* Relative

- Absolute : place name or coordinates, e.g. Taffs Well
- Relative : spatial relationship to absolute position

e.g. North of Cardiff / Near Cardiff



# Coordinate Systems: Latitude and longitude (global)

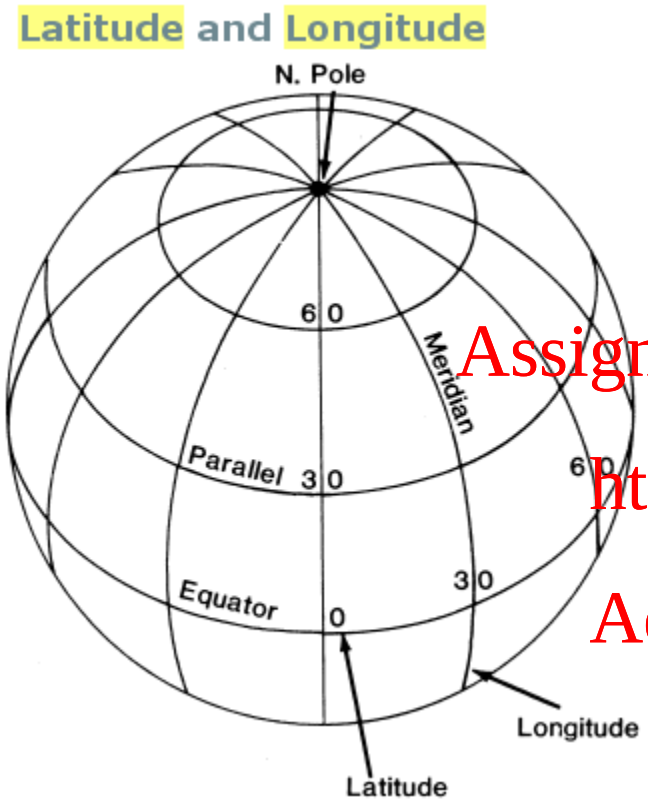





Latitude Φ at point P is the angle between the line perpendicular to the Earth's surface at P and the plane of the equator

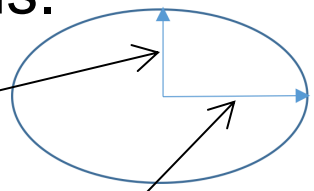
Longitude λ is angle between the plane passing through P and the Earth's poles and the plane of the prime meridian (also passes through poles)

# A Datum



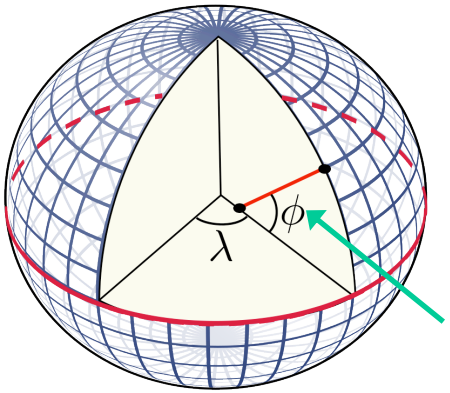



For purposes of specifying coordinates and map projections, the Earth is approximated in shape by a spheroid – an ellipse rotated about its vertical axis.

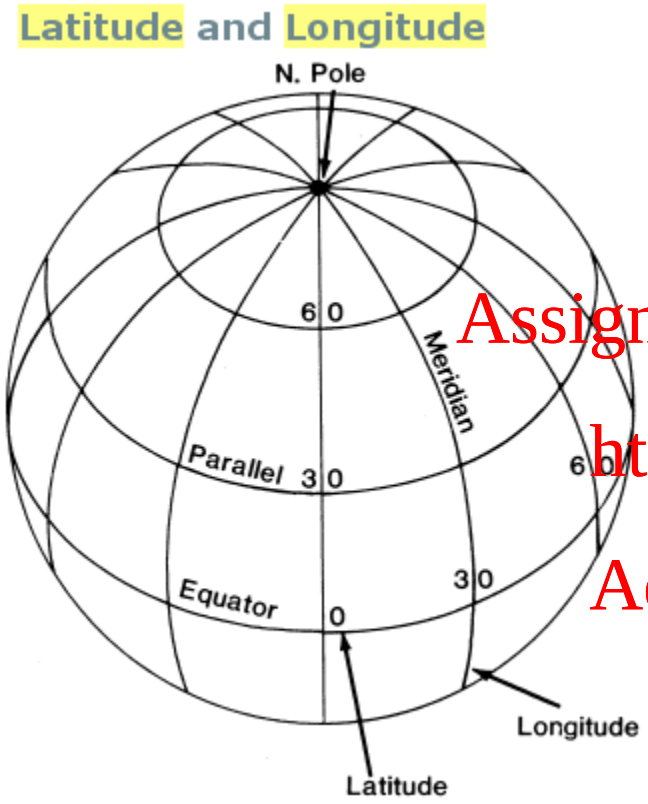


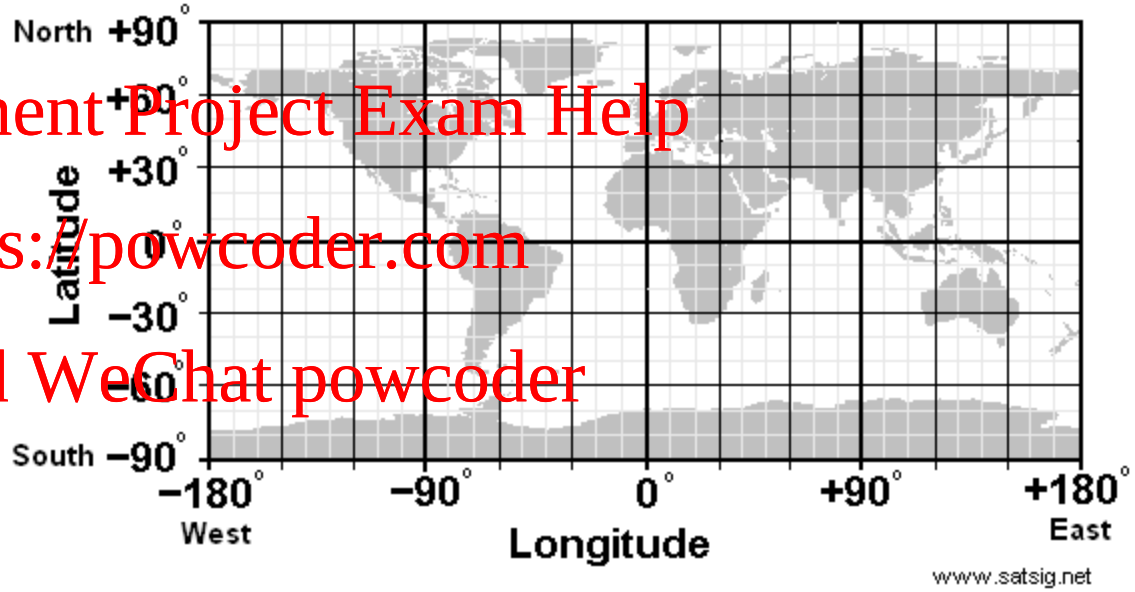

The spheroid is a called a Datum and is defined by the lengths of two radii.

The datum used for GPS coordinates is called WGS84



# Latitude & Longitude Coordinate Systems






A global coordinate system.

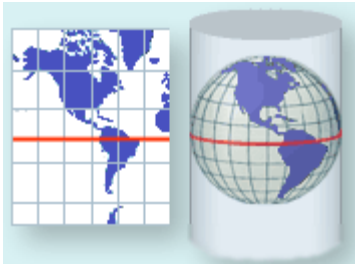
Treating latitude and longitude as Cartesian (rectangular ) coordinates (above right) results in major distortions of areas and angles → hence need local map projections

# Map Projections and Map Grids

Project rays from centre of globe to a surface (cylinder, cone, plane) that can be unwrapped to, or is, a plane.

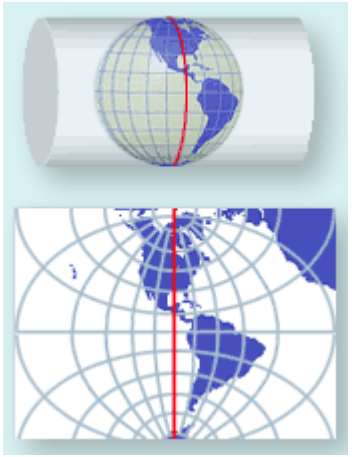
The resulting 2D planar coordinate system – in metres (not angles) - is referred to as a Map Grid or Map Projection





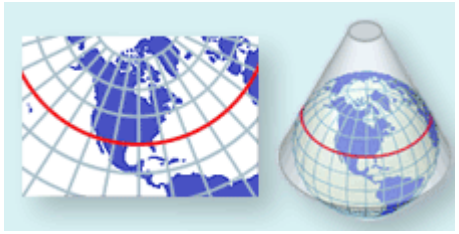
Cylindrical Projection

Azimuthal Projection (a plane)



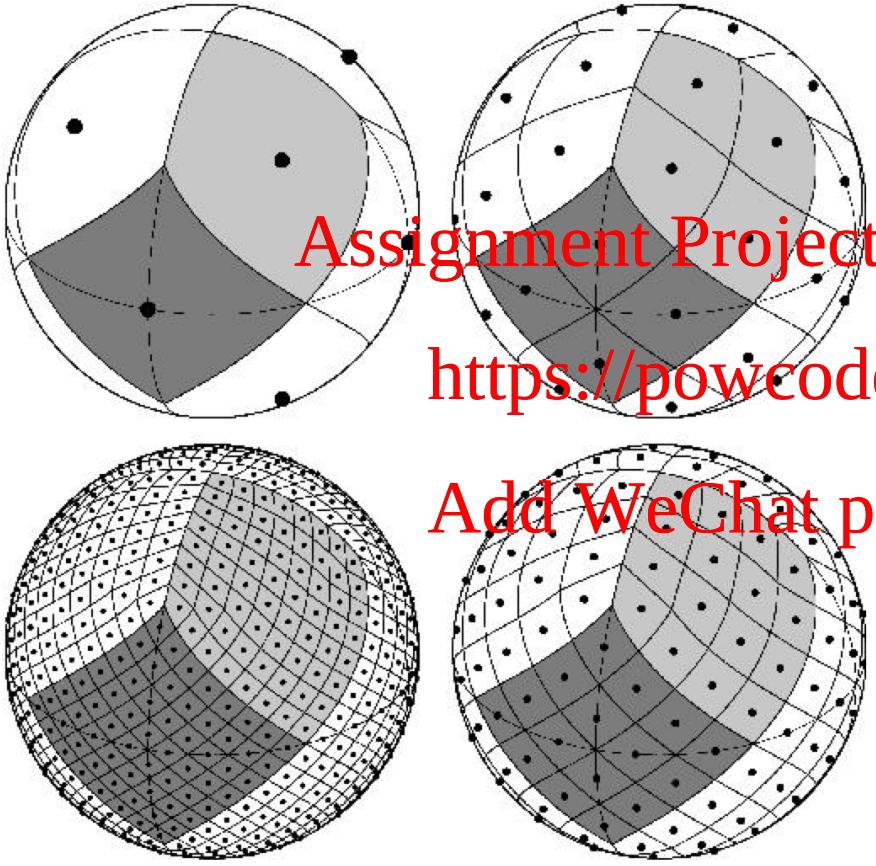
Transverse Cylindrical Projection

e.g. British National Grid



Conical Projection

# Discrete Global Grids, e.g. HealPix (the previous map grids are local)





DGG divide surface of Earth into approx equal size cells (at different resolutions.

HealPix uses rectangular cells. Others DGGs use triangulations of the Earth's surface

Nasa Jet Propulsion Lab.
http://healpix.jpl.nasa.gov/healpixBackgroundPurpose.shtml

# Large Scale Databases





## Spatial Relationships

# Significance of Spatial Relationships to Spatial Databases

- Queries to spatial databases often employ spatial relationships to express what data are to be retrieved



For example:

- *which objects lie inside a particular region?*
- *which regions are neighbouring another region?*
- *which areas does a region overlap?*
- *what objects a particular class are within some particular distance of another object?*

# Spatial Relationships: *topological, proximal, orientation*

**Topological relations (qualitative)**

- Different types of connectivity between objects (e.g. containment, touching, overlap..)
- Invariant to rotation, translation and scaling



**Proximal relations (qualitative / quantitative)**

- Refer to distance between objects

**Orientation relations (qualitative / quantitative)**

- Refer to the direction/orientation between objects

# Topological relations between points, lines and areas

The main topological relations are:

- **meet (or touch)**
- **inside / contained** (converse relations)
- **overlap**
- **separate / disjoint** (not connected)
- **equal**

- **Other relations are covers and covered by**

# Topological relation: meets (touch)

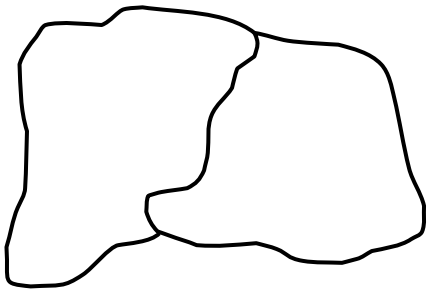
**meets (or touch)**

*Two geometry objects meet/touch if there is some intersection between A and B but their interiors do not intersect*



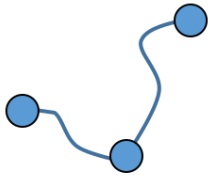
**Examples**

Two regions A and B meet if part of the boundary of A coincides with part of the boundary of B, but they do not otherwise overlap.
E.g. county A shares boundary with county B



# Meets / touch continued

Two lines touch if their end points (boundary) intersect or if the boundary of one intersects the interior of the other



e.g. road A meets another road B



*(the boundary of a line is its end points, and the remainder is the interior of the line)*



A point and a line touch if the point coincides with the boundary of the line

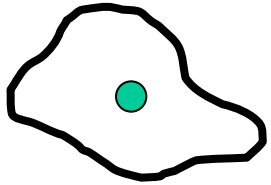
*(NB a point has no extent and so is not regarded as having a boundary)*

# Topological relation: Inside / Contained

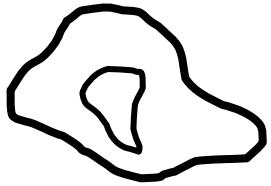
**inside / contains** (converse relations)
A is inside (contained by) B
(i.e. B contains A)
if all parts of A coincide with the interior of B and no parts of their boundaries coincide.
[= *"proper part" but see covers relation*]



e.g.
town A inside county B



nation X contains province Y



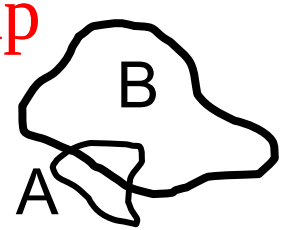
# Topological relation: Overlap

*Two objects A and B **overlap** if part of the interior of A intersects with part of the interior of B*

*while part of the interior of A intersects with the exterior of B*



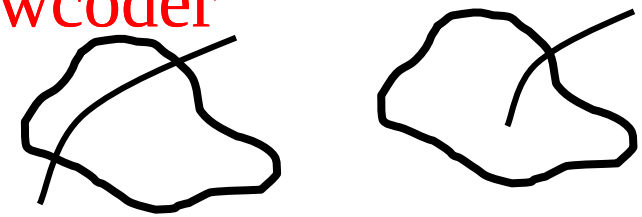
E.g. national park X overlaps county Y



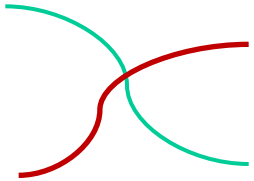





E.g. road A crosses county X



E.g. road A crosses road B



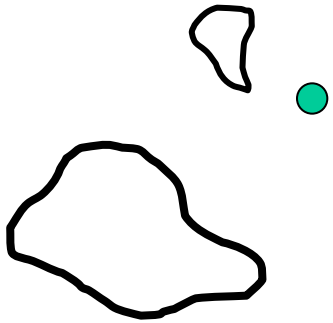
# Topological relation: Disjoint

**separate / disjoint** (not connected)

A is disjoint from B if no part of A coincides with any part of B



e.g. lake X is separate from the city Y



# Topological relation: Equal

**Equal**

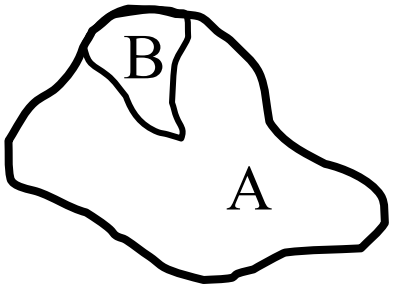
Two objects A and B are equal if each (and every) part of A coincides with a part of B and each (and every) part of B coincides with a part of A

# Topological relation: Covers / Covered By

**Covers / covered by**

Object A covers object B if A contains B and part of B's boundary coincides with part of A's boundary. In that case, B is covered by A.

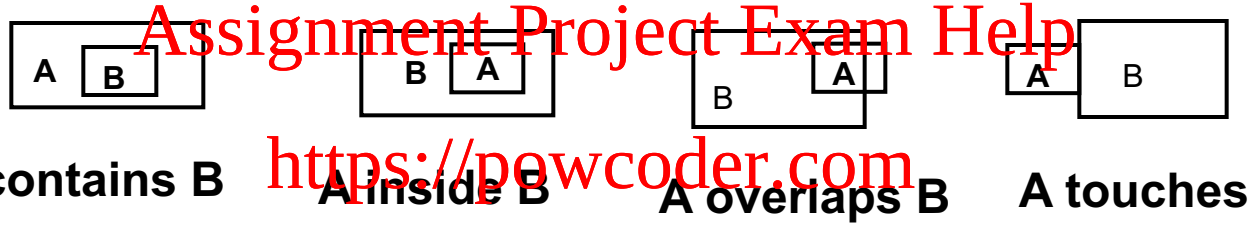


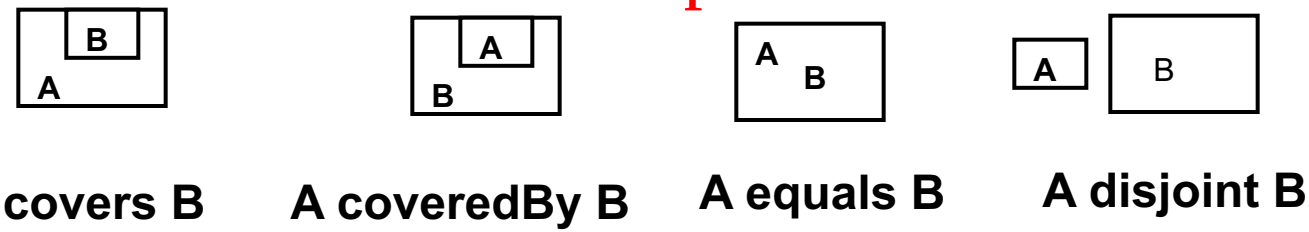



e.g. England covers the county of Kent (the coastline of Kent is the coastline of England)

# Summary of Topological Relations



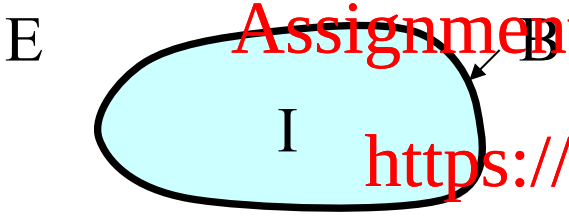


# 9-Intersection model to define topological relations between regions

Treats spatial objects as regions having 3 components:

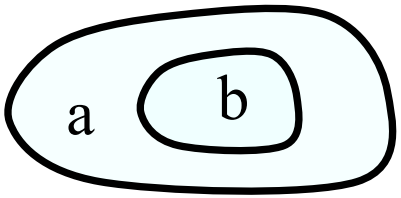
Boundary (B), Interior (I) and Exterior (E)






The topological relations between two objects can be characterised by which of these components intersect each other

For Example:




a contains b

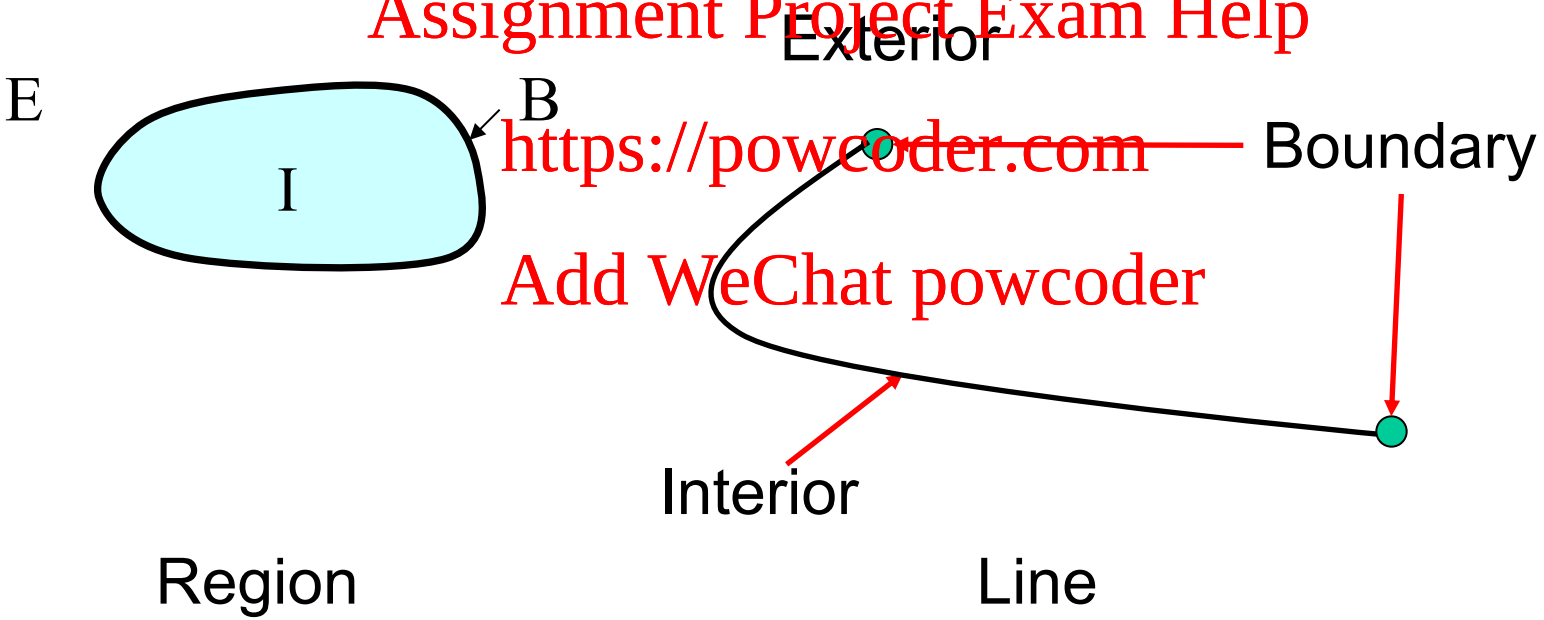
| a \ b | Interior | Boundary | Exterior |
|---|---|---|---|
| Interior | I(a) ∩ I(b)= 1 | I(a) ∩ B(b)= 1 | I(a) ∩ E(b)=1 |
| Boundary | B(a) ∩ I(b)=0 | B(a) ∩ B(b)=0 | B(a) ∩ E(b)=1 |
| Exterior | E(a) ∩ I(b)=0 | E(a) ∩ B(b)=0 | E(a) ∩ E(b)=1 |

# Applying the 9-intersection model to lines

Treats spatial objects as having 3 components:

Boundary (B), Interior (I) and Exterior (E)

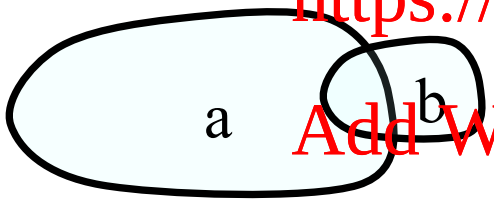





A Point has no boundary.

# Exercise: define the intersection matrix for **a Overlaps b** relation where a and b are regions






| a \ b | Interior | Boundary | Exterior |
|---|---|---|---|
| Interior | | | |
| Boundary | | | |
| Exterior | | | |

# Orientation relations

- *quantitative*: measured angular bearing, relative to north, or some other direction
  - e.g. person on bearing 45 degrees from lamp post (measured clockwise relative to north, but bearing could also be measured anticlockwise)
- *qualitative*: e.g. north, south, east, west, above, below, in front.....
  - e.g. person is north east of the lamp post

# Proximal relations

- quantitative
  - measured distance

  e.g. city A is 10Km away from city B
- qualitative
  - "near", "close", "far", "distant" etc.

# References for the 9-intersection model and RCC (Region Connection Calculus)


M.J. Egenhofer, R. Franzosa (1991) Point-set topological spatial relations. International Journal of Geographical Information Systems, 5 (2), pp. 161-174

M.J. Egenhofer, J. Herring (1991), Categorizing binary topological relations between regions, lines and points in geographic databases, Technical Report, Department of Surveying Engineering, University of Maine, 1991.





E. Clementini, P. Di Felice, and P. van Oosterom (1993), A Small Set of Formal Topological Relationships Suitable for End-User Interaction, Proceedings of the Third International Symposium on Advances in Spatial Databases, p.277-295, June 23-25.




*You can obtain copies of these articles by searching for their titles on Google.*

Wikipedia (https://en.wikipedia.org/wiki/DE-9IM) has a reasonable explanation of the dimensionally extended 9-intersection model as introduced in Clementini


G. Cohn; B. Bennett; J. Gooday; M. M. Gotts (1997). "Qualitative Spatial Representation and Reasoning with the Region Connection Calculus". GeoInformatica. **1** (3): 275–316

Randell, D. A., Cui, Z. and Cohn, A. G. “A spatial logic based on regions and connection,” *Proc. 3rd Int. Conf. on Knowledge Representation and Reasoning*, Morgan Kaufmann, San Mateo, pp. 165–176, 1992.

# Spatial Database Management: Representation of Geographical Information



Chris Jones

# Modelling Geographical Information

- Geographical Objects have
  - identity
    - name, unique id
  - classification
    - one or more classes
  - thematic attributes / properties / statistics
    - various (non-spatial) data types
  - spatial representation
    - vector or raster structured data

# Example geographical objects

| **County Object** | **City Object** | **Street Object** |
|---|---|---|
| Name | Name | Name |
| Capital City | County | City |
| Population | Population | Surface-material |
| **Boundary** | Date of origin | Width |
| | **Boundary** | **Centre-Line** |

# Data Types

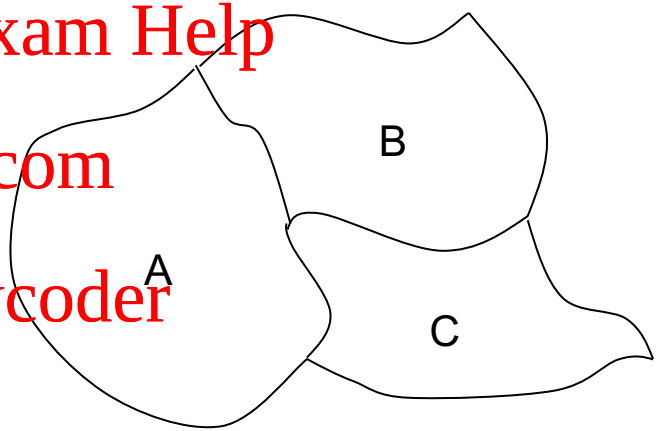
- Identity, class, thematic attributes
  - standard database types such as string, number, date
- Spatial representation (e.g. boundary)
  - specialised geometric data for vector or raster data models

# Example data for a set of counties

County records

| Name | Capital | Population | Boundary |
|---|---|---|---|
| Bigshire | Upton | 4002 | A |
| Mereshire | Downton | 67358 | B |
| Bottleshire | Fizzton | 108 | C |
| …. | ….. | ….. | ….. |






Boundary geometry records / objects

| Boundary | Coordinates |
|---|---|
| **A** | x1,y1,x2,y2…..xm, ym |
| **B** | x1,y1,x2,y2…..xn, yn |
| **C** | x1,y1,x2,y2…..xo, yo |

Alternative ways in which the geometry could be stored : e.g.
Boundary as sequence of line segments - segments shared

# Fixed and variable length fields

- County records
  - all fields consist of single data item, one of which is geometry data type
- Boundary geometry data type
- Coordinates = variable length list
  - Order (sequence) of coordinates must be maintained

# Applying relational databases to spatial data

- The need for Object-Relational Databases and OO Databases with spatial capability
  - [ Normalisation of the geometric data would result in very inefficient storage and performance ]



- Complex data types
  - to represent geometry objects of points, lines, polygons



- Spatial functions and operators
  - to compute properties of and spatial relations between geometry objects



- Spatial indexes
  - to provide efficient execution of spatial query operators such as for topological and proximity (distance) relations.

# Object-Relational Spatial Databases

- Examples of DBMS with spatial functionality
  - Oracle
  - PostgreSQL (PostGIS)
  - Microsoft SQL Server
  - IBM DB2 (Spatial Extender)
  - MySQL



*[[ + NoSQL databases such as MongoDB. ]]]*

# Example data type declarations

```
CREATE TYPE  Point AS OBJECT (
                X NUMBER,
                Y NUMBER,
                Z NUMBER,
                MEMBER FUNCTION Distance (P2 IN Point)
                                        RETURN NUMBER );

CREATE TYPE LineType AS VARRAY(1000) OF Point;

CREATE TYPE LineString AS OBJECT(
                Number_of_Points     INT,
                Geometry             LineType,
                MEMBER FUNCTION Length (SELF IN LineString)
                                        RETURN NUMBER );
```

# Example Table Declarations

- CREATE TABLE Settlements (
  Name Varchar(30),
  Population Number,
  Location Point );



- CREATE TABLE Roads (
  Name Varchar(30),
  Type Varchar(30),
  Location LineString );

# Note on Oracle Spatial Data Types

- Has (very complex) geometry object called

SDO_GEOMETRY

  - Can represent points, lines, polygons etc…
  - Multiple parameter settings to specify different sorts of geometry
  - Is associated with SDO_GEOM.XX functions to compute spatial properties and SDO_XX operators to determine spatial relations between geometry objects

Also: SDO_POINT_TYPE

  - Just for single points
  - Can be used as part of the SDO_GEOMETRY object

# Oracle SDO_GEOMETRY object

```
CREATE TYPE sdo_geometry AS OBJECT (
        SDO_GTYPE    NUMBER,
        SDO_SRID     NUMBER,
        SDO_POINT    SDO_POINT_TYPE,
        SDO_ELEM_INFO     SDO_ELEM_INFO_ARRAY,
        SDO_ORDINATES     SDO_ORDINATE_ARRAY);

CREATE TYPE sdo_point_type AS OBJECT (
        X NUMBER,
        Y NUMBER,
        Z NUMBER);

CREATE TYPE sdo_elem_info_array AS VARRAY (1048576) of NUMBER;

CREATE TYPE sdo_ordinate_array AS VARRAY (1048576) of NUMBER;
```



*See Exercise for examples of using SDO_GEOMETRY*

# Spatial Database Management: Query of Geographical Information



Chris Jones

# Querying Spatial Databases

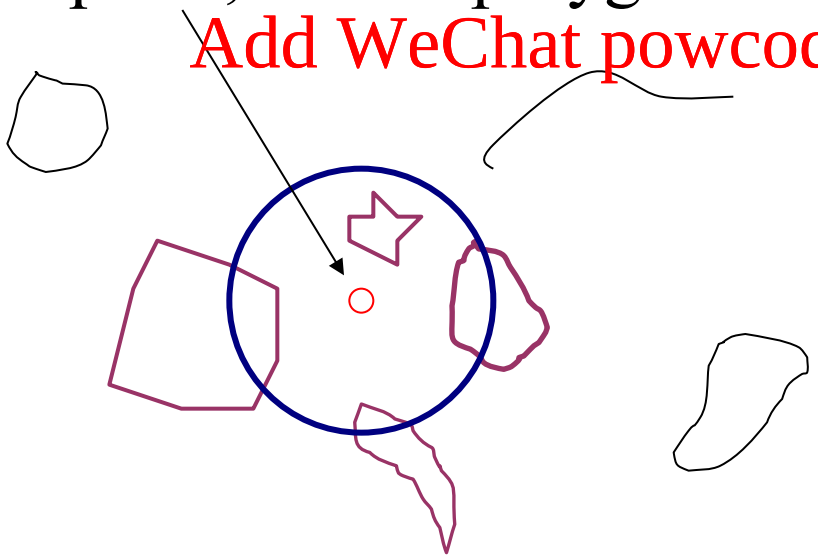
- Proximity / distance -based
  - find objects within a distance of a specified feature
- Topological relations
  - Retrieve objects that satisfy topological relations
- Intersection (spatial join / map overlay)
  - Retrieve ***geometry*** at intersection of locations of spatial objects
- Region Containment / Spatial Filter
  - retrieve data including geometry in a specified region – Filter is approximate and results could be at least partly outside

# Proximity / Distance Queries

## Within distance query

- Find properties of features within specified distance of specified feature
  - relative to point, line or polygon feature





# Generic Spatial SQL operators for Within Distance

**Find Names of villages within 5km of Cardiff:**

```
Select S2.name
From Settlements S1, Settlements S2
Where S1.name = “Cardiff” and
      distance(S1.location, S2.location) < 5000
      and S2.type = “village”;
```



Function **distance** returns a real number (the distance) and takes as parameters two geometric objects

NB: Oracle has a distance function SDO_GEOM.SDO_DISTANCE that computes the distance. It also has a within distance operator SDO_WITHIN_DISTANCE

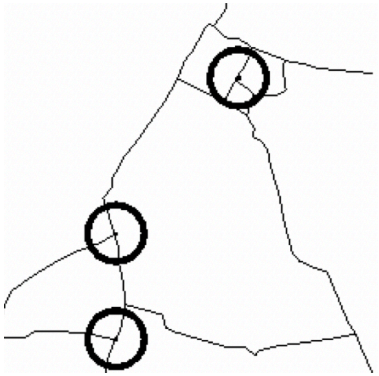
# Distance Buffers

- Generate a region of space that is some distance from a specified geometry object.
- The buffers are then used in intersection queries to retrieve the data inside them. See later slides.



Oracle has a Buffer function (SDO_GEOM.SDO_BUFFER)

Point buffers

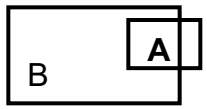


Line buffers

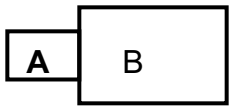
# Topological relations typically supported (OpenGIS standard)


A contains B


A inside B

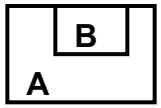



A overlaps B




A touches B

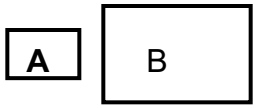





A covers B


A coveredBy B


A equals B




A disjoint B

# Generic Spatial SQL queries for topological relations

**e.g. Names of woodlands greater than 2000 square metres inside, or partially inside, Dyfed:**

```
Select  Woodlands.name
From  Woodlands, Counties
Where
    inside(Woodland.boundary, Counties.boundary)
    or
    overlap(Woodland.boundary, Counties.boundary)
    and
    Woodlands.area > 2000 and
    Counties.name = “Dyfed”;
```



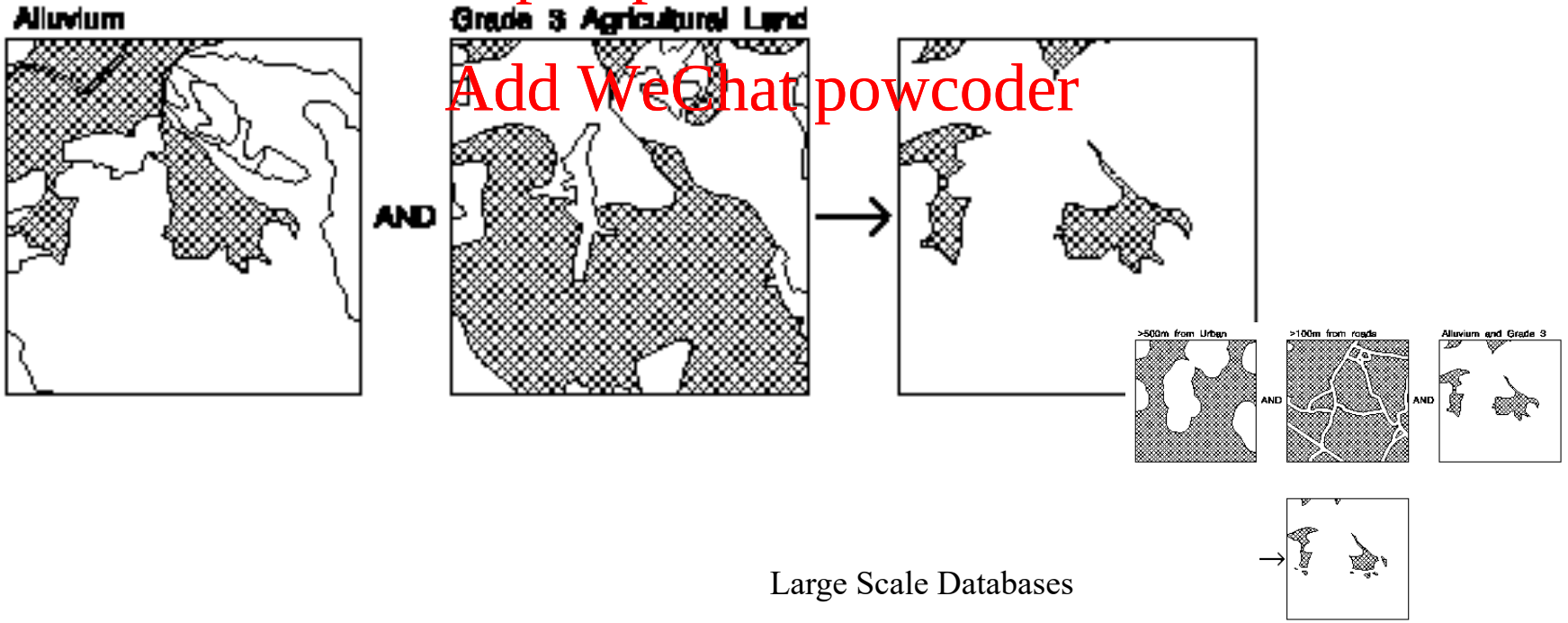
inside and overlap return a Boolean and take as parameters two geometric objects that could be lines or polygons

# Intersection (also called overlay or spatial join)

- Returns geometry of locations where a set of two or more spatial features coincide or intersect
- The spatial features could be, for example
  - regions with particular thematic classification or identity
  - regions that are within specified distance of other objects, i.e. buffer zones



Alluvium and Grade 3 Agricultural Land



# Buffer queries (intersection)

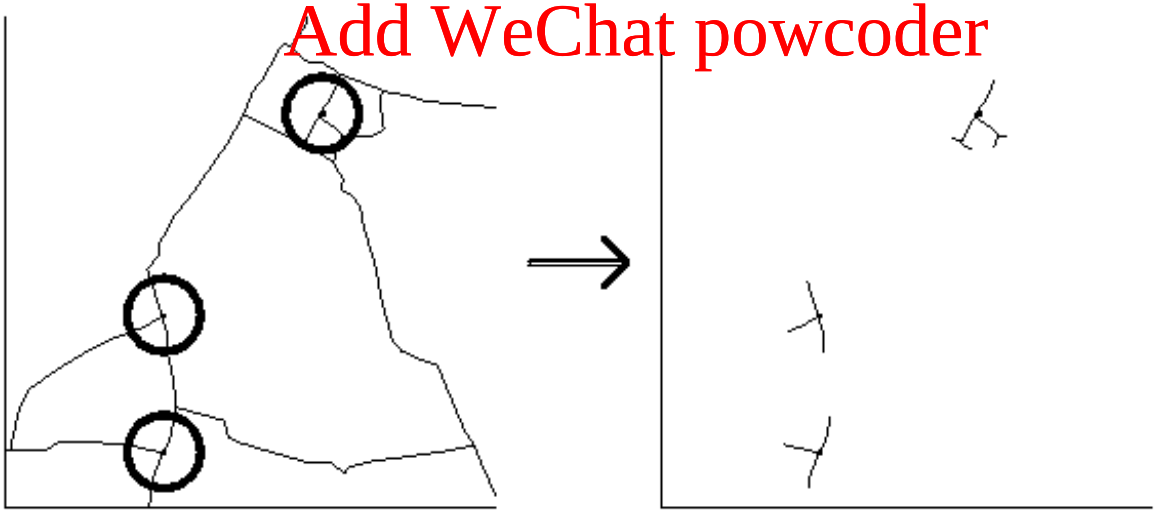
- ***Retrieve geometry*** within distance of a another geometry objects (as opposed to measuring distance or finding objects within a distance)



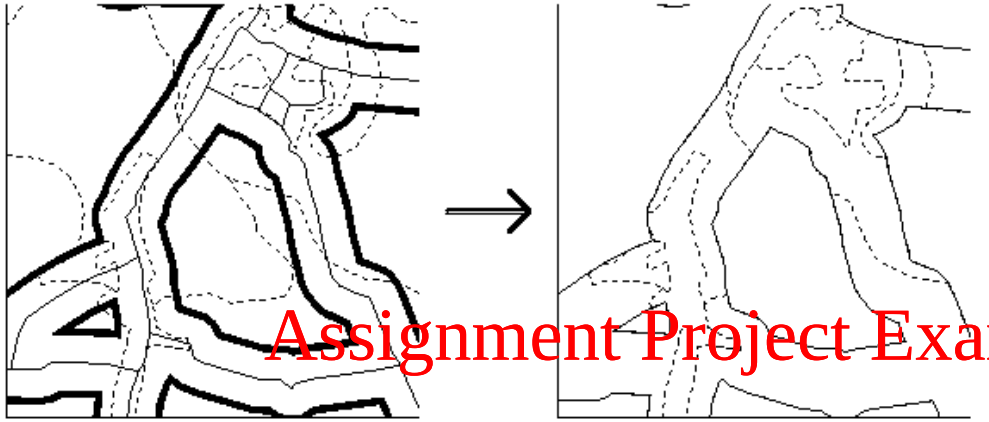
Retrieve geometry within distance of a point object





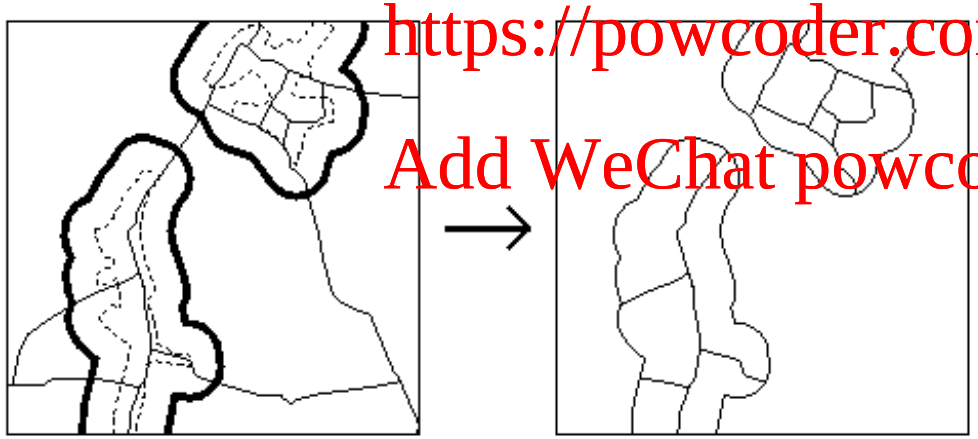


# Buffer queries (continued)



Retrieve geometry within distance of a line

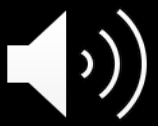




Retrieve geometry within distance of a polygon

Note that when Oracle Buffer function SDO_GEOM.SDO_BUFFER is applied to lines it operates separately on each individual segment
So in the above examples lines would need to be continuous

# Generic Spatial SQL query for intersection



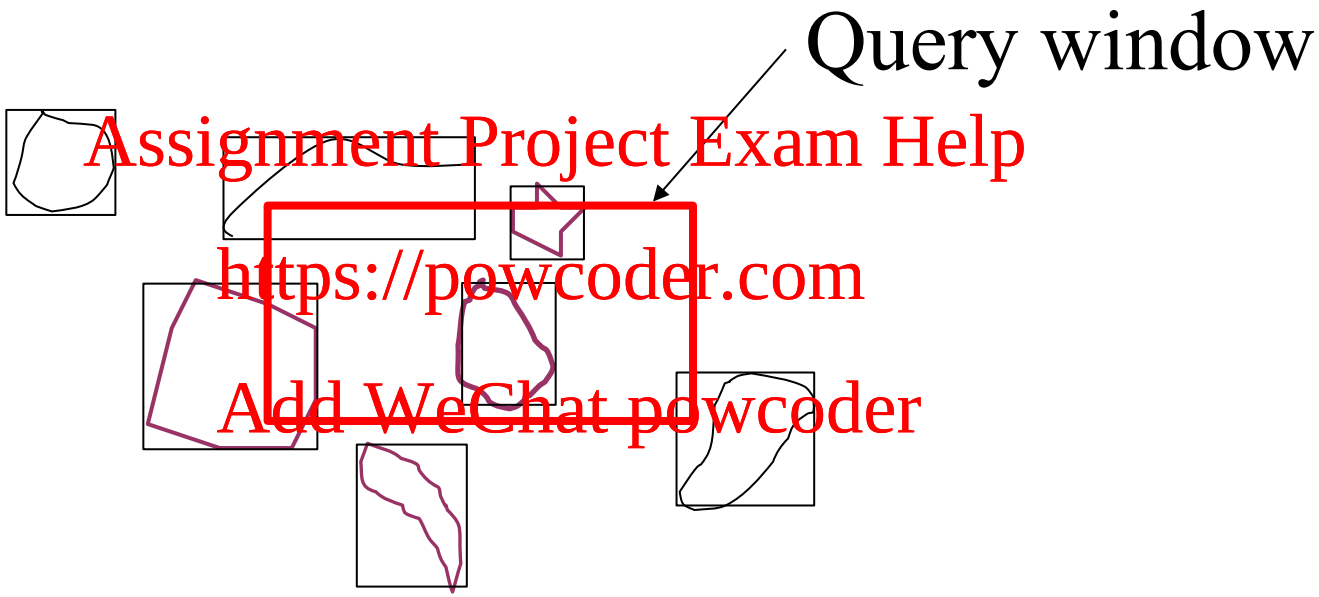
**Total area of woodlands in county of Dyfed:**

```
Select
    sum(area(intersection(Woodlands.boundary,
                          County.boundary)))
From  Woodlands, County
Where  County.name = 'Dyfed';
```



intersection returns the geometry resulting from the intersection of two geometry objects given as parameters.

# The filter operation for region containment



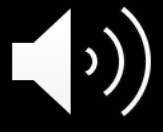



All objects (entire geometry) whose bounding rectangles intersect the query rectangle are retrieved, i.e. not strict containment or overlap

# Generic Spatial SQL

## Rectangular Region Containment query

## Filter version (a low cost operation)



```
Select S.name, S.geometry
From Settlements S
Where Filter(S.geometry,
                Rectangle(xmin,ymin,xmax,ymax);
```



**`S.geometry`** represents the geometry of settlements

**`Rectangle`** defines the geometry of a rectangular query region

Here the settlement geometry is tested for intersection with a rectangular query region.

If the settlements are area features, in query they would be approximated by bounding rectangles (see figure)

# Spatio-Textual Indexing: Geoparsing 1



## Toponym Recognition



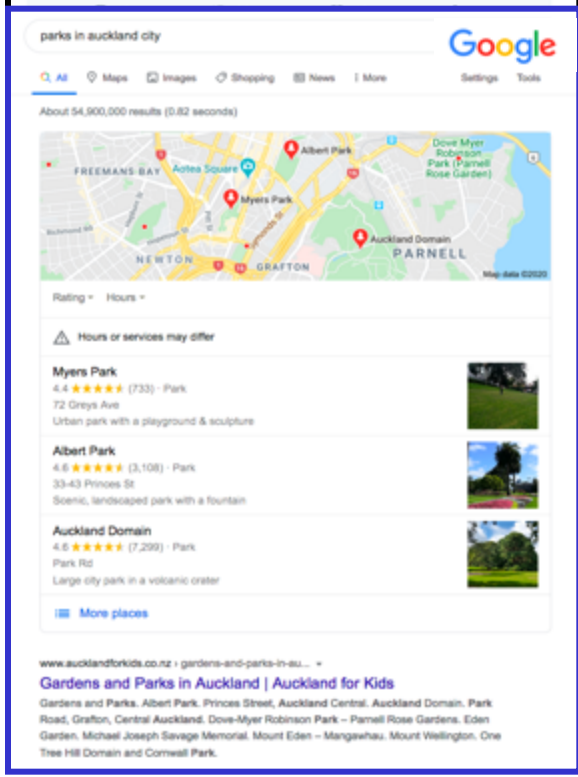


Chris Jones
Cardiff University, UK

# Geographical Information Retrieval

*Retrieval of geo-information from text documents* (or weakly-structured sources)

- Social media messages / posts (. g. current events; disasters; health of people; public gatherings; wildlife; etc..
- Web pages: many forms of geographic info
  - Many queries are geographical as indicated by a place name
  - Relevant pages usually found, as normal, by matching the place name to document text
  - Some web content is explicitly spatially indexed (with coordinates) usually "points of interest" (POI), e.g. restaurants, shops, banks, cinemas, hotels, museums.….
- Digital libraries, scientific documents….

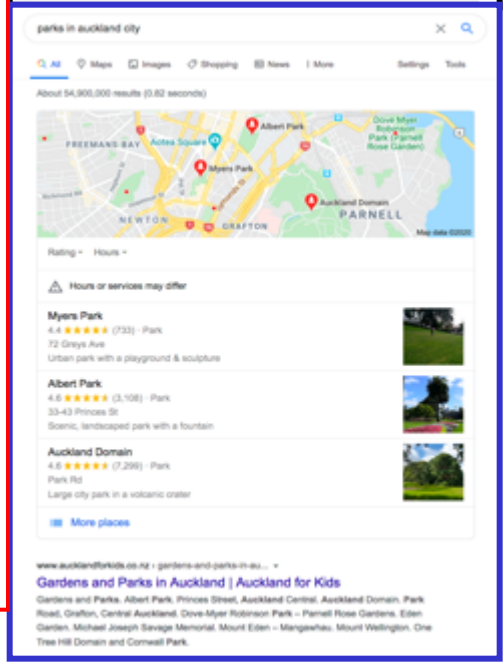



# How is text information geographically referenced?

- Some social media have coordinates from GPS
- POIs are associated with explicit coordinates attached by search engine
- Otherwise place names in documents indicate geographical locations
- The place names in documents allow them them to be indexed spatially



All of **Auckland**'s volcanoes come from one magma source. Underlying **Auckland** is a diffuse pool of magma that occasionally finds its way to the surface. Unlike a 'classic' volcano – such as **Mt Taranaki** or **Mt Ngāuruhoe** with a single vent through the crust – in Auckland, the magma


www.sciencelearn.org.nz/

# Some challenges in GIR

- Finding relevant documents for a geospatial query
- Indexing documents with respect to text and space
- Recognition of geographic references in text documents (toponym recognition)
- Disambiguation and geocoding of toponyms (toponym resolution) for spatial indexing of documents
- Extracting structured geospatial facts / events from natural language
- Understanding vague spatial language
  - Vague / vernacular place names (*city centre, the south,*)
  - Vague / imprecise language (*near, beside, on, north of, in front of, between…)*

# Geoparsing

*Identifying and geocoding geographic references in text (place names, addresses…)*

- Toponym recognition:
  - Many words for place names are also used in a non-geographic sense (people's names, names of things) = non-geo - geo ambiguity
- Toponym resolution
  - Many different places are referred to by same name = geo - geo ambiguity
  - Need to find the correct coordinates for a toponym. Usually use gazetteers (lists of place names + coordinates + feature type)

# Ambiguity in toponym recognition

Jack Hagel, Staff Writer
Redevelopment of the **World Trade Center** site in **New York** is getting some input from a **Raleigh** real-estate maven.

York Properties President Smedes York was chairman of an Urban Land Institute panel at the **World Trade Center** and **Lower Manhattan** Summit last month.

The group heard presentations on how the area surrounding the site of the Sept. 11, 2001, terrorist attacks should be redeveloped. It suggested retail be a central focus for developers. The institute will issue a report based on the recommendations before the end of the year.

York was chairman of the Urban Land Institute, a **Washington** nonprofit organization, from 1989 to 1991. His dad, J.W. "Willie" York, joined the Urban Land Institute in 1947. That's where he met J.C. Nichols, the developer of **Country Club Plaza** in **Kansas City, Mo.** - the center that inspired Willie York to build **Raleigh**'s **Cameron Village**, the Southeast's first shopping center.

Jack Hagel, Staff Writer
Redevelopment of the World Trade Center site in New York is getting some input from a Raleigh real-estate maven.

York Properties President Smedes York was chairman of an Urban Land Institute panel at the World Trade Center and Lower Manhattan Summit last month.

The group heard presentations on how the area surrounding the site of the Sept. 11, 2001, terrorist attacks should be redeveloped. It suggested retail be a central focus for developers. The institute will issue a report based on the recommendations before the end of the year.

York was chairman of the Urban Land Institute, a Washington nonprofit organization, from 1989 to 1991. His dad, J.W. "Willie" York, joined the Urban Land Institute in 1947. That's where he met J.C. Nichols, the developer of Country Club Plaza in Kansas City, Mo. -- the center that inspired Willie York to build Raleigh's Cameron Village, the Southeast's first shopping center.

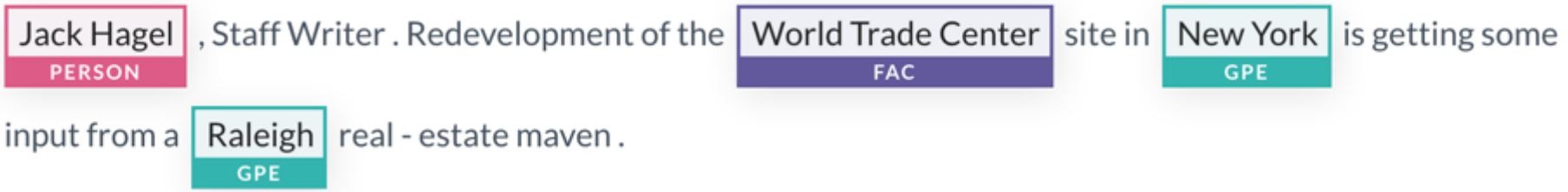


The words in red are toponyms that need to be recognized. On the right are all words that can be used as toponyms – but some of them (coloured magenta) are not actually toponyms in this text. This is non-geo - geo ambiguity

# Named entity recognition

- The task of recognizing place names in text is part of the natural language processing method of Named Entity Recognition (NER)
- NER software distinguish entities such as people, location, organisations, dates, numbers

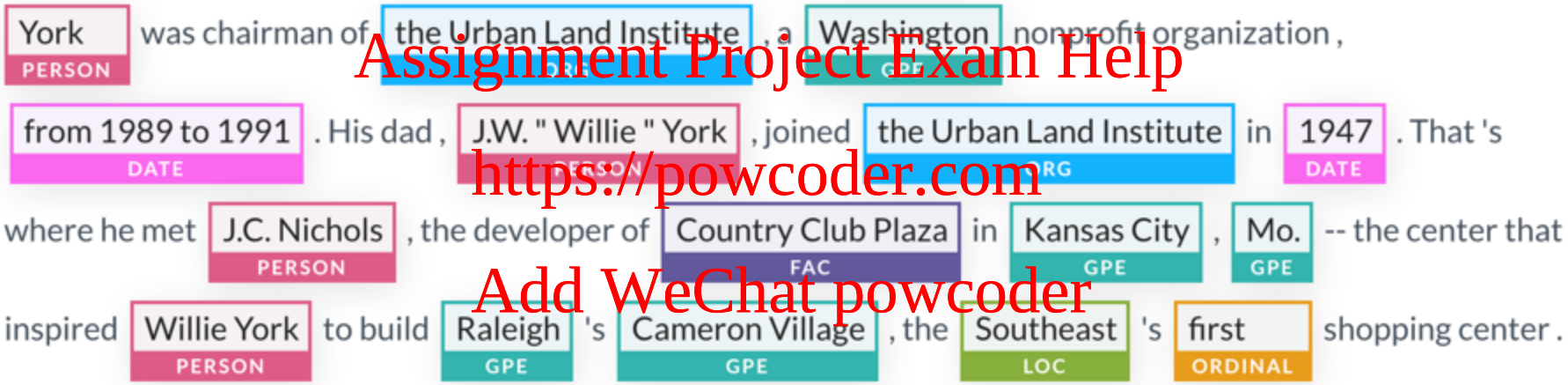





GPE = country, city or state;
FAC = built structures: (Buildings, airports, highways, bridges etc)

# NER with the Allen NLP software

Demo at
https://demo.allennlp.org/named-entity-recognition/






ORG = organization
GPE = country, city or state;
FAC = built structures: (Buildings, airports, highways, bridges etc)
LOC = other locations including natural features

# What could distinguish a place name from other entities?

For example, ***the presence or the absence*** of :

- Initial capital letter
- Preceded by a spatial preposition / suffix such as in, near
- Can be found in a gazetteer of place names
- Preceded by initial letters
- Preceded by "Mr", "Miss", "Dr"
- A sequence of numeric digits
- Can be found in a list of first names
- Is preceded or followed by a word that refers to type of place (city, lake, river…)



Which of the above apply to place names in English?

# Methods of automating toponym recognition

- Use machine learning (ML) with features (items of evidence) such as those listed in previous slide.

- Classifiers, such as Decision Tree (e.g. Random Forest); Support Vector Machine (SVM) or a deep learning method (neural network).



- Input features could also include the actual words to the left and the right of the word or phrase to be categorized (e.g. 3 words to left and 3 to right).





Club Plaza in *Kansas City*, Mo. the center

-*Kansas City* had been identified as a noun phrase or proper noun and hence a potential place name

ML Features either a one-hot vector of all words in collection – set elements to 1 if word in the target sentence. Or similar vector of tf-idf (text frequency / inverse document frequency) values

# Word Embeddings as features for machine learning

- Rather than the actual words, can use *word embeddings* of neighbouring words as input features

**Club Plaza in *Kansas City*, Mo. the center**



- Represent a word as a multi-dimensional vector
- Each dimension represents an aspect of meaning of the word.





- Vector obtained with dimensionality reduction techniques based on the co-occurrence of target word and other words commonly used in the context of the word.
- Words with similar meaning are closer in the vector space
- word embedding methods, e.g. GloVe, Word2Vec, FastText, Bert


Image: Zenun Kastrati WET: Word Embedding-Topic distribution vectors for MOOC video lectures dataset


Image: Sudeep Das: "Navigating themes in restaurant reviews with Word Mover's Distance"

# References


Purves R.S, P.D. Clough, C.B. Jones, M.M. Hall, M. Murdoch (2018) "Geographic Information Retrieval: Progress and challenges in spatial search of text". Foundations and Trends in Information Retrieval. Vol. 12, No. 2-3, pp 164-318.

F. Melo and B. Martins (2017). Automated geocoding of textual documents: A survey of current approaches. Transactions in GIS, 21(1):3–38.

J. L. Leidner and M. D. Lieberman (2011). Detecting geographical references in the form of place names and associated spatial natural language. SIGSPATIAL Special, 3(2):5–11.

Geonames Gazetteer - https://www.geonames.org

AllenNLP NER demo - https://demo.allennlp.org/named-entity-recognition/

Nominatum Geocoding API - https://nominatim.org/

GeoPy Python library for geocoding - https://geopy.readthedocs.io/en/1.22.0/

SpaCy guidance on NER (and other NLP) https://spacy.io/usage/linguistic-features

Stanford Named Enity Recognizer - https://nlp.stanford.edu/software/CRF-NER.html and https://corenlp.run/

# Spatio-Textual Indexing: Geoparsing 2



## Toponym Resolution



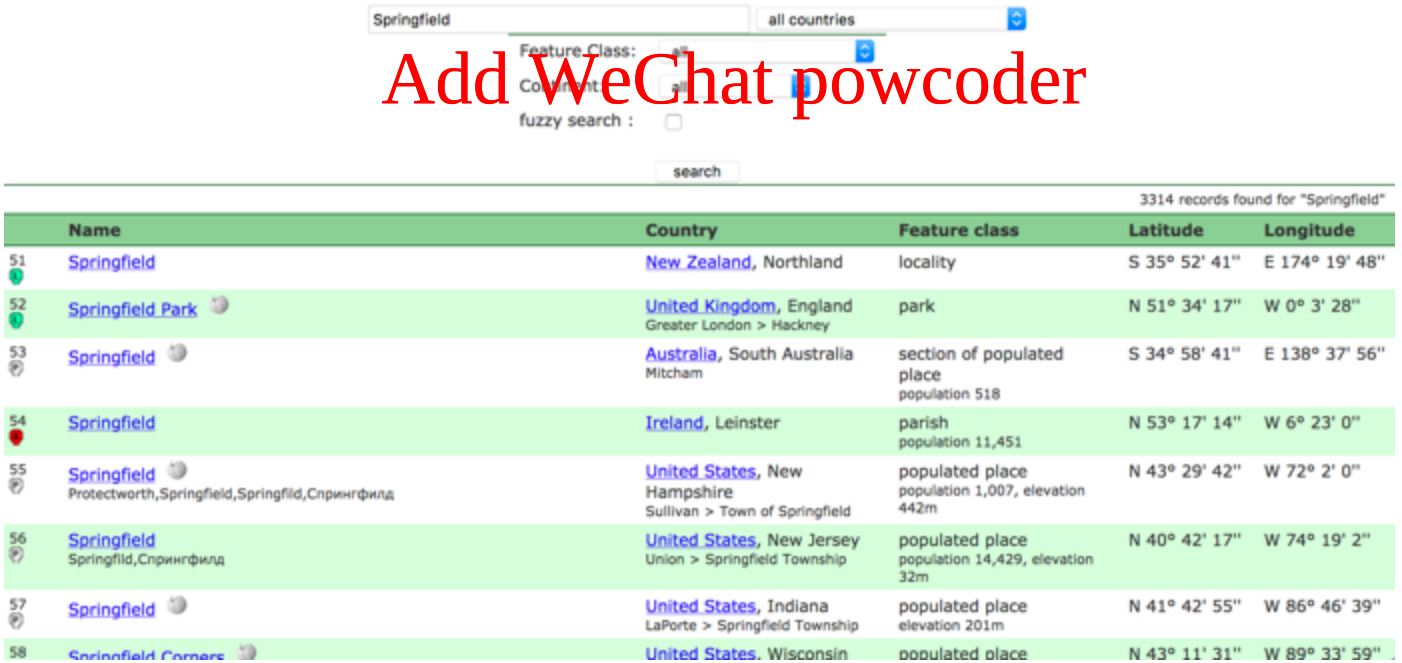


Chris Jones

# Toponym Resolution (Geocoding or georeferencing)

- Having identified a word as a probable place name, we need to find its coordinates
- Can look up the name in a **gazetteer** but in many cases there are several instances of the same name
- Geonames gazetteer has more than 100 "Springfield"s






| | Name | Country | Feature class | Latitude | Longitude |
|---|---|---|---|---|---|
| 51 | Springfield | New Zealand, Northland | locality | S 35° 52' 41" | E 174° 19' 48" |
| 52 | Springfield Park | United Kingdom, England<br>Greater London > Hackney | park | N 51° 34' 17" | W 0° 3' 28" |
| 53 | Springfield | Australia, South Australia<br>Mitcham | section of populated place<br>population 518 | S 34° 58' 41" | E 138° 37' 56" |
| 54 | Springfield | Ireland, Leinster | parish<br>population 11,451 | N 53° 17' 14" | W 6° 23' 0" |
| 55 | Springfield<br>Protectworth,Springfield,Springfild,Спрингфилд | United States, New Hampshire<br>Sullivan > Town of Springfield | populated place<br>population 1,007, elevation 442m | N 43° 29' 42" | W 72° 2' 0" |
| 56 | Springfield<br>Springfild,Спрингфилд | United States, New Jersey<br>Union > Springfield Township | populated place<br>population 14,429, elevation 32m | N 40° 42' 17" | W 74° 19' 2" |
| 57 | Springfield | United States, Indiana<br>LaPorte > Springfield Township | populated place<br>elevation 201m | N 41° 42' 55" | W 86° 46' 39" |
| 58 | Springfield Corners | United States, Wisconsin | populated place | N 43° 11' 31" | W 89° 33' 59" |

# Toponym geo-geo disambiguation

- Ambiguous names include:

  Raleigh, Kansas City.

Various strategies (see next slides)

- Several methods look for associations between names in same document

  Nearby places or places with geographical hierarchical relations e.g. New York and World Trade Center

  -and Manhattan and World Trade Center

or between names and other text: (language models)

Jack Hagel, Staff Writer
Redevelopment of the **World Trade Center** site in **New York** is getting some input from a **Raleigh** real-estate maven.

York Properties President Smedes York was chairman of an Urban Land Institute panel at the **World Trade Center** and **Lower Manhattan** Summit last month.

York was chairman of the Urban Land Institute, a **Washington** nonprofit organization, from 1989 to 1991. His dad, J.W. "Willie" York, joined the Urban Land Institute in 1947. That's where he met J.C. Nichols, the developer of **Country Club Plaza** in **Kansas City, Mo**. - the center that inspired Willie York to build **Raleigh**'s **Cameron Village**, the Southeast's first shopping center.

# Computing methods for toponym resolution

- Default sense – choose the biggest / best known instance of the place, e.g. New York in USA as opposed to the village in UK
- If an ambiguous name has a hierarchical relation with an unambiguous name in the document then choose the instance of the name that has that relation:



"Kansas City, Mo " implies the Kansas City in Missouri (Mo)

"Raleigh's Cameron Village" implies the Raleigh that contains Cameron Village which is in North Carolina

- If an instance of an ambiguous place is geographically close to instances of unambiguous places in the text, choose that instance (see slide on spatial minimality heuristic).
- Use language modelling methods : locations are represented by the words with which they are associated. See later slide.

# Disambiguation with Default Sense

- choose the biggest / best known instance of the place

Results from OpenStreetMap Nominatim

City New York, United States of America

State Boundary New York, United States of America

Village New York, Tyne and Wear, North East England, England, NE29 8EP, United Kingdom

Village New York, Caldwell County, Missouri, United States of America

Islet New York, Dubai, United Arab Emirates

Locality New York, Drohobych, Drohobych City Council, Lviv Oblast, 82106, Ukraine

Hamlet New York, Henderson County, Texas, 75770, United States of America

Look up place name in a gazetteer

– could choose instance of settlement/city that has largest population

It works here for *New York, Kansas City and Raleigh*



Jack Hagel, Staff Writer
Redevelopment of the **World Trade Center** site in **New York** is getting some input from a **Raleigh** real-estate maven.

York Properties President Smedes York was chairman of an Urban Land Institute panel at the **World Trade Center** and **Lower Manhattan** Summit last month.

York was chairman of the Urban Land Institute, a **Washington** nonprofit organization, from 1989 to 1991. His dad, J.W. "Willie" York, joined the Urban Land Institute in 1947. That's where he met J.C. Nichols, the developer of **Country Club Plaza** in **Kansas City, Mo**. - the center that inspired Willie York to build **Raleigh**'s **Cameron Village**, the Southeast's first shopping center.

# Disambiguation with Hierarchical Relations

If text includes a geographical parent or child of a candidate instance of place: – assign ambiguous toponym to that instance



Hierarchical relations in this text:





- Kansas City → Mo. [Mo. = Missouri]
- Cameron Village → Raleigh
- World Trade Center → New York
- Lower Manhattan → New York
- Country Club Plaza → Kansas City

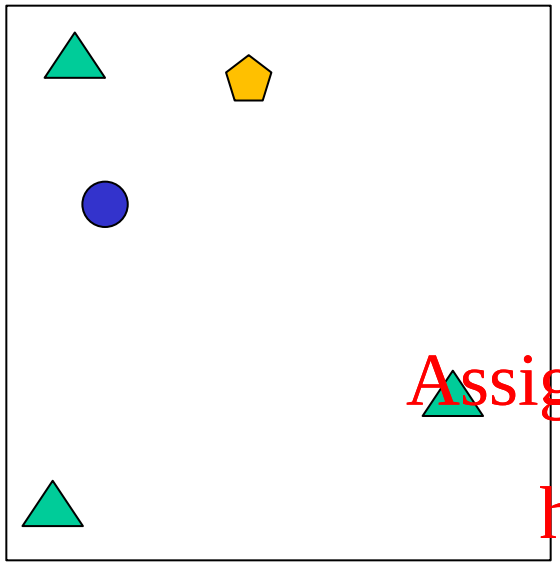
[ Raleigh → Southeast (vernacular)]

Jack Hagel, Staff Writer
Redevelopment of the **World Trade Center** site in **New York** is getting some input from a **Raleigh** real-estate maven.

York Properties President Smedes York was chairman of an Urban Land Institute panel at the **World Trade Center** and **Lower Manhattan** Summit last month.

York was chairman of the Urban Land Institute, a **Washington** nonprofit organization, from 1989 to 1991. His dad, J.W. "Willie" York, joined the Urban Land Institute in 1947. That's where he met J.C. Nichols, the developer of **Country Club Plaza** in **Kansas City, Mo**. - the center that inspired Willie York to build **Raleigh**'s **Cameron Village**, the **Southeast**'s first shopping center.

# Resolving ambiguity with spatial minimality



Toponyms with coordinates

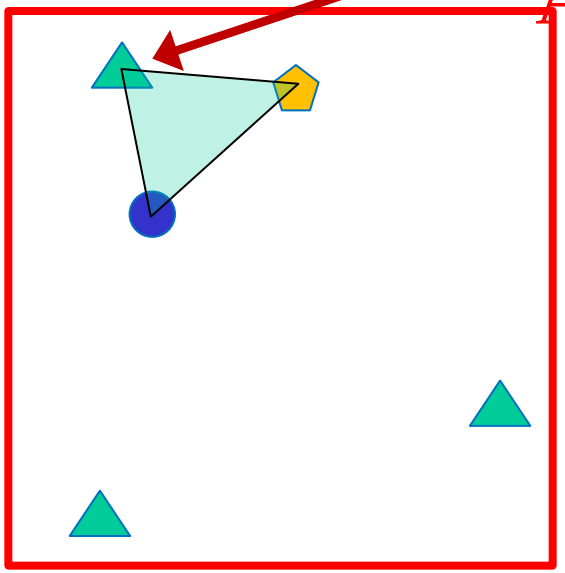
- Ambiguous – 3 candidate locations
- Not ambiguous – 1 location
- Not ambiguous – 1 location

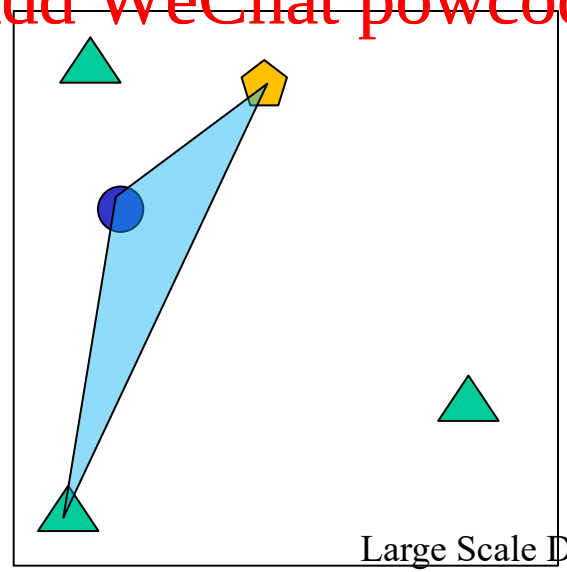


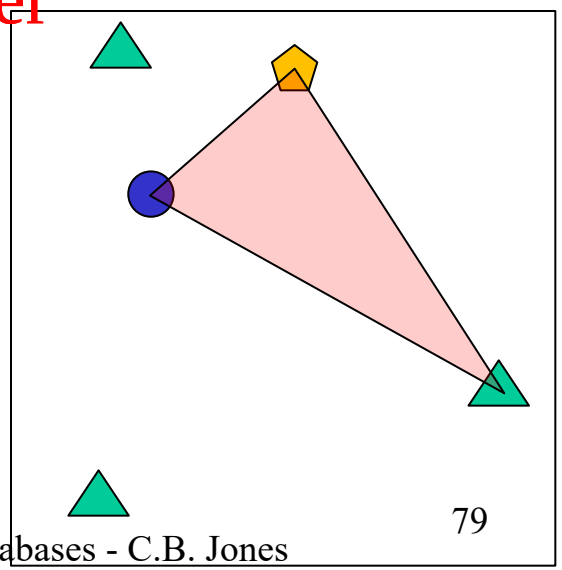
*For ambiguous toponym select candidate that has smallest polygon when clustered with instances of other names*











# Language modelling approach for toponym resolution

- Use all the associated words to resolve ambiguity – might or might not include place names
- E.g. *Albany, New Zealand* can be represented by the words in Wikipedia article that describes it (mentions Westfield shopping centre, Massey University etc).
- Albany, New York is associated with different words – hence a different language model
- Language model is a weighted set of words associated with toponym that is being modelled


**Albany, New Zealand**

**Albany** is one of the northernmost suburbs of the contiguous Auckland metropolitan area in New Zealand. It is located to the north of the Waitematā Harbour,

15 kilometres (9.3 mi) northwest of the Auckland city centre. The suburb is in the Albany ward, one of the thirteen administrative divisions of Auckland Council. One of the city's newest suburbs, it was until relatively recently a town in its own right, and still has a feeling of not being truly a part of the city, which lies predominantly to the south-east of it. Much of the land to the north of Albany is still semi-rural.

The Māori name for the area was Okahukura (literally, 'place of rainbows' or 'place of butterflies'). The town was originally known as Lucas Creek. By 1890 it was a fruit-growing area and in that year it was renamed 'Albany' after the fruit-growing district called 'Albany' in Australia, pronounced with a short 'a' as in *Albert*.[1][2] The name *Albany* derives from *Alba* (Gaelic for Scotland) and its Latinisation.

**City planning**

In 2005, there were plans to turn a major swath of Albany into a planned mini-urban centre, described as a "happy mix of businesses, hotels, shops, apartments, and entertainment (including) an environment of parks and lakes and of tree-lined streets, paths and cycleways linking to the new park-and-ride bus station and the rapid-busway lanes along the Northern Motorway to downtown Auckland", according to a newspaper report.[3] It would be home to 10,000 people.[3] All the flats wanted sound-proofed apartments against outside noise.[3] Initial plans called for hotels, library, municipal swimming pool as well as the headquarters for the North Shore City Council.[3] In some respects, development has proceeded accordingly, but the 2008–09 economic downturn has blunted some of this activity.

**Demographics**

The population was 3,057 in the 2013 Census, an increase of 888 from the 2006 Census.[4] There were 1,092 occupied dwellings in Albany in 2013,[4] and demographic makeup was 73% European, 5% Maori, 2% Pacific peoples, 22% Asian, 3% Middle Eastern/Latin American/African, and 1% other.[4] The median income of $32,600 was higher than for the Auckland Region of $29,600.[4] Unemployment in Albany was 7.0%, lower than the Auckland average of 8.1%.[4] 91% had internet access and 88% had cell phones.[4] Cars were prevalent.[4] A near majority (48%) were born overseas.[4] Ethnically, in keeping with the wider North Shore, Albany was predominantly Pakeha and Asian, and had a relatively high proportion of recent migrants from both elsewhere in New Zealand and overseas.

**Retail and commercial activity**

Albany has become, in some respects, a substantial shopping and retail zone within the northwestern Auckland area.

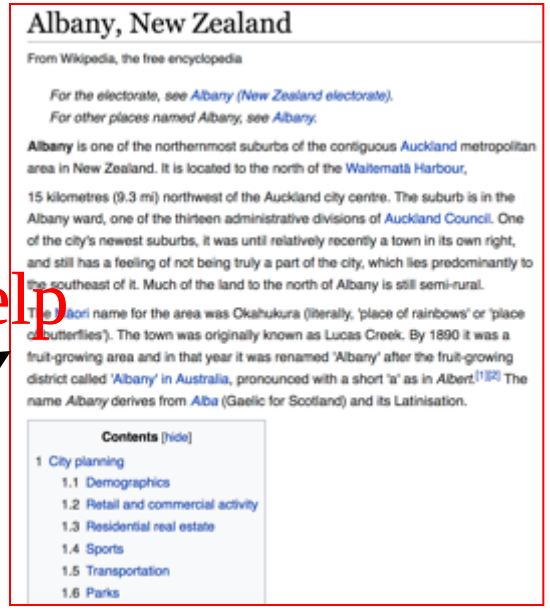
The area (the future 'Albany Town Centre') is fast-growing in terms of its population and the development of the built environment, following planning decisions and land sales made by central and local governments in the 1980s and 1990s. Through the 1990s industrial and retail areas were rapidly produced, predominantly owned and occupied by local and foreign corporate capital. A major shopping centre hub was opened in the late 1990s and has since expanded, with Westfield Albany becoming New Zealand's largest shopping centre. The so-called supermall opened in August 2007 on McKinnon Drive costing $210 million with 142 shops built by over 3500 workers, which features 1800 cinema seats and an indoor area of 7ha.[5] There is parking for 2300 vehicles.[5] Kmart, Farmers, and New World stores are anchors.[5] The mall claimed it provides "free space for community organisations for awareness and fundraising activities" but one volunteer claimed he was ejected from the premises while trying to raise money for veterans because of a dispute with mall management.[6]

# Disambiguation with language models

Given an ambiguous name in a document:

- Match the words in the document to the language models of each instance of ambiguous toponym
- Assign ambiguous toponym to the instance that has language model most similar to the text in the document.

Westfield's new **Albany** Mall is their first NZ greenfield development, situated on a 20.8ha site in **Albany**. It is one of the larger malls in the Southern Hemisphere



Albany, New Zealand

From Wikipedia, the free encyclopedia

*For the electorate, see Albany (New Zealand electorate).*
*For other places named Albany, see Albany.*

**Albany** is one of the northernmost suburbs of the contiguous Auckland metropolitan area in New Zealand. It is located to the north of the Waitematā Harbour,

15 kilometres (9.3 mi) northwest of the Auckland city centre. The suburb is in the Albany ward, one of the thirteen administrative divisions of Auckland Council. One of the city's newest suburbs, it was until relatively recently a town in its own right, and still has a feeling of not being truly a part of the city, which lies predominantly to the southeast of it. Much of the land to the north of Albany is still semi-rural.

The Māori name for the area was Okahukura (literally, 'place of rainbows' or 'place of butterflies'). The town was originally known as Lucas Creek. By 1890 it was a fruit-growing area and in that year it was renamed 'Albany' after the fruit-growing district called 'Albany' in Australia, pronounced with a short 'a' as in *Albert*.[1][2] The name *Albany* derives from *Alba* (Gaelic for Scotland) and its Latinisation.

Contents [hide]

1 City planning
1.1 Demographics
1.2 Retail and commercial activity
1.3 Residential real estate
1.4 Sports
1.5 Transportation
1.6 Parks

Albany, New York

From Wikipedia, the free encyclopedia

*For other uses, see Albany (disambiguation).*

**Albany** (/ˈɔːlbəni/ (listen) *AWL-be-nee*) is the capital of the U.S. state of New York and the seat and largest city of Albany County. Albany is located on the west bank of the Hudson River approximately 10 miles (16 km) south of its confluence with the Mohawk River and approximately 135 miles (220 km) north of New York City.[4]

Albany is known for its rich history, commerce, culture, architecture, and institutions of higher education. Albany constitutes the economic and cultural core of the Capital District of New York State, which comprises the Albany–Schenectady–Troy, NY Metropolitan Statistical Area, including the nearby cities and suburbs of Troy, Schenectady, and Saratoga Springs. With a 2013 Census-estimated population of 1.1 million[5] the Capital District is the third-most populous metropolitan region in the state. As of the 2010 census, the population of Albany was 97,856.

The area that later became Albany was settled by Dutch colonists who, in 1614, built Fort Nassau for fur trading and, in 1624, built Fort Orange. In 1664, the English took over the Dutch settlements, renaming the city as Albany, in honor of the then Duke of Albany, the future James II of England and James VII of Scotland. The city was officially chartered in 1686 under English rule. It became the capital of New York in 1797 following formation of the United States. Albany is one of the oldest surviving settlements of the original British thirteen colonies, and is the longest continuously chartered city in the United States.[10]

During the late 18th century and throughout most of the 19th, Albany was a center

# References


Purves R.S, P.D. Clough, C.B. Jones, M.M. Hall, M. Murdoch (2018) "Geographic Information Retrieval: Progress and challenges in spatial search of text". Foundations and Trends in Information Retrieval. Vol. 12, No. 2-3, pp 164-318.

F. Melo and B. Martins (2017). Automated geocoding of textual documents: A survey of current approaches. Transactions in GIS, 21(1): 3–38.

J. L. Leidner and M. D. Lieberman (2011). Detecting geographical references in the form of place names and associated spatial natural language. SIGSPATIAL Special, 3(2):5–11.

G. DeLozier, J. Baldridge, L. London (2015) Gazetteer-Independent Toponym Resolution Using Geographic Word Profiles. AAAI'15: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence: 2382-2388

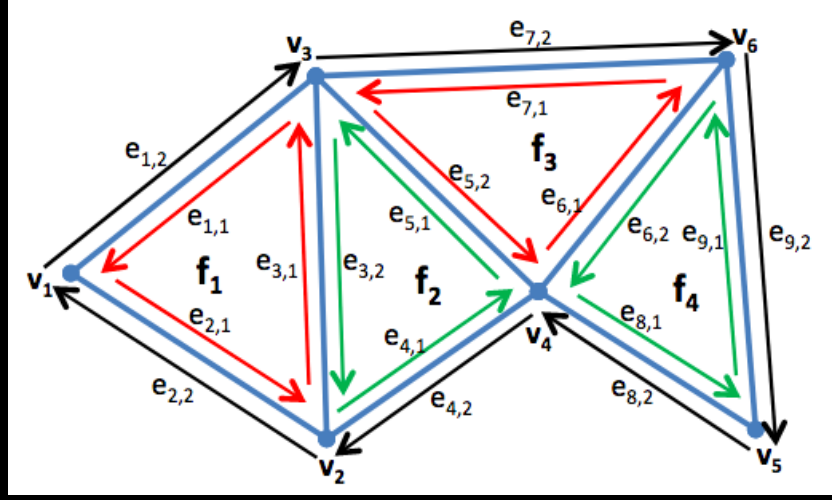
Geonames Gazetteer - https://www.geonames.org

AllenNLP NER demo - https://demo.allennlp.org/named-entity-recognition/

Nominatum Geocoding API - https://nominatim.org/

GeoPy Python library for geocoding - https://geopy.readthedocs.io/en/1.22.0/

SpaCy guidance on NER (and other NLP) - https://spacy.io/usage/linguistic-features






# Large Scale Databases



## Topologically Structured Data



Chris Jones

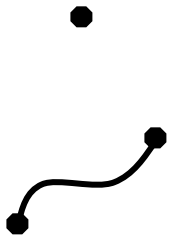
# Topologically Structured Vector Data

- records explicitly what geometry is connected to what (independent of distance and orientation), e.g..
  - adjacency between land parcels,
  - roads and rivers in networks



- many variations on what information is stored, depending on purpose
  - e.g. polygon map, network, triangulated network

# Topological primitives for 2D space



Node - isolated point or terminus of arc
0-dimensional;

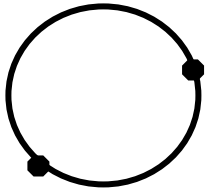


Arc (link) - (1-dimensional);

– linear feature bounded by two nodes;

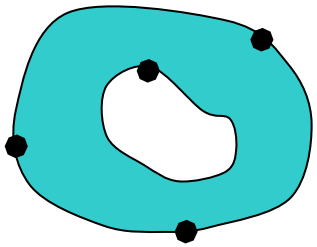
Directed arc (1-dimensional);

– has direction



Ring (1-dimensional);

– circuit of one or more arcs



Face (2-dimensional)

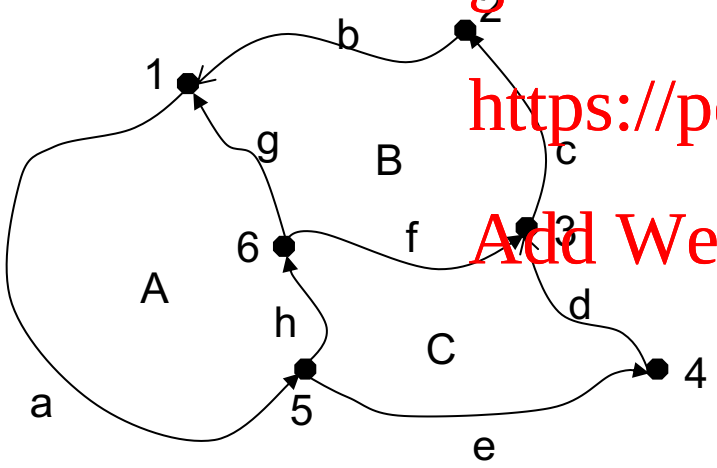
– region bounded by rings (one outer and 0 or more inner rings, i.e. holes)

# Polygon Map

could be used, for example, to record administrative boundaries or land use categories



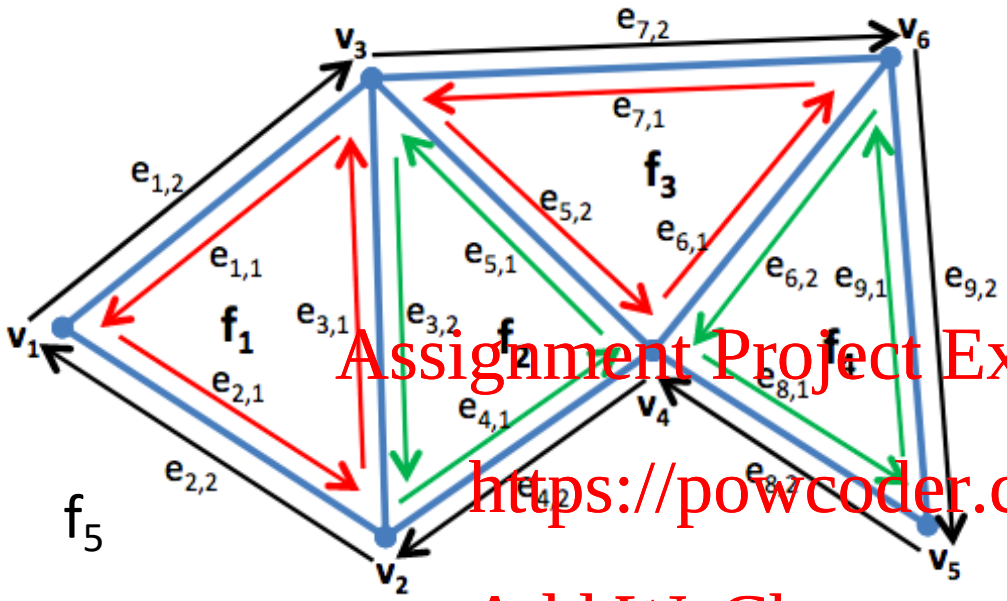
| Polygon (face) | Bounding arcs |
|---|---|
| A | a h g |
| B | f c b -g |
| C | e d -f -h |




| Arc | Start Node | End Node | Left face | Right face |
|---|---|---|---|---|
| a | 1 | 5 | A | |
| b | 2 | 1 | B | |
| c | 3 | 2 | B | |
| d | 4 | 3 | C | |
| e | 5 | 4 | C | |
| f | 6 | 3 | B | C |
| g | 6 | 1 | A | B |
| h | 5 | 6 | A | C |

*A simple approach to representing the relations. Can be more complex - see next slide*

# Doubly Connected Edge List (DCEL)




| Vertex | Coordinates (approx.) | Incident Edge |
|---|---|---|
| $v_1$ | 0,3 | $e_{1,2}$ |
| $v_2$ | 3,0 | $e_{2,2}$ |
| $v_3$ | 3,5 | $e_{7,2}$ |
| $v_4$ | 6,3 | $e_{4,2}$ |
| .... | .... | .... |



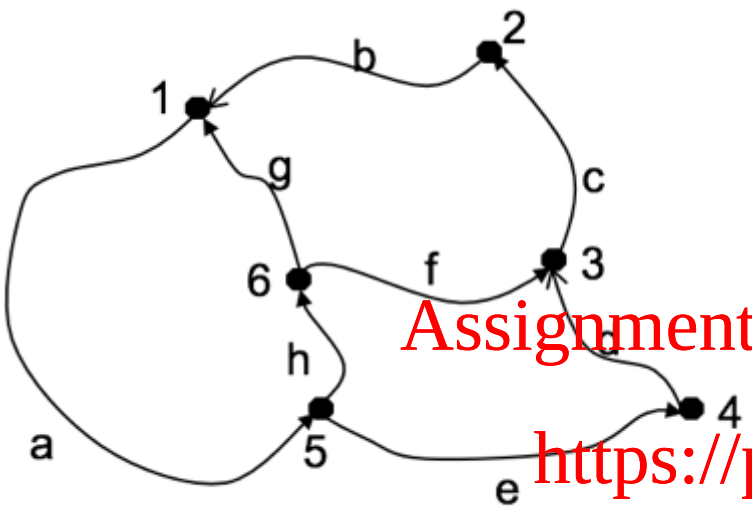
http://www.cs.sfu.ca/~binay/813.2011/DCEL.pdf

| Half-edge | Origin | Twin | Incident Face | Next | Previous |
|---|---|---|---|---|---|
| $e_{3,1}$ | $v_2$ | $e_{3,2}$ | $f_1$ | $e_{1,1}$ | $e_{2,1}$ |
| $e_{3,2}$ | $v_3$ | $e_{3,1}$ | $f_2$ | $e_{4,1}$ | $e_{5,1}$ |
| $e_{4,1}$ | $v_2$ | $e_{4,2}$ | $f_2$ | $e_{5,1}$ | $e_{3,2}$ |
| $e_{4,2}$ | $v_4$ | $e_{4,1}$ | $f_5$ | $e_{2,2}$ | $e_{8,2}$ |
| .... | .... | .... | .... | .... | .... |

| Face | Outer Component | Inner Components |
|---|---|---|
| $f_1$ | $e_{1,1}$ | nil |
| $f_2$ | $e_{4,1}$ | nil |
| $f_3$ | $e_{5,2}$ | nil |
| $f_4$ | $e_{6,2}$ | nil |
| $f_5$ | nil | $e_{1,2}$ |

M. de Berg, M. van Kreveld, M. Overmars and O. Scharzkopf, Computational Geometry, 2000, Springer. [ Diagram: http://www.cs.sfu.ca/~binay/813.2011/DCEL.pdf ]

# Linear Network






Node Table

| Node id | connected nodes (arcs) |
|---|---|
| 1 | 2(b) 5(a) 6(g) |
| 2 | 3(c) 1(b) |
| 3 | 4(d) 2(c) 6 (f) |
| 4 | 5(e) 3(d) |
| 5 | 4(e) 6(h) 1(a) |
| 6 | 3(f) 1(g) 5(h) |

For finding routes and shortest paths

*link (arc)* defined by

link id.; start node; end node

*node* defined by

node id.;

ordered list of connected nodes or of connected links.

Links may be accompanied by cost / impedance values such as ?, and by direction restrictions

# Large Scale Databases



# Spatial Databases: GIS Applications

Chris Jones

# Who uses GIS?

GIS are used by range of public, commercial and industrial services as well as private individuals – you and me - who want to find out about places and things that are in geographic space.



In practice they are used by anybody who needs access to geographical information

# Topographic and Land Survey Organisations

Provide base mapping for wide range of applications

- major users are local government and utilities




Types of information represented

- buildings; land parcels; administrative boundaries; rivers, coastline, lakes; spot heights, contours, DEM grids; roads, railways, canals; forests, woodlands; tourist facilities

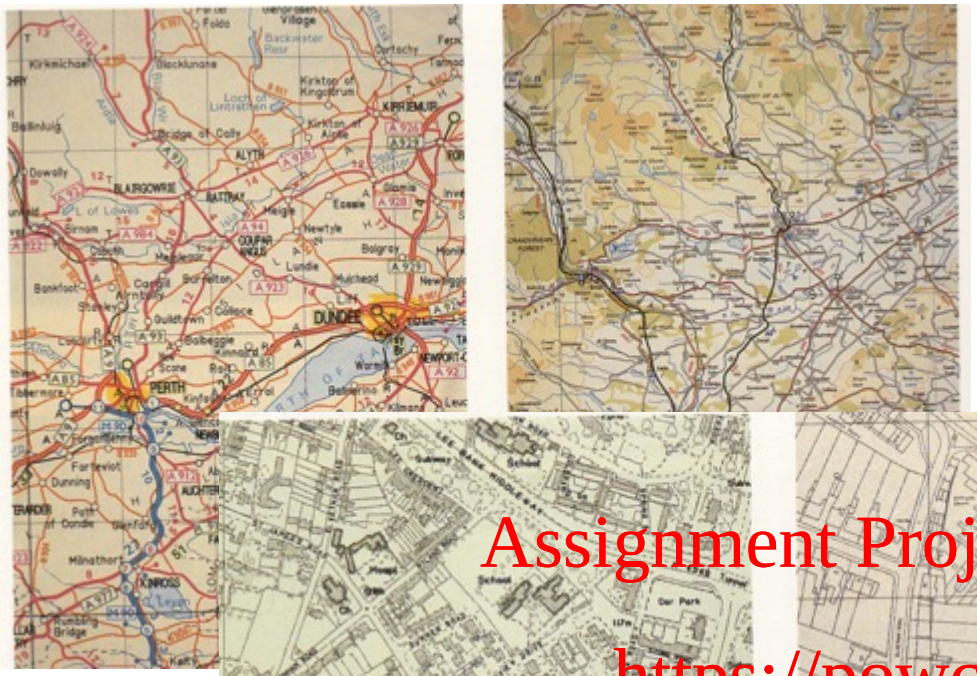





Example organisations:

- Ordnance Survey

  Vector Digital maps (1:1250, 1:2500, 1:10,000, 1:250000, 1:625000 ) + raster maps; terrain models
- AA - Vector and raster products
- Harper Collins/Bartholomews
- USGS (United States Geological Survey);
- US Bureau of Census: TIGER/Line files (urban)

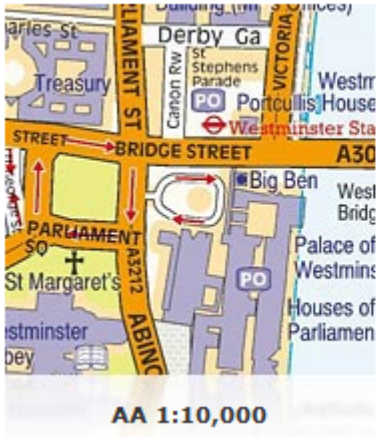


# Ordnance Survey Digital Maps at scales

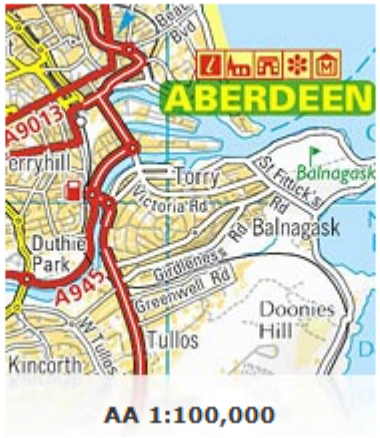
1:625,000; 1:250,000

1:10,000, 1:1250 + Terrain data

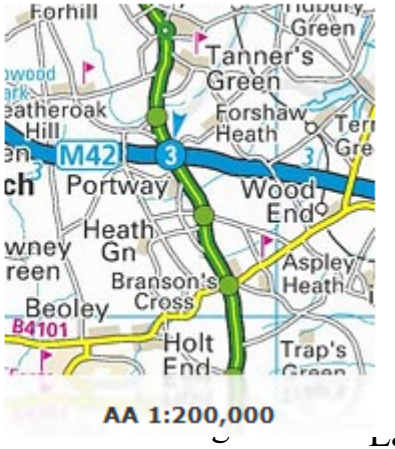





AA 1:10,000




AA 1:100,000




AA 1:200,000

AA Digital map data at different scales

# Topographic and Land Survey Organisations

## Example GIS Activities

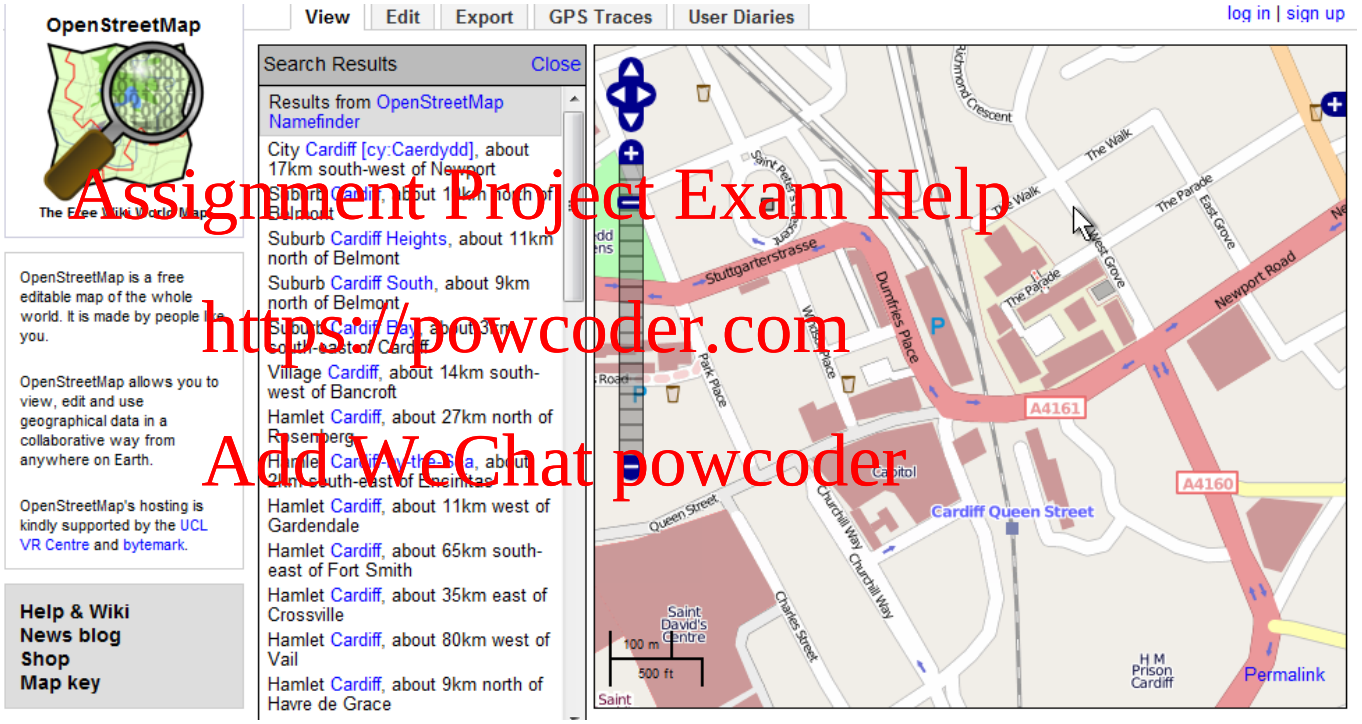
- Update existing commercial map products to reflect changes
- Create maps – for publication and to order (could require "generalisation")
- Create a detailed terrain model of a proposed development area



# Web 2.0 sources of geo-base data

**Open Street Map (www.openstreetmap.org)**

User-contributed map data from GPS and aerial photography





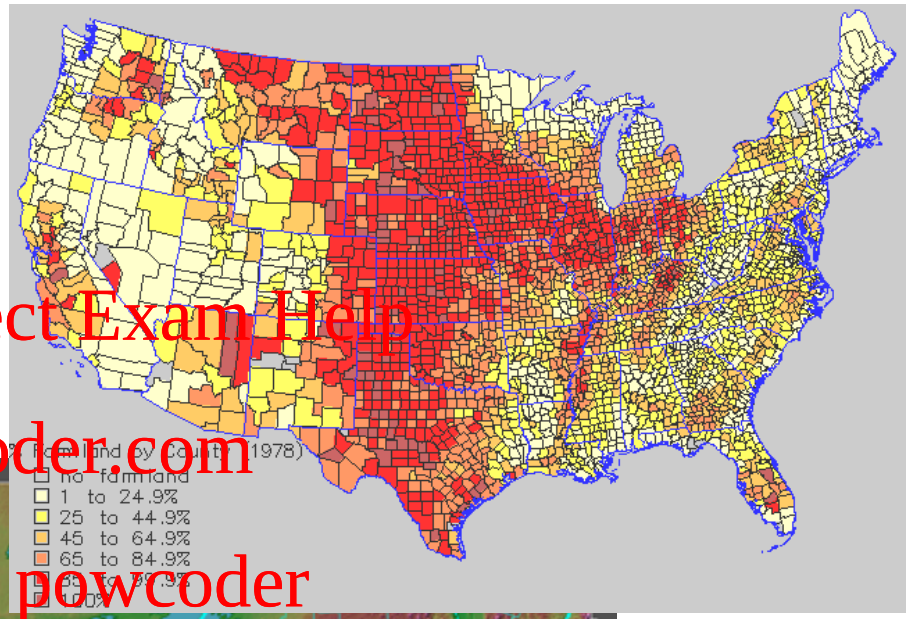

**GeoNames Gazetteer (www.geonames.org)**

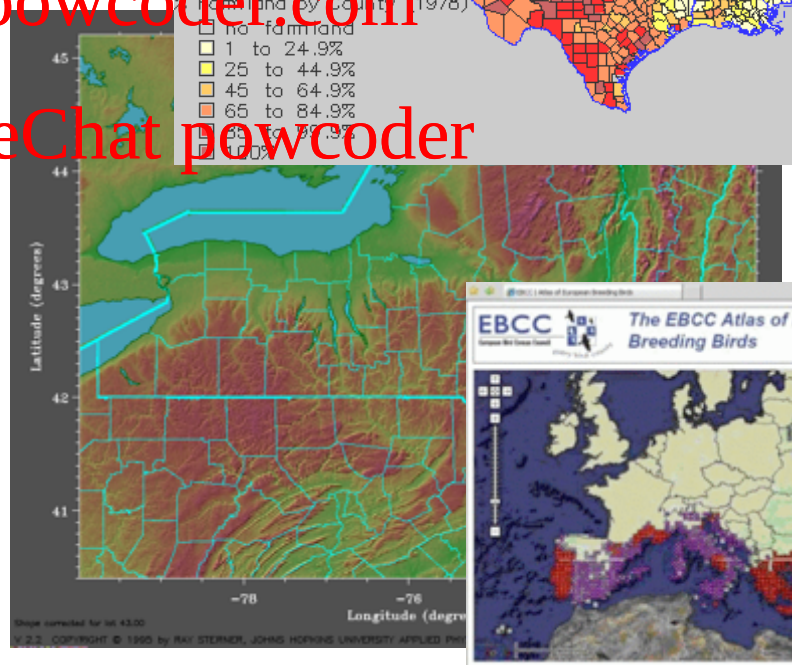
User-contributed information on place names, alternative names, country, coordinates (footprint), feature type

# Environmental / Geoscientific Survey Agencies

- Uses of GIS include:
  - creating databases of sample data and classified data
  - interpretation, modelling and analysis of sample data
  - creation of maps and reports







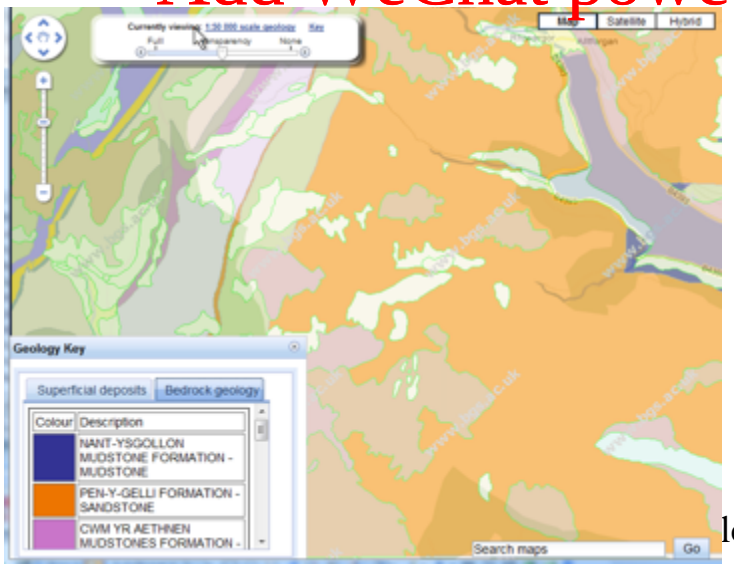


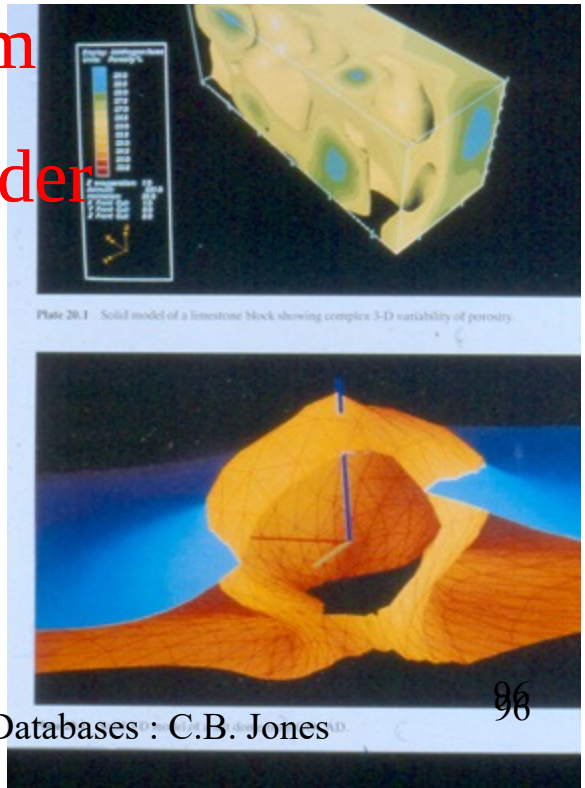
Large Scale Databases : C.B. Jones

# Environmental GIS

- Soils
  - erosion susceptibility, crop suitability
- Geology
  - outcrop maps; earthquake events; 3D models of subsurface from borehole, seismic, magnetic and resistivity studies



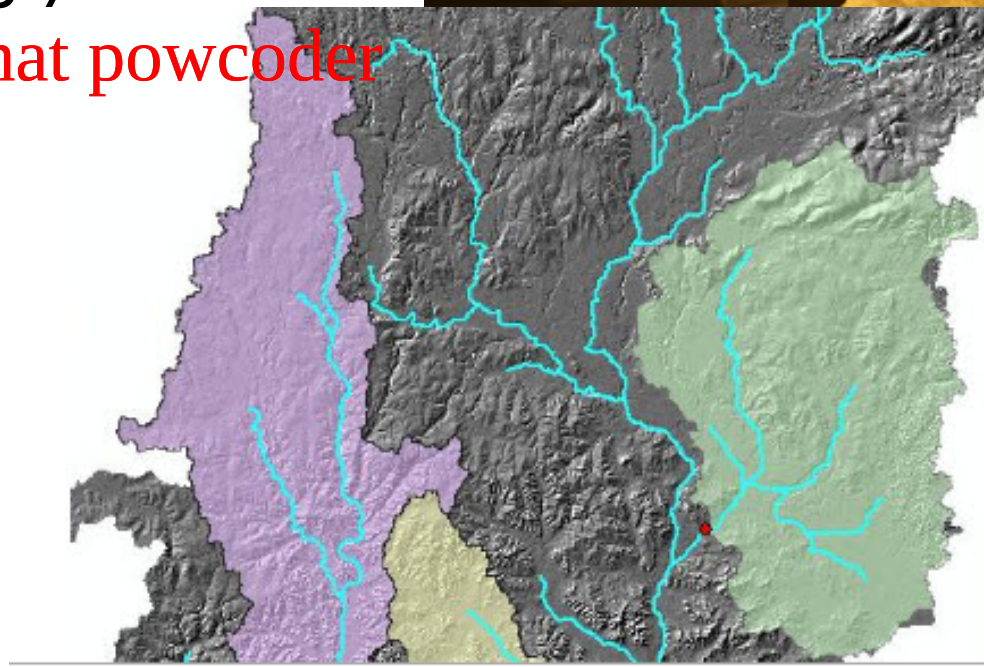
# Ecology

- identify environmentally sensitive areas;
- relationships between flora and fauna and environmental factors




# Hydrology (Rivers / Lakes / Coastal Zones)

- create valid terrain models
- analyses: e.g. drainage basins, runoff, pollutant monitoring



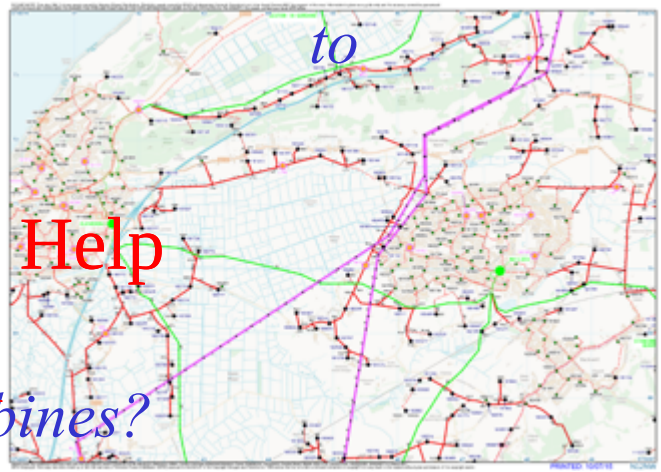
# Government Administration and Planning

- Data on property type, ownership, valuations, transactions;
  - *which properties do we own and what is their value?*
- Maintenance of roads, buildings, parks
- Example analyses:
  - development plans (housing, industry, retailing, leisure, transport, education, health...)
    - *where is a good place to create a new industrial site?*
  - deprivation / needs
  - emergency response strategies
    - *which properties are most at risk if the chemical plant emits toxic gases?*

# Utilities / Facilities Management

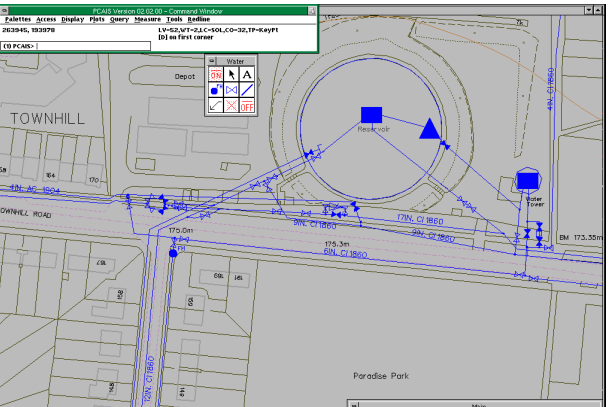
gas, electricity, water, sewers, telecomms

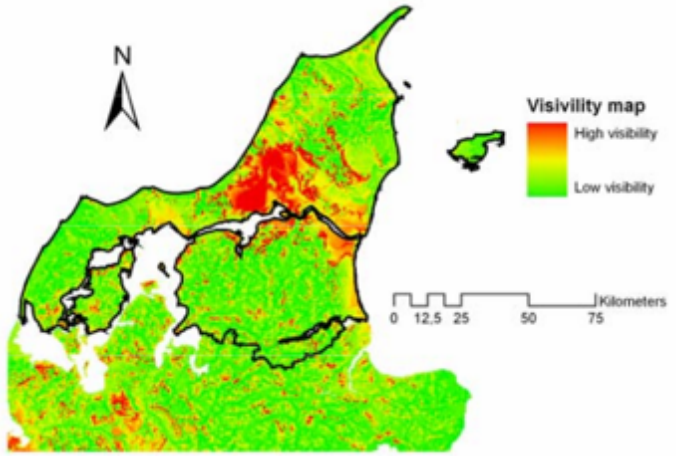
- databases for maintenance, customers, valuation
  - *where should we dig in Mary Street to find the source of the gas leak?*





- Example analyses :
  - *where are the best sites to locate wind turbines?*
  - *...which properties will be able to see them?*




Large Scale D

# Navigation


- Route maps
  - Static / mobile
- Driving instructions
- Analyses of optimal routes (various criteria)



*What requirements for structuring the data?*

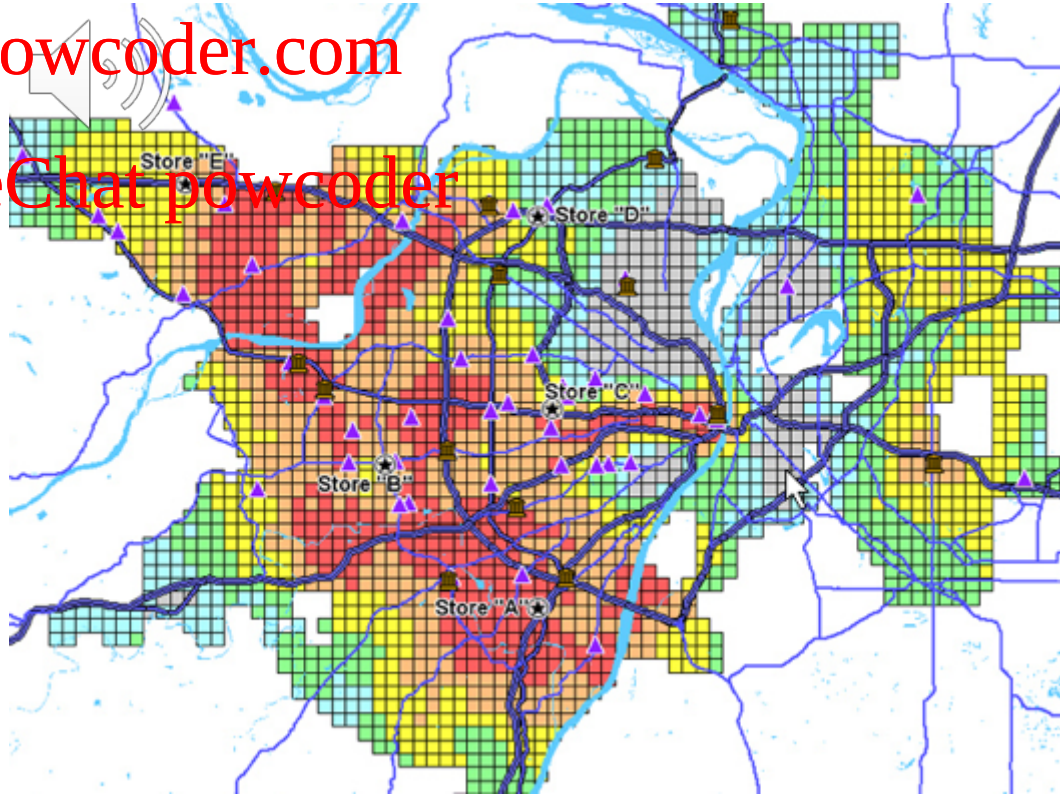





# Marketing / Retailing

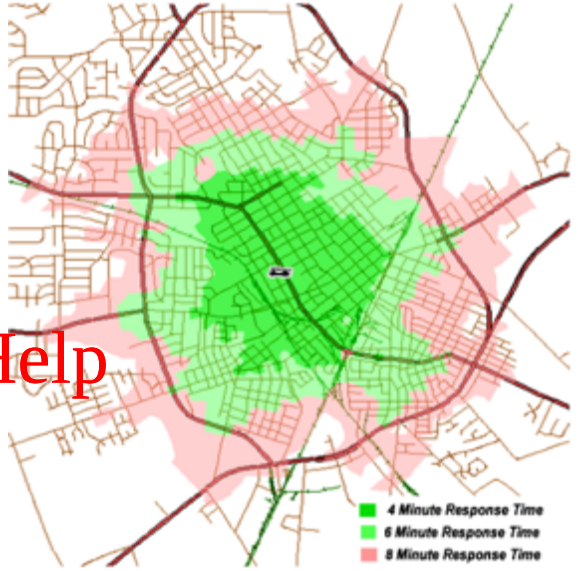
- analysis: store siting, targeting sales
  - *will it be profitable to build a Morrisons in Bogford?*
  - *to which postcodes should the Harden centre send advertising?*
- produce 'mosaics' classifying economic status

*Where does the data come from?*



# Healthcare

- analyses:
  - assessment of needs
  - epidemiological studies
    - identification of clusters
    - correlation with environmental, and socio-economic factors
  - location of health centres / hospitals
- Data sources
  - census, topographic survey,
  - medical records from hospitals
  - GP records




# Crime and Law Enforcement

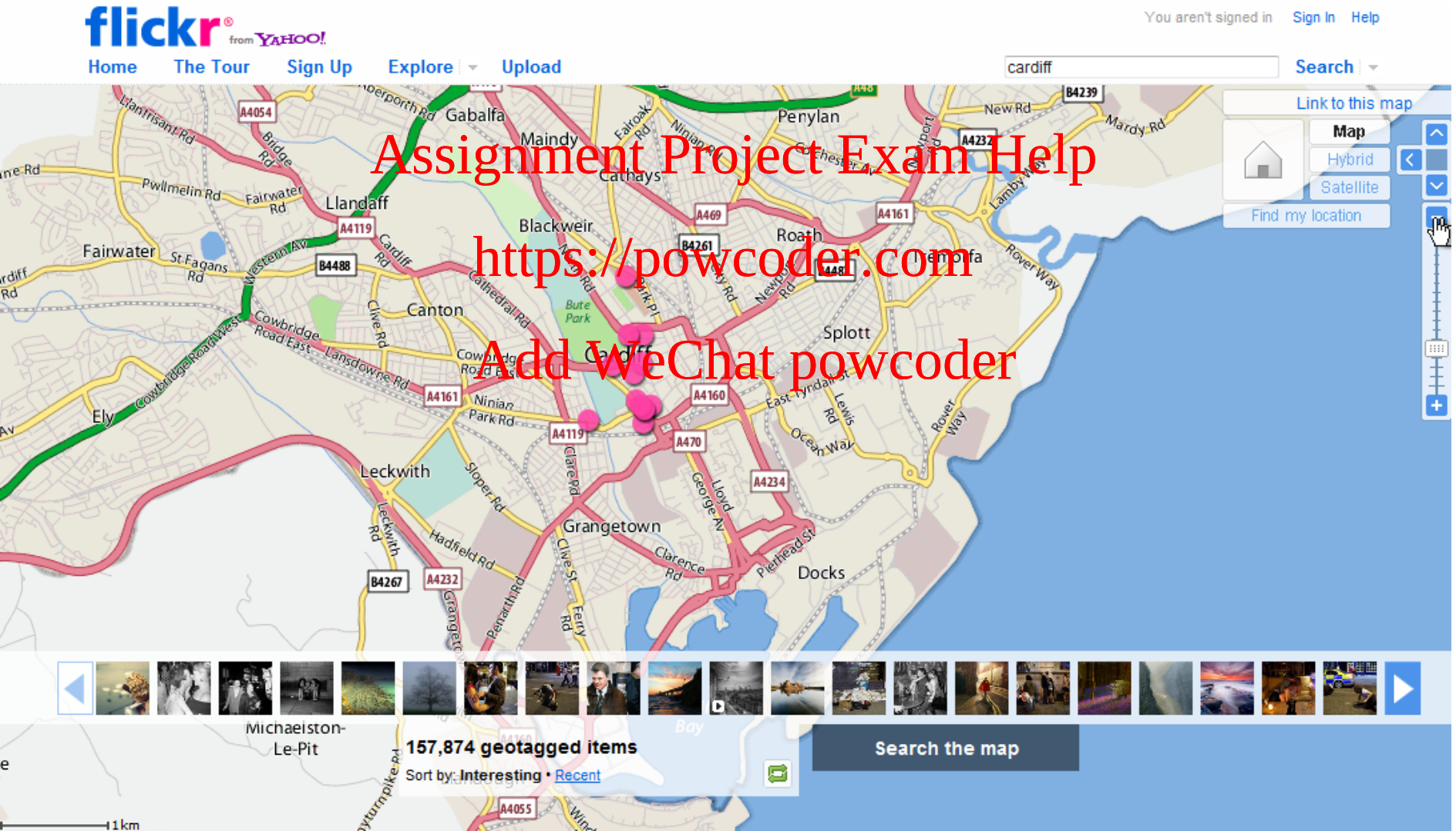
- Analyses : identify hotspots, in order to plan policing / detection activities;
- Correlation of crime with demographic factors
- Monitoring tagged individuals with GPS




https://www.police.uk/pu/your-area/south-wales-police/plasnewydd/?tab=CrimeMap

# Web 2.0 / Social Web

- E.g. Geo-tagging photos with location (e.g. Instagram, Flickr) – finding pictures about particular places



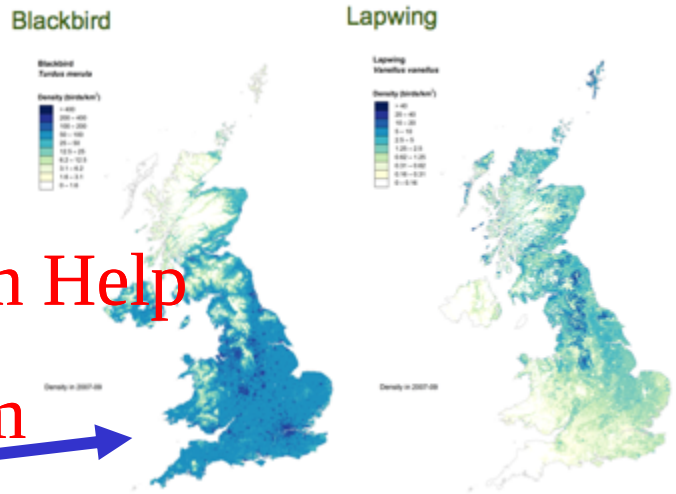
# Web 2.0 / Social Web thematically specific data

**"Volunteered geographic information"**

Add features to existing base maps using Goog
GoogleEarth etc

e.g.








- Bird species distribution (citizen science)



- Cycle maps (with OpenStreetMap) http://www.opencyclemap.org/


- Running and cycling : https://www.strava.com/

# Large Scale Databases
## Spatial Indexing


## Regular Grids

# Indexing Coordinate-based Spatial Data

Requirement

- fast access to spatial data relating to a location specified by coordinates in 2 dimensions

Problem

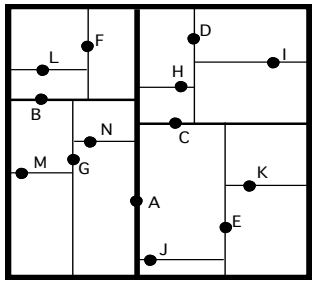


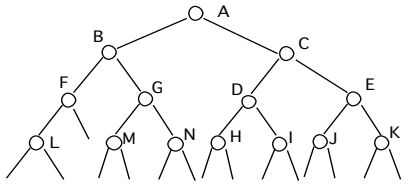
- Conventional (relational) databases provide one dimensional indexing methods (e.g. B-Trees) that allow fast access to data specified by a single key field such as a name or a unique identifier
- OK for indexing on place names, but geometric data has at least 2 dimensions
- Create new methods or adapt the existing ones
  - both done in practice

# 2 approaches to spatial indexing

- Space-directed
  - divide the region of interest into uniform cells and record what is in each cell (analogous to raster data model). Index the content of cells. Cell size may be adapted to data density.
  - e.g. regular grid, quadtree, grid file
- Object-directed
  - place bounding boxes around spatial objects and index the content of the boxes, e.g. R-Tree
  - Various other approaches, e.g. k-D tree in which the locations of points split space vertically and horizontally – but less suited to DBMS



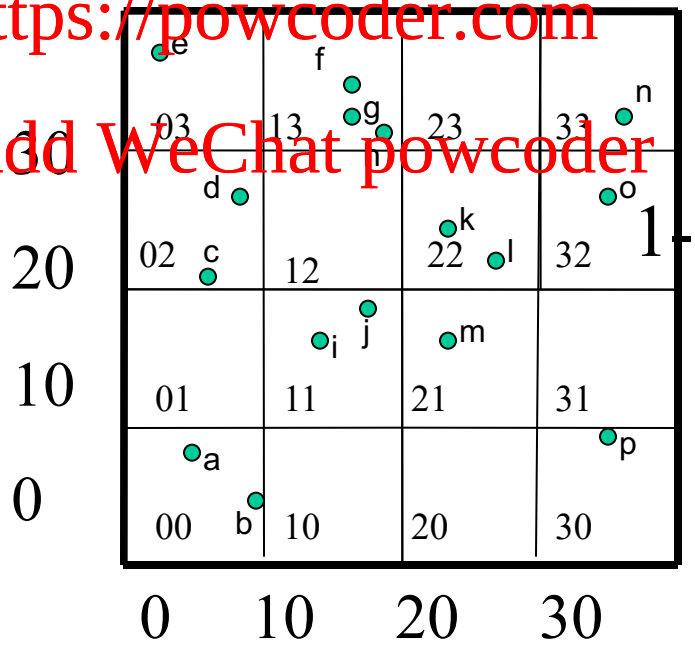


# Regular Grid

- Each cell is given unique numerical id i.e. a ***location code***, derived typically from the coordinates of its lower left corner
- Entries in each index record refers to geometry in corresponding cell
- **Use location : cell_id** for method (B-tree) access geometry





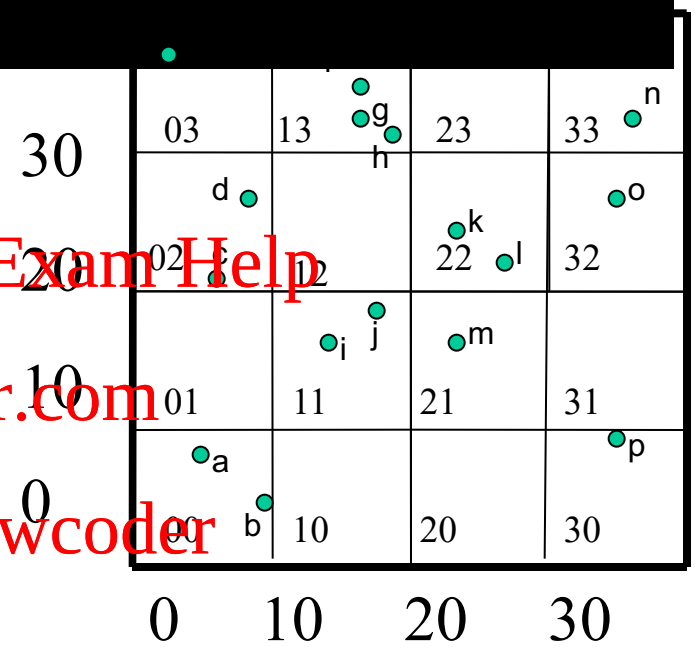

1-D index

| Cell_Id | Geometry_Id |
|---|---|
| 00 | a, b |
| 01 | |
| 02 | c, d |
| 03 | e |
| 10 | f, g, h |
| 11 | I, j |
| 12 | |
| 13 | |
| 20 | |
| 21 | m |
| ..... | |

**code as the key** to id data

# Location Codes: x, y concatenation

- One way to create location codes is to concatenate the x and the y coordinates
  - alternative is bit interleaving (see Quadtree Lecture)
- Allocate a fixed length to the x and y coordinate values.





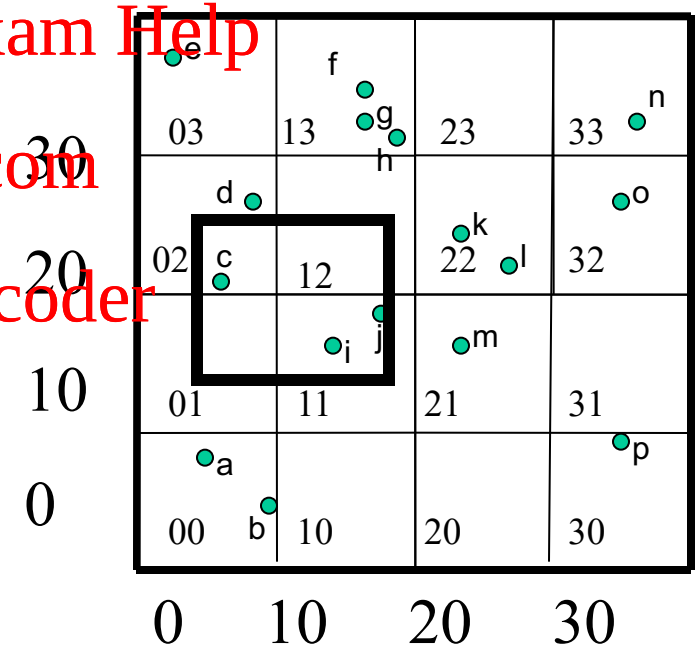

In example, concatenation involves initially dividing the coordinates by grid size – as they are always multiples of 10

More generally: set fixed length to x and y and pad to left with zeros, before concatenation.

# Retrieving data inside a rectangular window from regular grid indexed data

- Given a search window, calculate which cells it overlaps. In example : 01,02,11,12
- Retrieve corresponding records from index to find candidate spatial objects (***filter stage***).
- Might need to perform a second access to the file(s) containing the geometric data.
- Omit geometry outside window (***refinement stage***)






- To insert data, determine containing cells and add data to indexed records

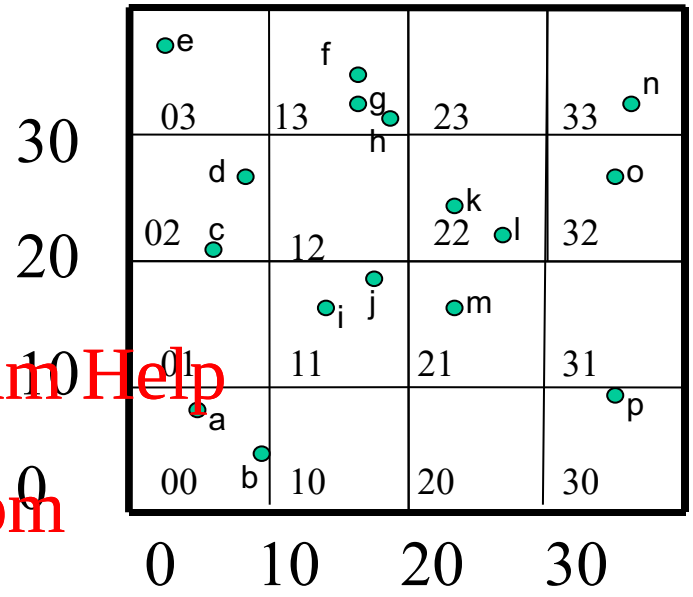
# Large Scale Databases
## Spatial Indexing



## Quadtrees

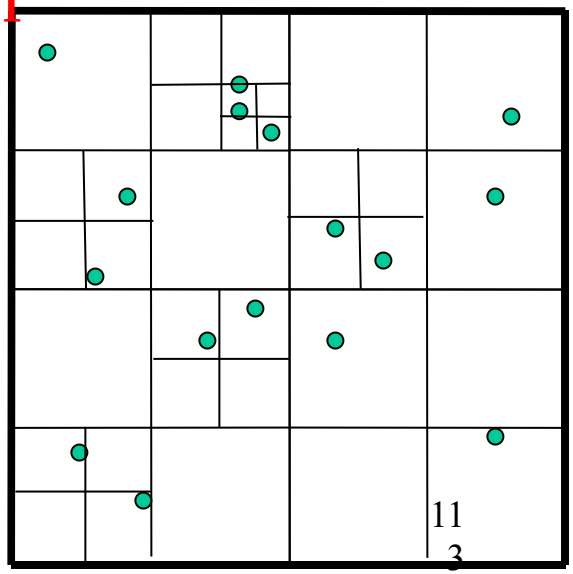
# Quadtree Indexing

- Issues with regular grid
  - *Data per cell?*
  - *Consequences for storage and processing?*



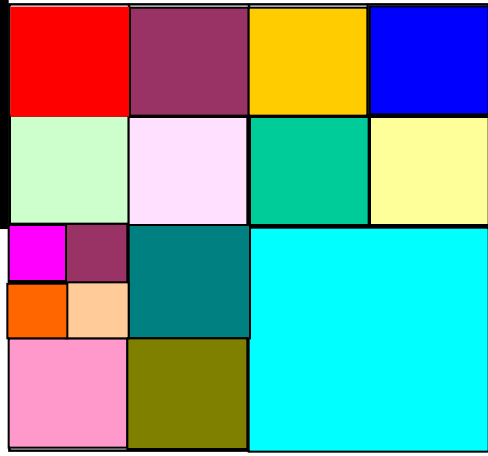



- Quadtree addresses problems by splitting cells recursively into 4 if some storage threshold is exceeded.



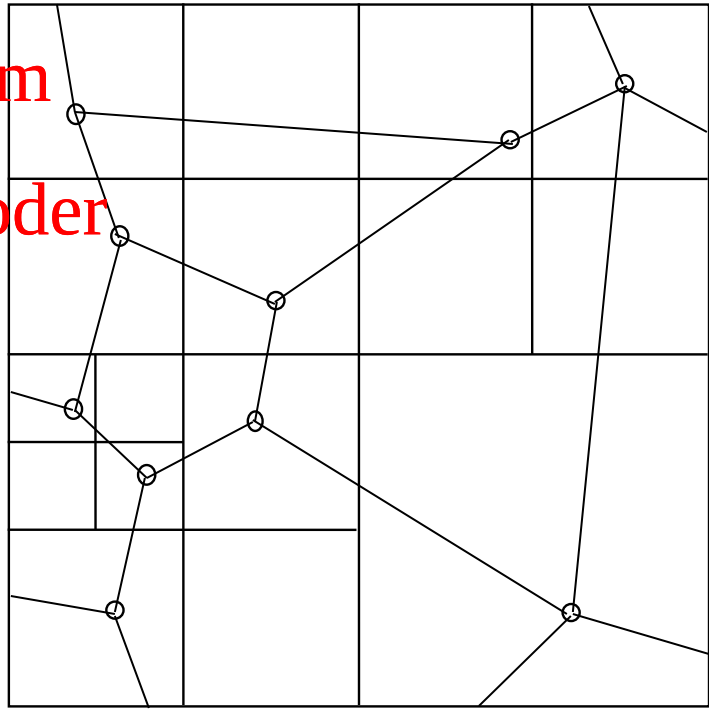

In this example the threshold is 1 point per cell

# Example types of quadtree



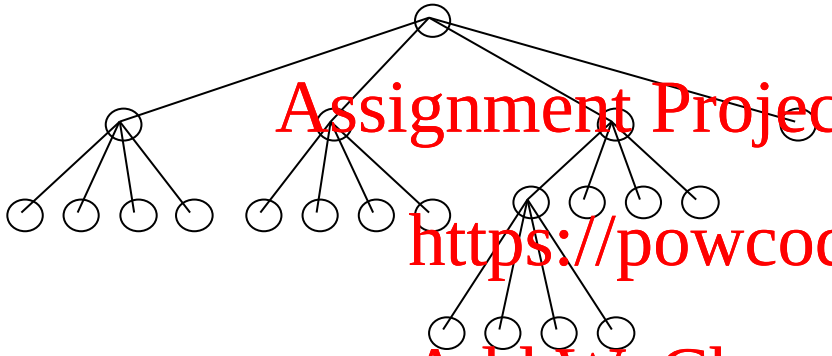
- *Region quadtrees* store raster data
  - cell has uniform pixel value
- *PR quadtrees* store points
  - cell stores point id (one, or more if higher threshold)
- *PM quadtree*
  - cells refer to line edges
  - thresholds: one complete intersecting edge per cell or one vertex per cell (plus its incident edges)
  - geometry stored separate from quadtree cell index
- Lots of other quadtrees

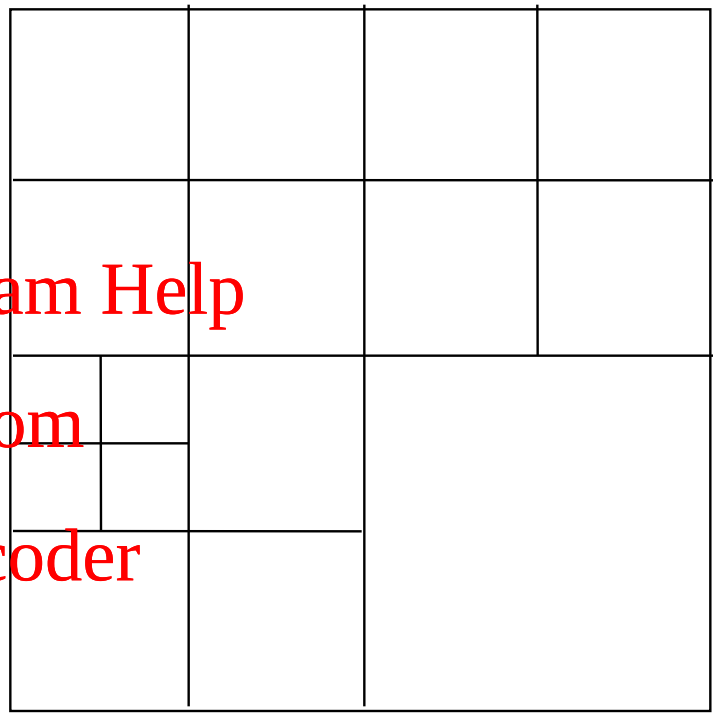




# Pointer-based quadtrees *vs* (Pointerless) linear quadtrees

- earliest quadtrees were all pointer-based tree structures







- not well adapted to database storage (e.g. variable number of levels)

- Alternative is to use location codes to identify cells
- Index cells with the location codes (=keys) - > linear quadtrees (can adapt to standard database indexing methods)

# Linear quadtree

Cells numbered with Morton Numbers (location codes) according to their ancestry in the quadtree.

Cells (identified with location codes) indexed with B-tree or similar

Two possible approaches to create numbers (there are others):

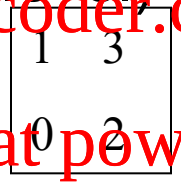


Decide on extent of quadtree as power of 2 (determines length of codes – e.g. one base 4 digit per level)



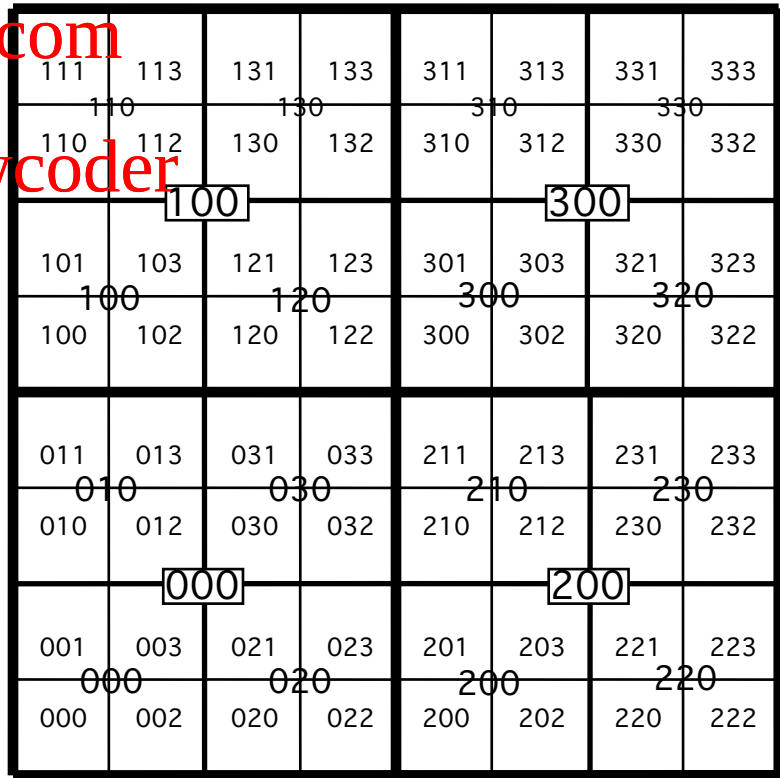
Hierarchically based on quadtree






assume quadrants numbered 0 1 2 3, in order, e.g. lower left, upper left, lower right, upper right. (base 4).

Add respective digit for each sub-quadrant and follow with trailing digits. See diagram in which zeros are used as trailing digits.



# Order of Morton codes / numbers

[Here using decimal numbers - *These are **not** the actual Morton numbers – just their **order***]

| | | | |
|---|---|---|---|
| 5 | 7 | 13 | 15 |
| 4 | 6 | 12 | 14 |
| 1 | 3 | 9 | 11 |
| 0 | 2 | 8 | 10 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 21 | 23 | 29 | 31 | 53 | 55 | 61 | 63 |
| 20 | 22 | 28 | 30 | 52 | 54 | 60 | 62 |
| 17 | 19 | 25 | 27 | 49 | 51 | 57 | 59 |
| 16 | 18 | 24 | 26 | 48 | 50 | 56 | 58 |
| 5 | 7 | 13 | 15 | 37 | 39 | 45 | 47 |
| 4 | 6 | 12 | 14 | 36 | 38 | 44 | 46 |
| 1 | 3 | 9 | 11 | 33 | 35 | 41 | 43 |
| 0 | 2 | 8 | 10 | 32 | 34 | 40 | 42 |

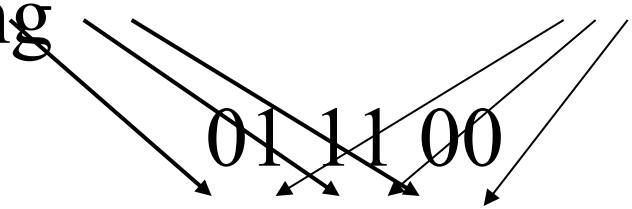
# Bit Interleaving

- Transforms a pair of coordinates into a single number
- Convert x and y coordinates of a point to binary numbers
- Create new number by alternating bits from each coordinate, starting with X coordinate



- Eg. Assuming quadtree of size $2^3$ (width = 8)
  X : 2 = 010 Y : 6 = 110 (3 bits each)

interleaving



= 01 11 00

= 1 3 0 base 4

# Linear Quadtree addressing

Problem with the interleaved (base 4) numbers. *Are they unique?*

Solve either by:

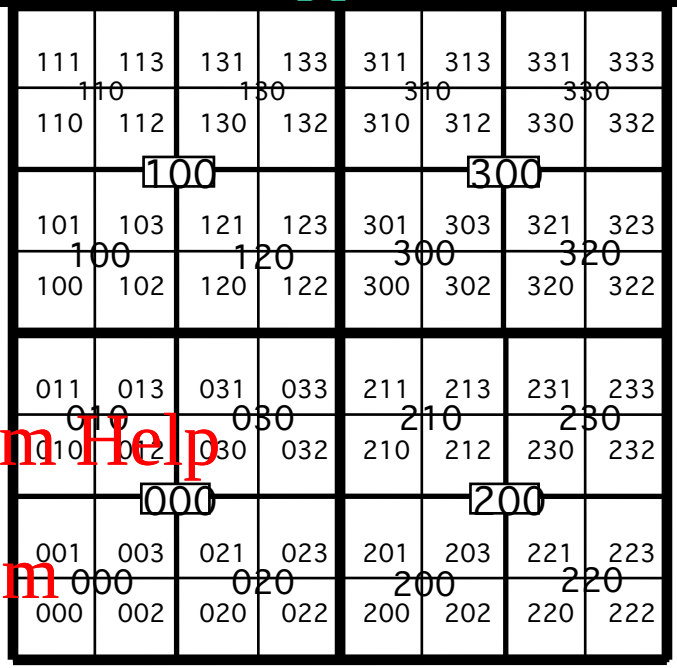
adding cell size to addresses;

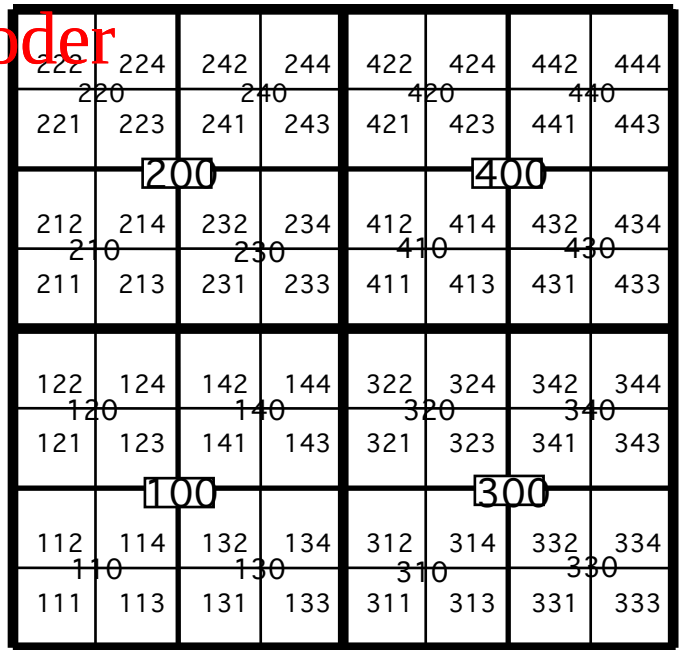
or use base 5 numbers for quadrants, i.e. numbering cells 1,2,3,4 and use 0 for trailing digits.

- add 1 to each of the bit interleaved base 4 digits - *except the trailing digits*:

$$\begin{aligned} & 01\ 11\ 00 \\ = \quad & 1\ \ 3\ \ 0 \\ = \quad & 2\ \ 4\ \ 0 \text{ (if cell is level L = 2 )} \\ = \quad & 2\ \ 4\ \ 1 \text{ (if cell is level L = 3 )} \end{aligned}$$

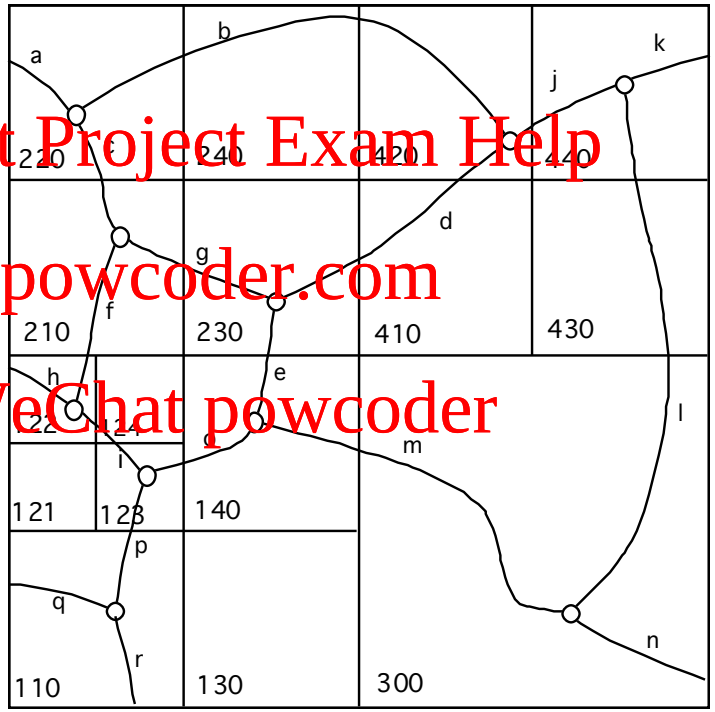
*Level L in quadtree is given by n-k where width of tree is $2^n$ and there are k (= n-L) trailing digits*



# Example quadtree for storing linear features

- Quadtree is of size 8X8
  → n=3
- Here each cell references entire line features that intersect it
- Cells subdivide if more than one vertex or if no vertex and more than one line feature






QUADTREE INDEX FILE

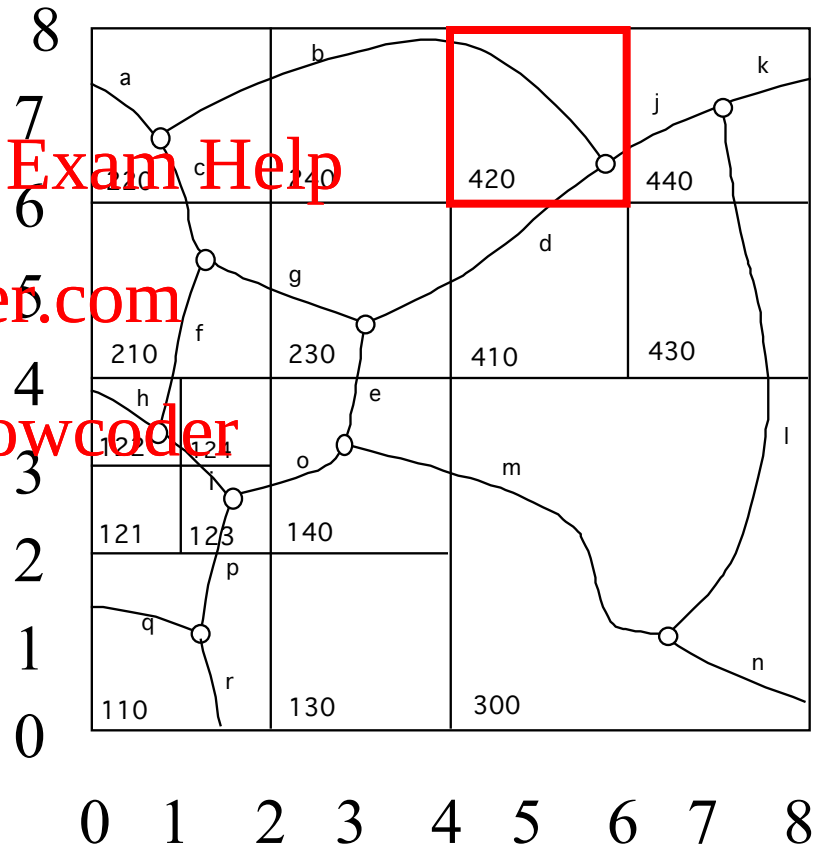
| Quadtree address | Geometry ids |
|---|---|
| 110 | r,p,q |
| 121 | |
| 122 | h,f,i |
| 123 | i,p,o |
| 124 | i |
| 130 | |
| 140 | o,e,m |
| 210 | f,c,g |
| 220 | a,b,c |
| 230 | g,e,d |
| 240 | b |
| 300 | m,n,l |
| 410 | d |
| 420 | b,d,j |
| 430 | l |
| 440 | j,k,l |

# Exercise with location codes

- Use bit interleaving to derive the base 5 code of the highlighted cell

(second level of subdivision so L = 2)

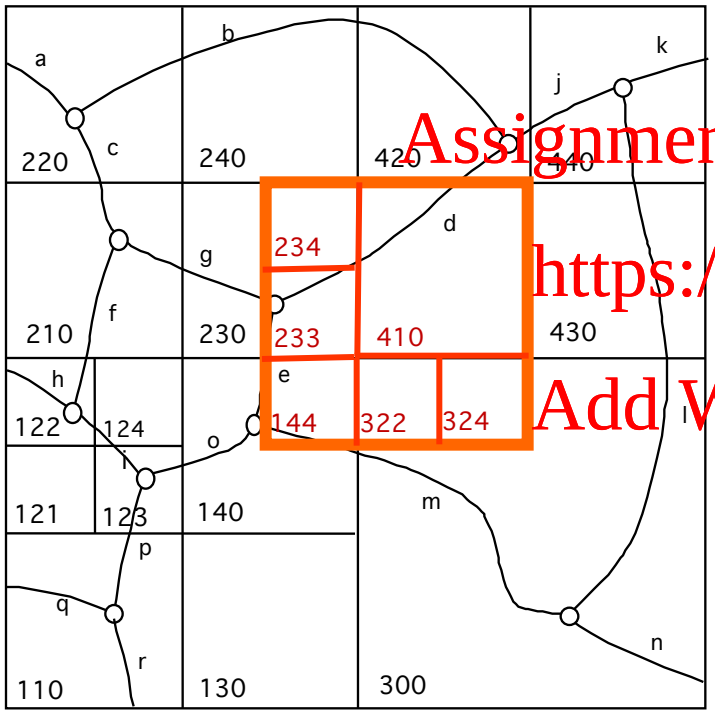





Note : 1st level of subdivision is L = 1

Maximum level of subdivision (smallest cell size) here is L=3

# Querying Linear Quadtrees with query quadtree blocks




- Convert Query window to maximal quadtree cells ("blocks")
- Match the query cells with the data cells
- inside / equal / contains

144 → 140 → e, o, m
233 → 230 → d, e, g
234 → 230 → d, e, g
322 → 300 → l, m, n
324 → 300 → l, m, n
410 → 410 → d



Then retrieve referenced geometry objects for each data cell and test against the query window

N.B. Check for duplicate data cells to avoid retrieving same data

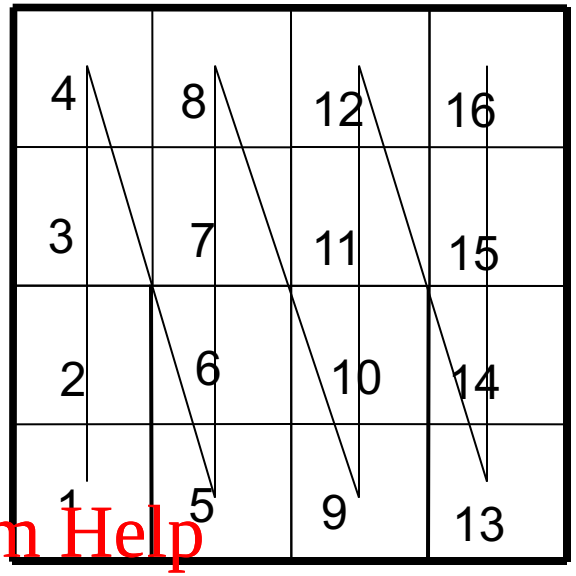
*Note: this is just one possible approach to querying quadtrees*

# Ideal cell numbering systems?

Cells that are adjacent in space should have numbers that are adjacent.

and numbers that are adjacent should refer to locations that are close in space





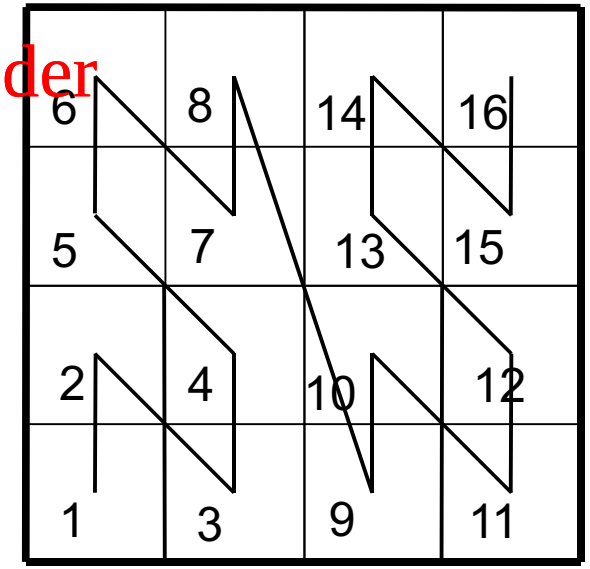

Key order based on x,y concater





data on disk that is sorted by the cell numbers will then tend to be adjacent on disk / storage (hence minimise search between different parts of storage)

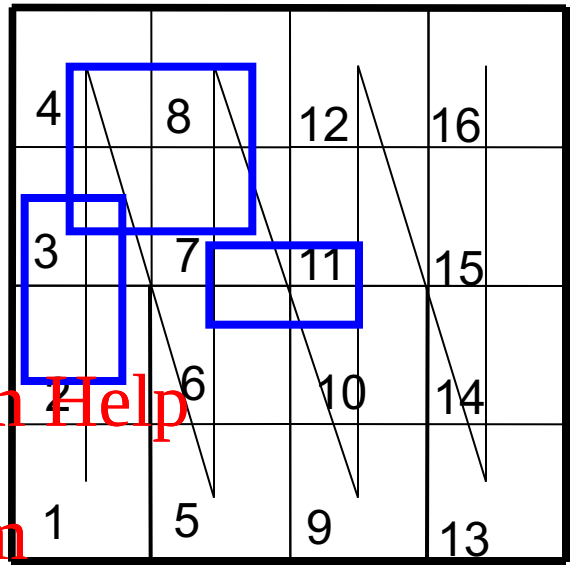
Ideally a query window will only need to access a single region of the storage




Key order based on x,y bit interleaving

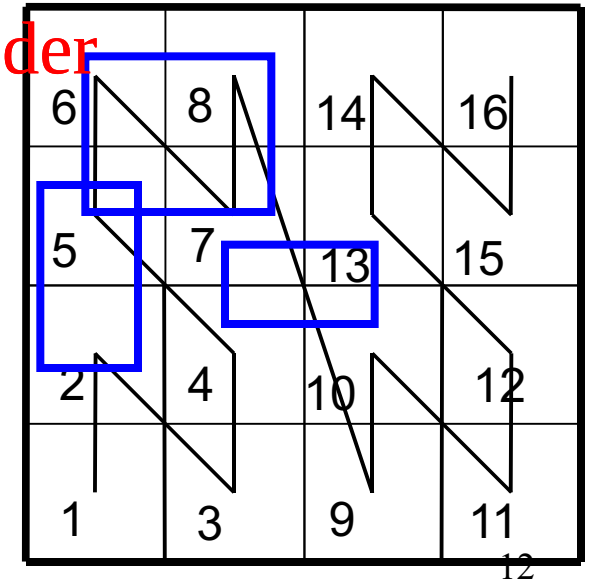
# Space filling curves

- **column-order numbering**
  - in general two cells adjacent in a column will have adjacent numbers, but cells adjacent in rows will not normally have adjacent numbers.






- **Morton Numbers / Z numbers**
  (bit interleaving)
  - increases the chances of spatially adjacent cells having similar numbers, but does not solve the problem entirely.

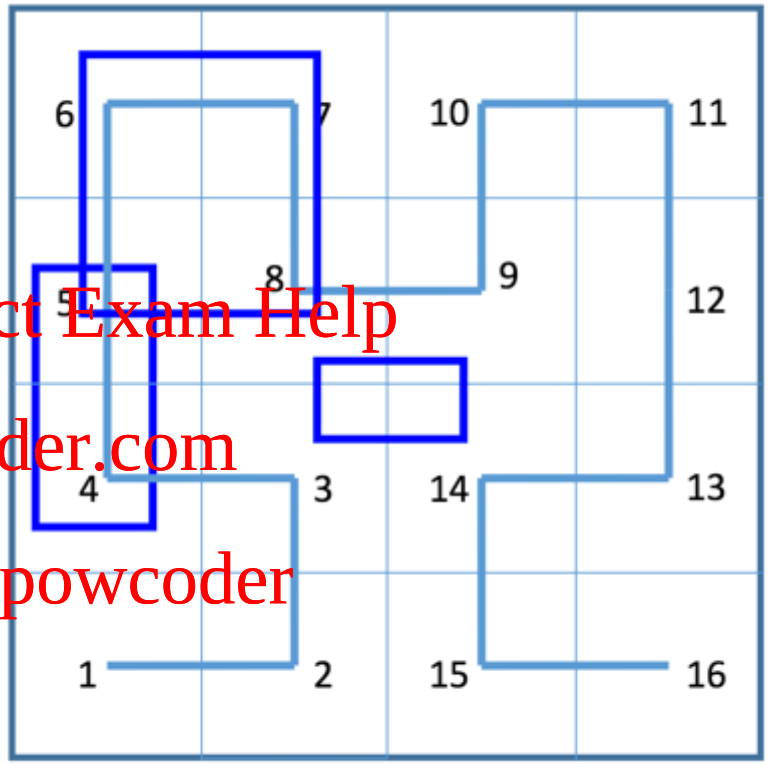


# Hilbert Curve

Hilbert Curve

Improves on Morton numbers / Z-curve, though more complicated to compute





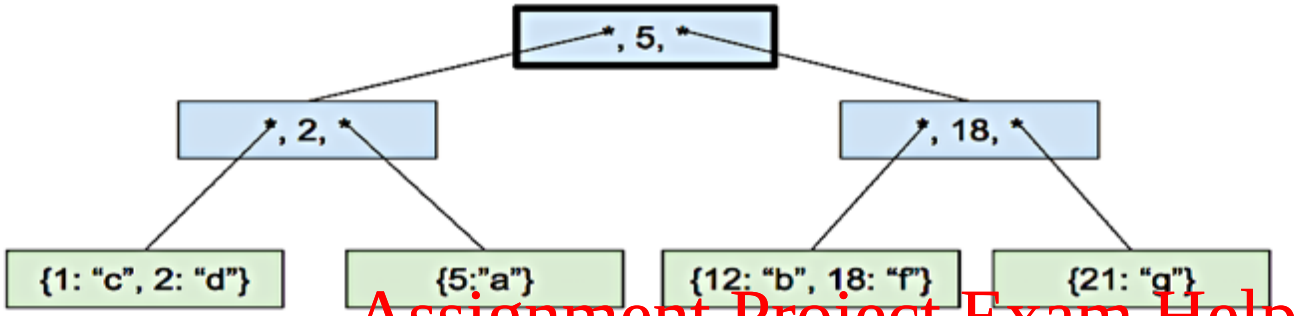

http://mathworld.wolfram.com/HilbertCurve.html

# Large Scale Databases



# Object-directed Spatial Indexing with R-trees

# Revision of B-tree / B$^+$-tree




Example B$^+$-tree [B-tree stores values in non-leaf as well as leaf nodes]



- Balanced, i.e. all leaf nodes at same level
- Each non-leaf node has maximum *m* children (= order) and a minimum of $\lceil m/2 \rceil$ children
- Each non-leaf node has maximum *m-1* keys
- Leaf nodes can store *m* keys and *m* data values
- B$^+$-tree non-leaf nodes store key + child pointers

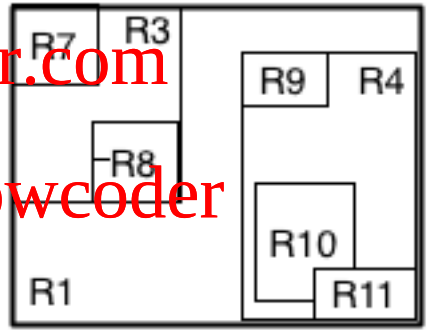
  Leaf nodes store entries of <key + value>
- Root has at least 2 children

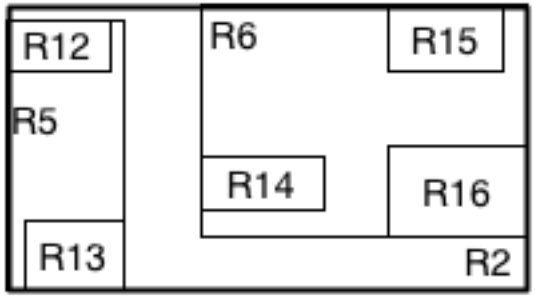
*Diagram:* https://cstack.github.io/db_tutorial/parts/part7.html

# Object-directed indexing with R-Trees

- Multi-dimensional indexing methods which find data within specified ranges of coordinates on the axis of each dimension.
- 2D R-Tree indexes rectangles = bounding boxes of spatial objects (i.e. minimum bounding rectangles **MBR**)
- Extension of B$^+$-tree
- The R-tree creates a hierarchy MBRs of objects.
- The leaf nodes contain the entries with (R7-R16) that enclose the geometries.
- At higher levels of the hierarchy MBRs enclose sets of lower level MBRs.





# R-trees properties

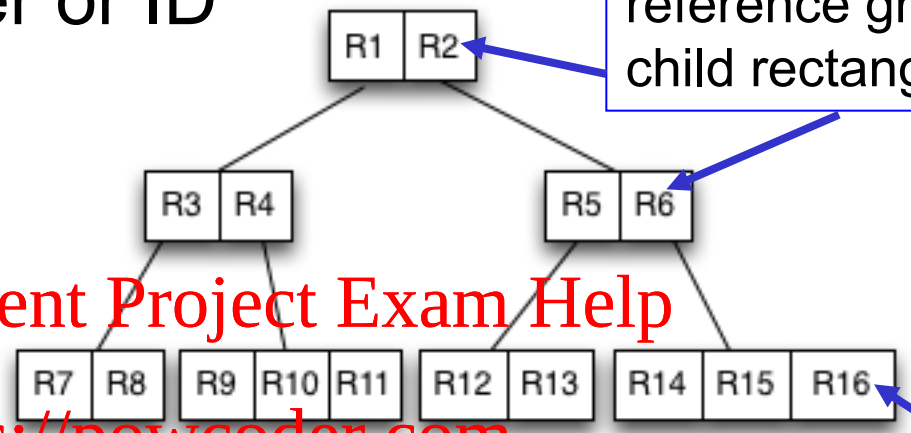
Each entry in a node/page stores a rectangle and a pointer or ID

|children| = |keys|

Each non-leaf node stores a maximum of *M* references to child MBRs (*M*=3 in example)

Each node (apart from root) has minimum of (⌈M/2⌉) entries

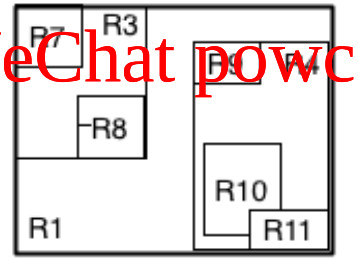
Root has at least 2 entries

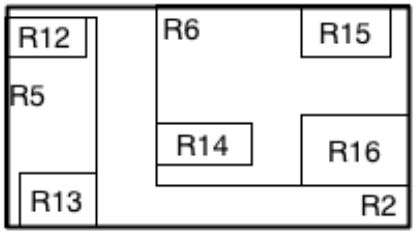



Non-leaf node entries reference groups of child rectangles

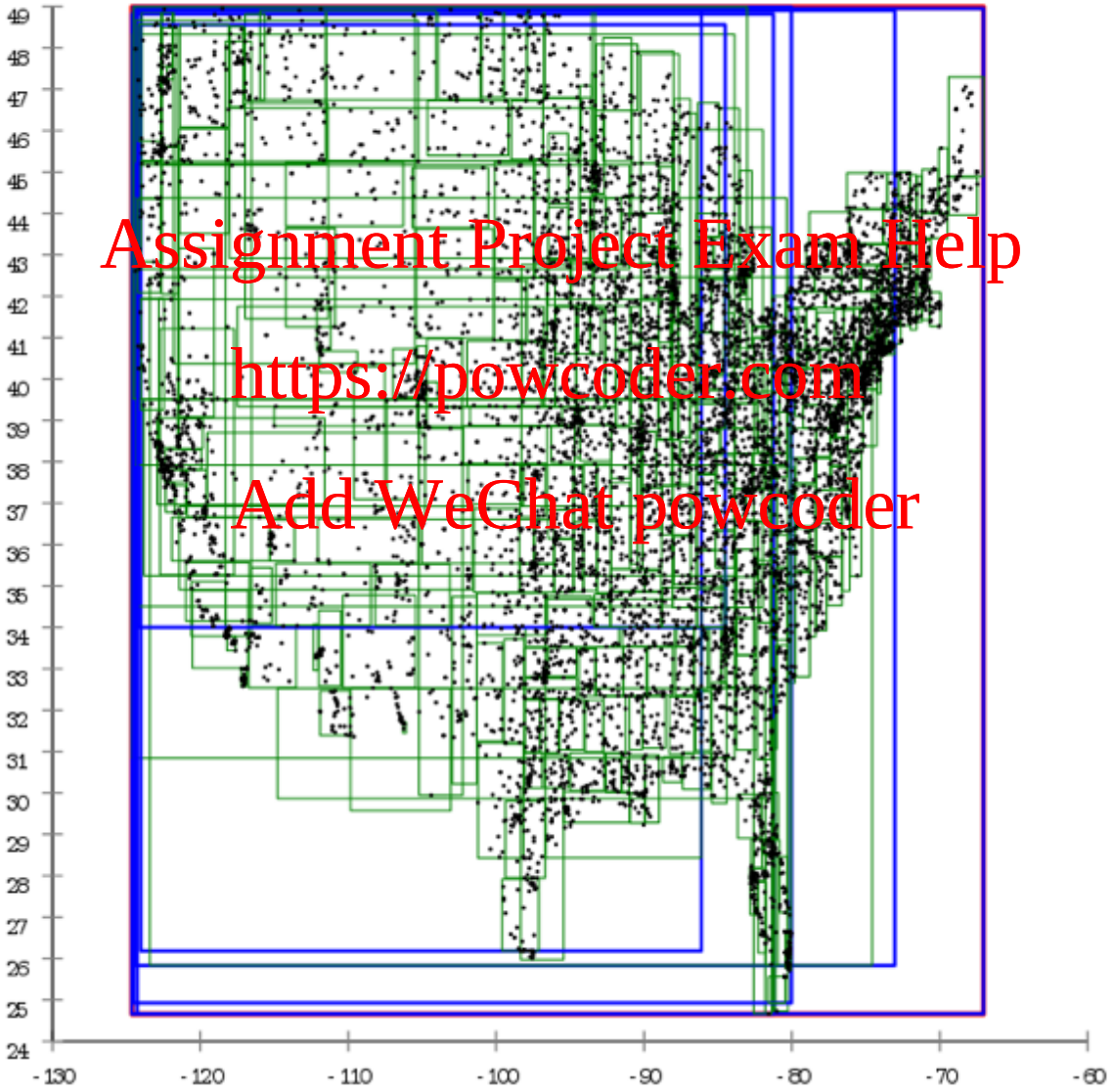
Leaf node entries reference geometry data records





# Example from R*Tree:



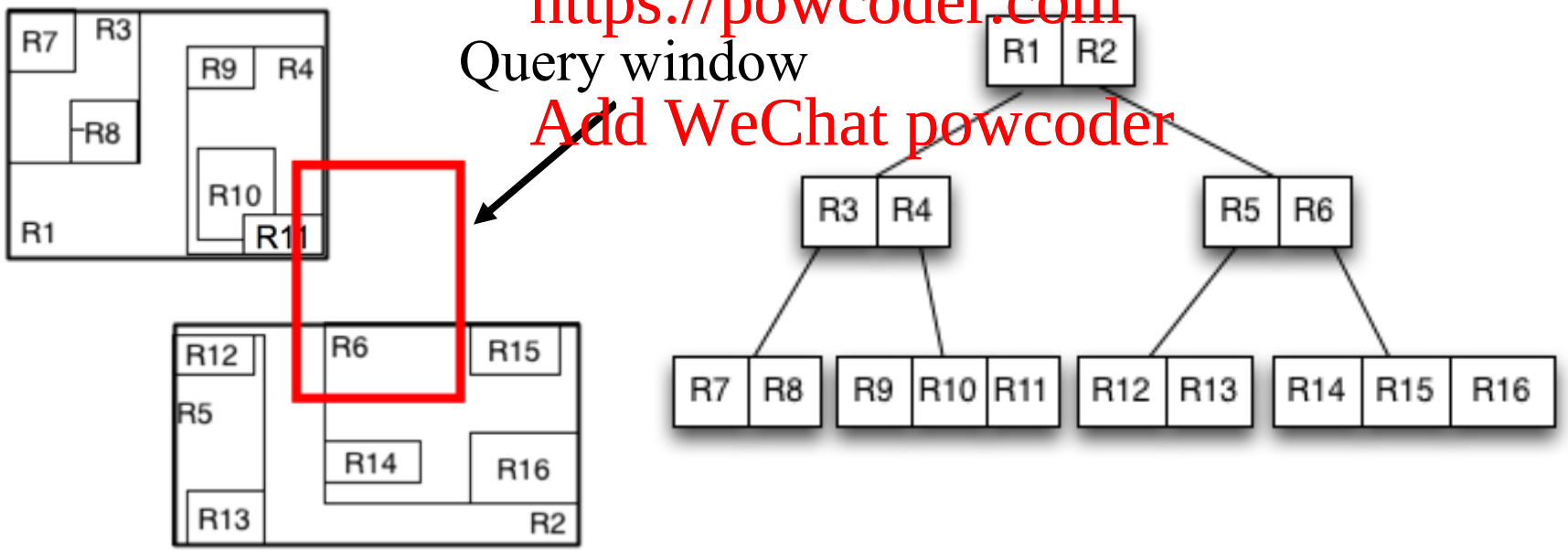



s

# Searching R-trees

- To retrieve data from the R-tree : start at the root and test the search window for overlap with the MBRs of each entry in the root.
- Search descends recursively down every branch for which there is overlap (depth first).

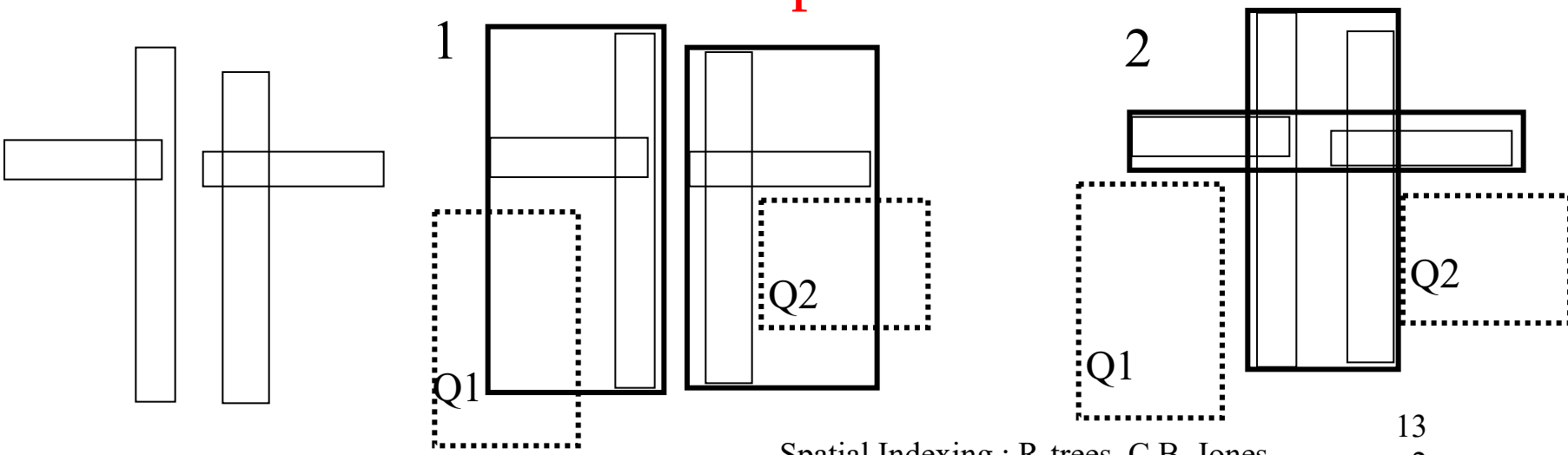




# Efficient R-tree structure

- Objective : minimise the number of fruitless searches down branches of the tree
- Figure shows two alternatives for constructing bounding rectangles around 4 MBRs, with M=3. In option 1, query rectangles Q1 and Q2 each result in fruitless searches. In option 2, these searches would be avoided.

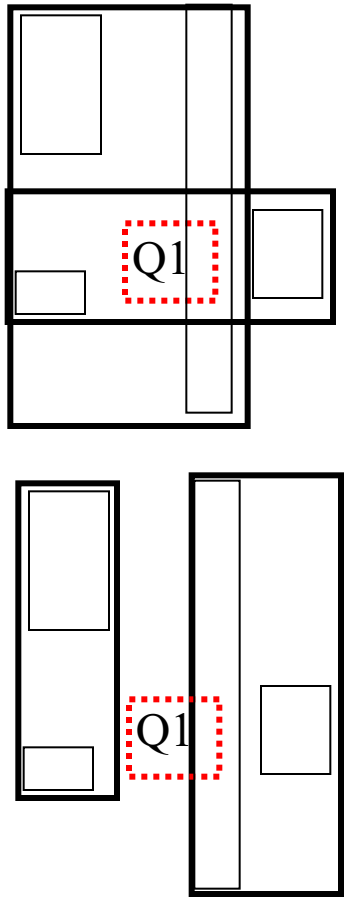




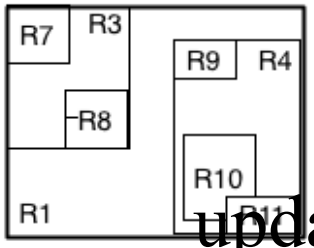
# Methods to improve efficiency

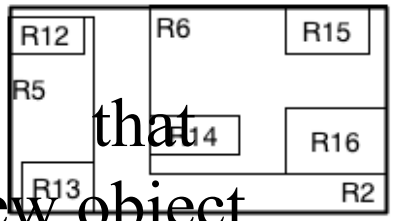
- Minimise area (***coverage***) of MBRs
  - reduces chances of query rectangle covering empty regions of an MBR
- Minimise ***overlap*** of MBRs
  - query windows covering the overlapping areas will always result in searching both branches of the tree - even though they may not contain data objects.
- Minimise ***size of tree***
  - fewer branches to traverse
- BUT the methods may counteract each other





# Building R-Trees



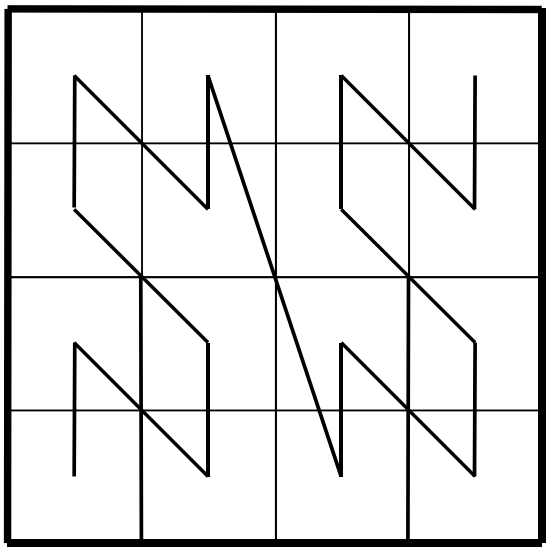



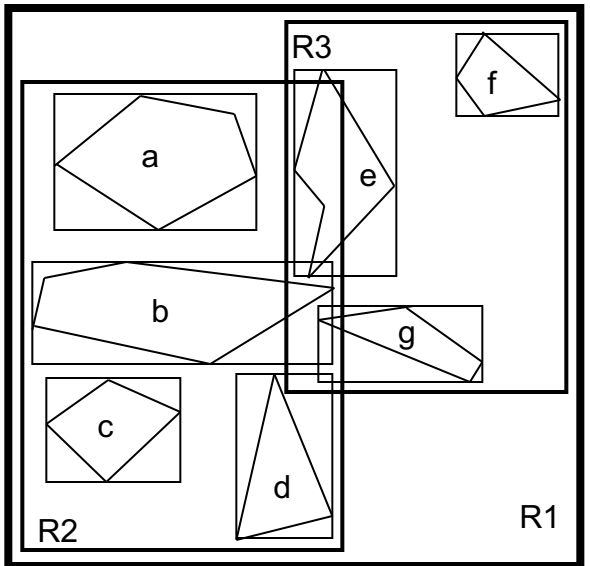

- *Dynamic* procedures perform local updates to the tree when object is inserted:
  - Usually descend tree selecting rectangles that need least or no size increase to fit new object
  - If leaf node is full, apply node splitting strategy
  - Various strategies for node splitting to minimise coverage and overlap
- *Static* methods rebuild the entire tree when a new node is inserted
  - they typically result in a **packed** R-tree
  - Grouping of rectangles can be based on spatial ordering (such as Morton numbers of their centroids)

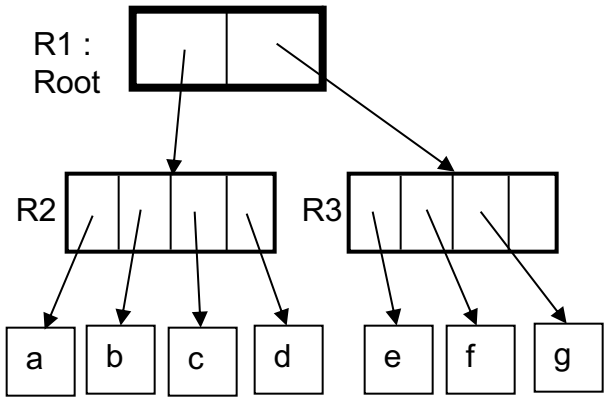
# Morton codes for static R-tree update

- A rectangle is given an ordering based on the Morton code of its centroid (i.e. by bit-interleaving the x and y coordinates of the centre point of the rectangle).
- Use the ordering to create successive groups of M rectangles
- In a packed node method of update the lowest level non-leaf nodes are first filled to capacity.
- Nodes at the remaining higher levels are then constructed by grouping the lower levels in order, again using the Morton numbers of the centroids of the MBRs at the lower level,







# R-Trees references

- Searching". Proceedings ACM SIGMOD 1984.

This is the original paper on R-trees. It presents several methods for node splitting strategies for updating the tree. Given a set a rectangles to be allocated to two new nodes, following node overflow, it attempts to find the two subsets of rectangles that are most distant from each other and which form two distinct clusters. It tends to minimise coverage.



- Beckmann, N., Kriegel H.P, Schneider, R., Seeger, B. (1990). "The R*-tree: an efficient and robust access method for points and rectangles". Proceedings ACM SIGMOD 1990.



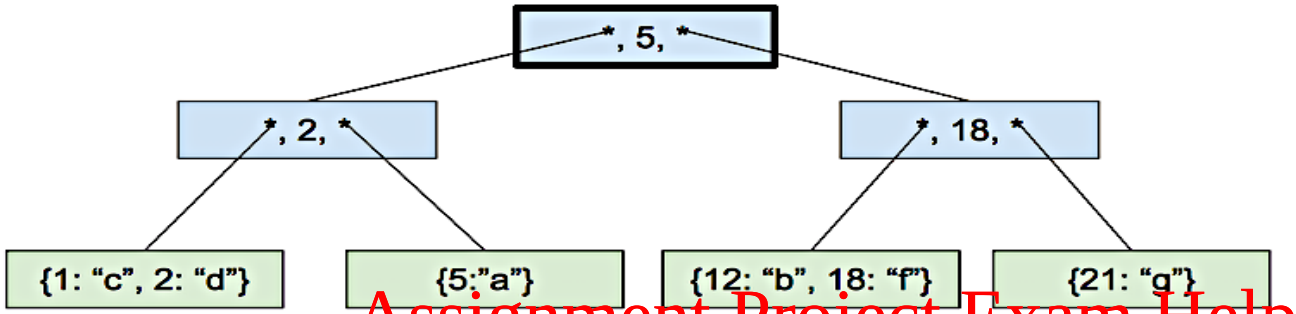
This paper presents a combination of methods to reduce overlap and coverage. When inserting a new rectangle in leaf nodes, choose an entry with free space that will both minimise increase in size and minimise overlap. Also when inserting, re-insert some of the existing data, as the order in which rectangles were inserted previously affects the quality of the R-tree.

# Large Scale Databases



# Object-directed Spatial Indexing with R-trees

# Revision of B-tree / B$^+$-tree




Example B$^+$-tree [B-tree stores values in non-leaf as well as leaf nodes]



- Balanced, i.e. all leaf nodes at same level
- Each non-leaf node has maximum *m* children (= order) and a minimum of ⌈m/2⌉ children
- Each non-leaf node has maximum *m-1* keys
- Leaf nodes can store *m* keys and *m* data values
- B$^+$-tree non-leaf nodes store key + child pointers

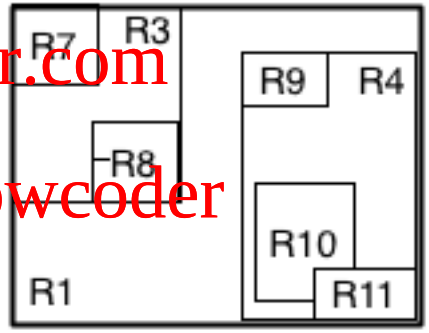
  Leaf nodes store entries of <key + value>
- Root has at least 2 children

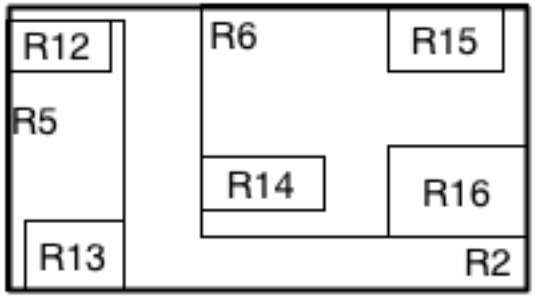
*Diagram:* https://cstack.github.io/db_tutorial/parts/part7.html

# Object-directed indexing with R-Trees

- Multi-dimensional indexing methods which find data within specified ranges of coordinates on the axis of each dimension.
- 2D R-Tree indexes rectangles = bounding boxes of spatial objects (i.e. minimum bounding rectangles **MBR**)
- Extension of $B^+$-tree
- The R-tree creates a hierarchy MBRs of objects.
- The leaf nodes contain the entries with (R7-R16) that enclose the geometries.
- At higher levels of the hierarchy MBRs enclose sets of lower level MBRs.





# R-trees properties

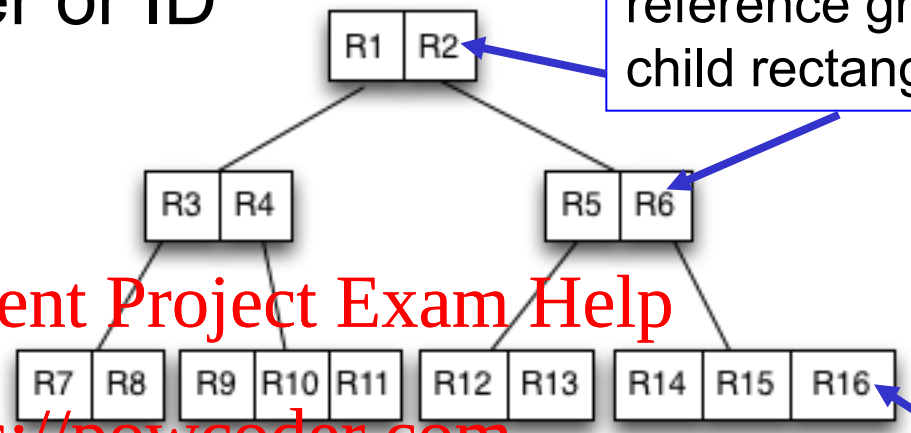
Each entry in a node/page stores a rectangle and a pointer or ID

$|\text{children}| = |\text{keys}|$

Each non-leaf node stores a maximum of $M$ references to child MBRs ($M$=3 in example)

Each node (apart from root) has minimum of ($\lceil M/2 \rceil$) entries

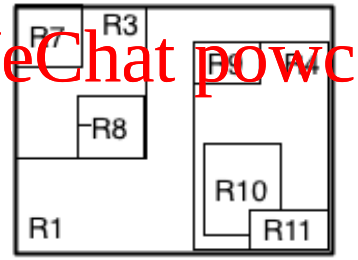
Root has at least 2 entries

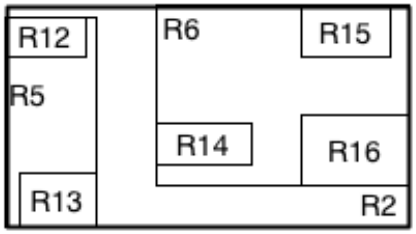



Non-leaf node entries reference groups of child rectangles

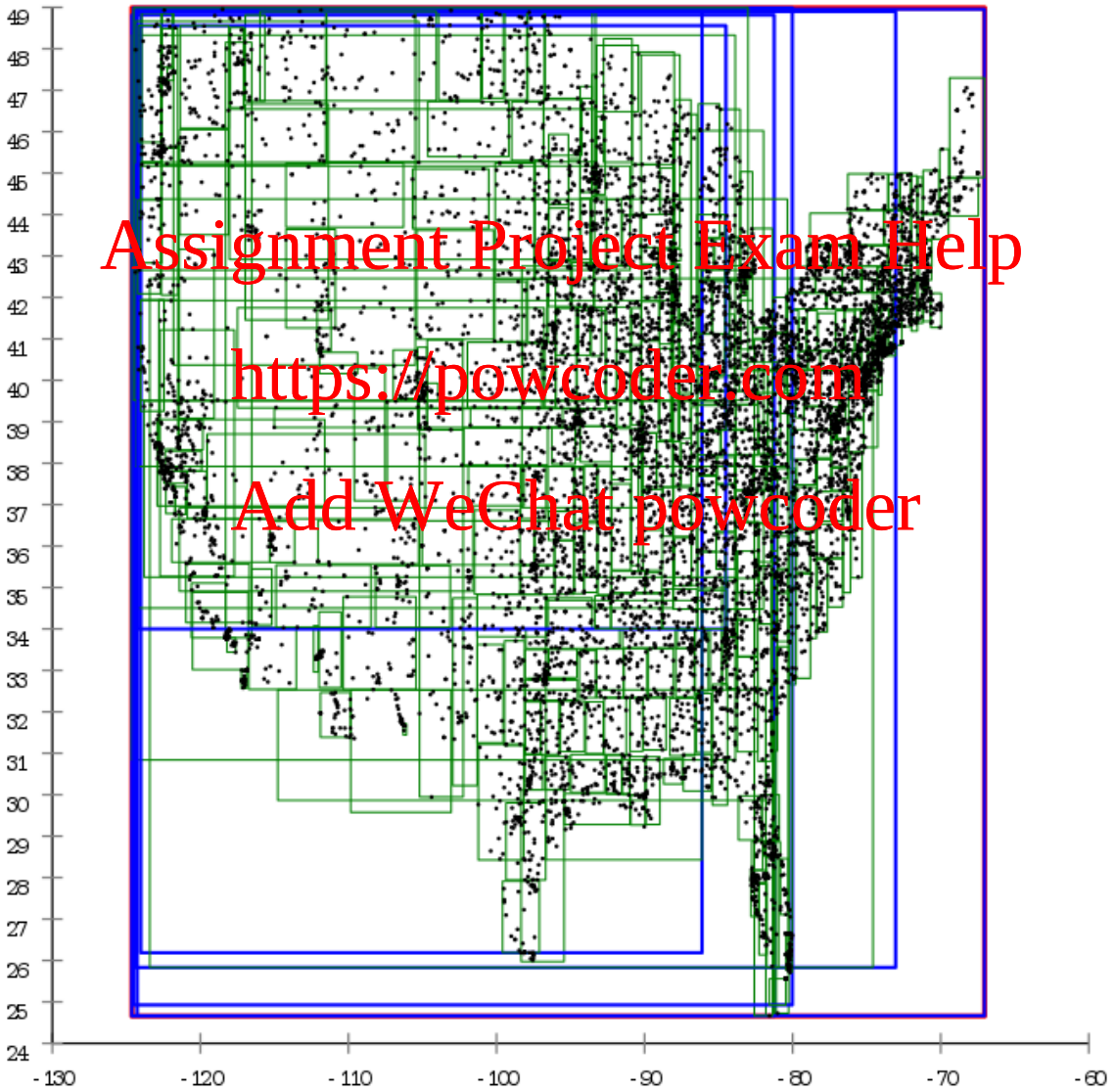
Leaf node entries reference geometry data records





# Example from R*Tree:



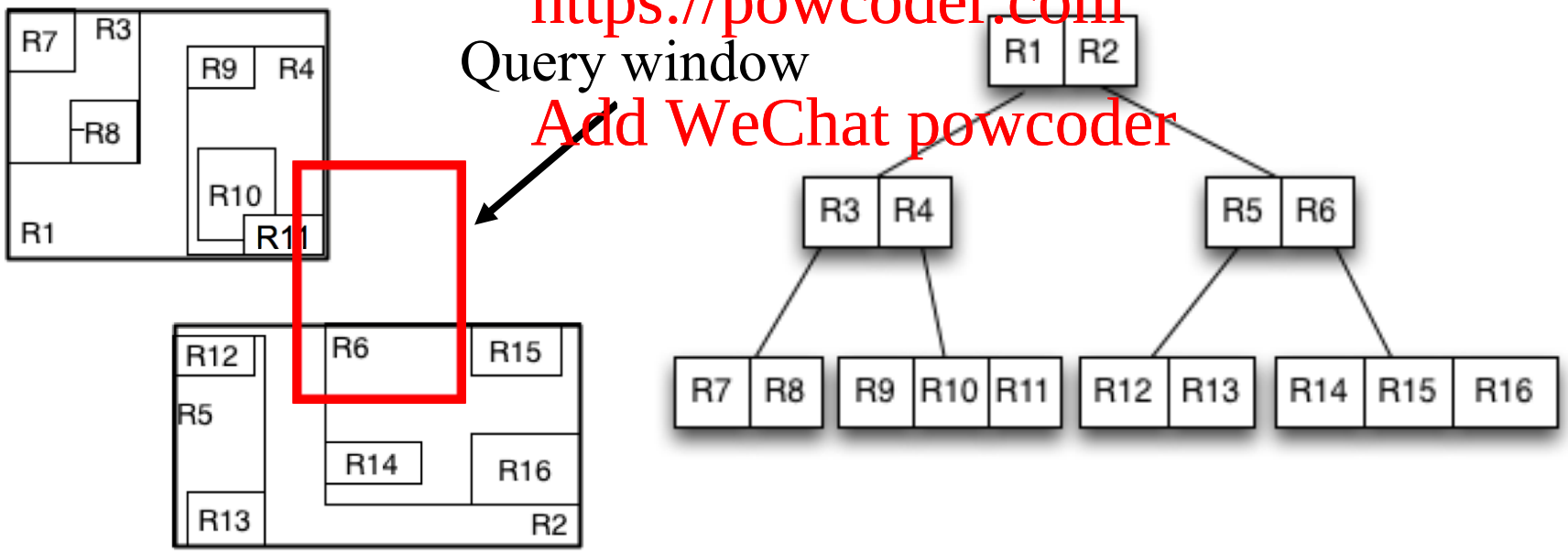



s

# Searching R-trees

- To retrieve data from the R-tree : start at the root and test the search window for overlap with the MBRs of each entry in the root.
- Search descends recursively down every branch for which there is overlap (depth first).

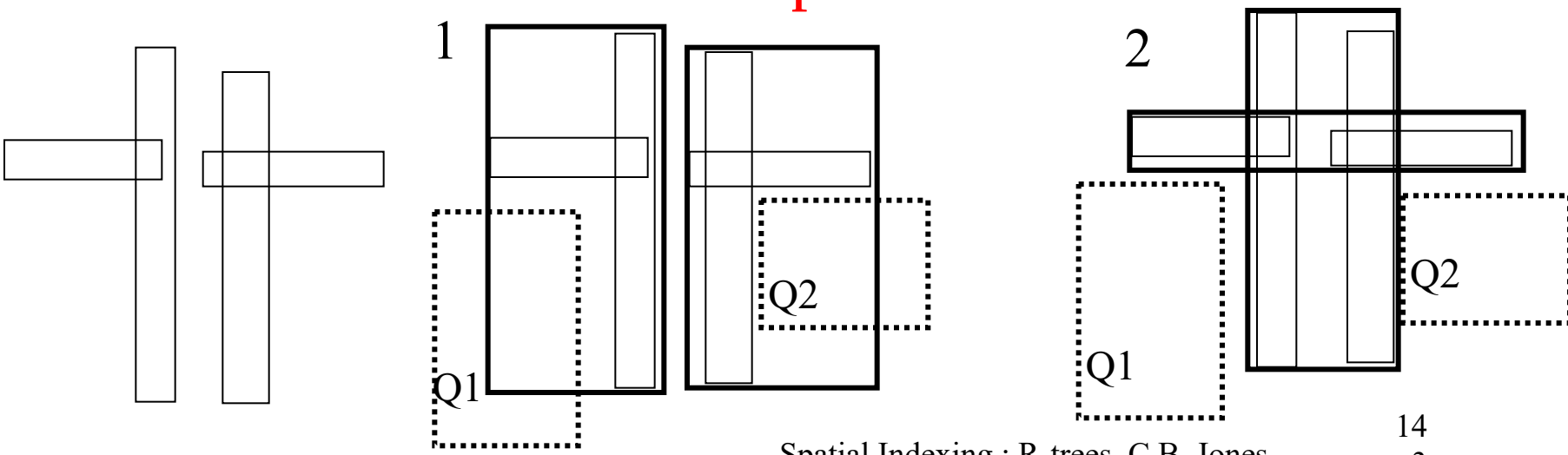




# Efficient R-tree structure

- Objective : minimise the number of fruitless searches down branches of the tree
- Figure shows two alternatives for constructing bounding rectangles around 4 MBRs, with M=3. In option 1, query rectangles Q1 and Q2 each result in fruitless searches. In option 2, these searches would be avoided.

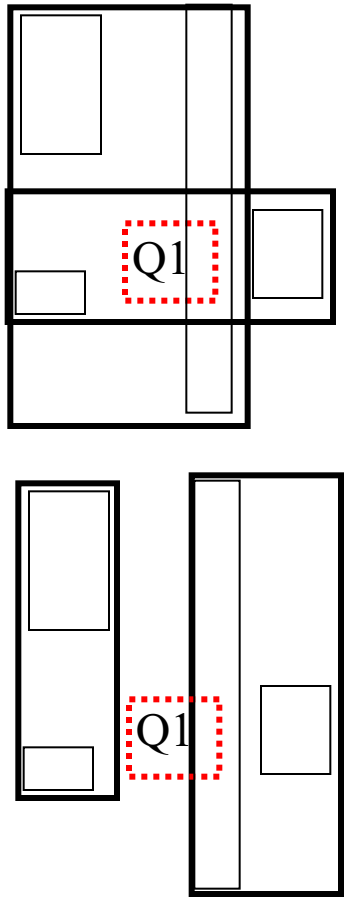




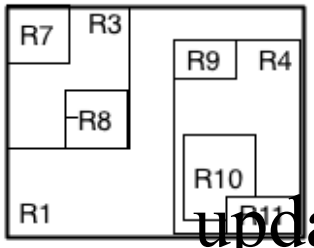
# Methods to improve efficiency

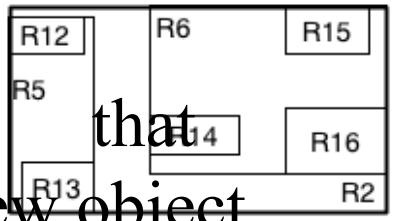
- Minimise area (***coverage***) of MBRs
  - reduces chances of query rectangle covering empty regions of an MBR
- Minimise ***overlap*** of MBRs
  - query windows covering the overlapping areas will always result in searching both branches of the tree - even though they may not contain data objects.
- Minimise ***size of tree***
  - fewer branches to traverse
- BUT the methods may counteract each other





# Building R-Trees



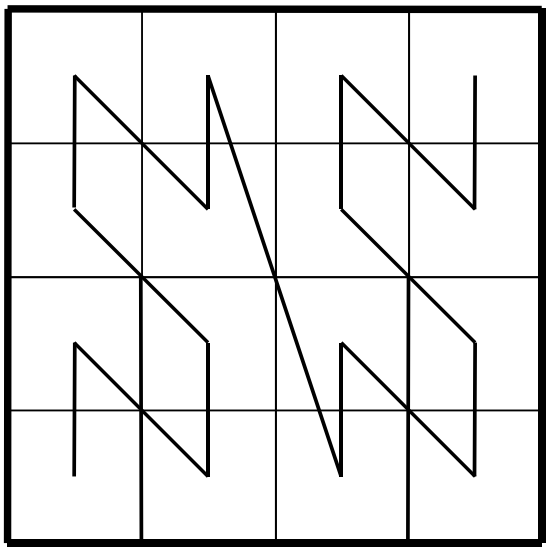



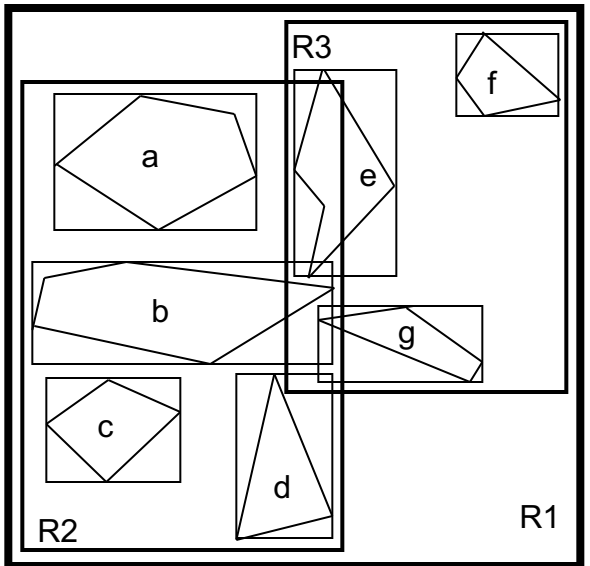

- *Dynamic* procedures perform local updates to the tree when object is inserted:
  - Usually descend tree selecting rectangles that need least or no size increase to fit new object
  - If leaf node is full, apply node splitting strategy
  - Various strategies for node splitting to minimise coverage and overlap
- *Static* methods rebuild the entire tree when a new node is inserted
  - they typically result in a **packed** R-tree
  - Grouping of rectangles can be based on spatial ordering (such as Morton numbers of their centroids)

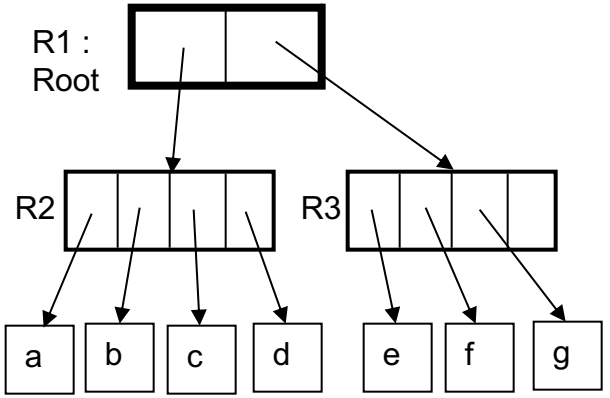
# Morton codes for static R-tree update

- A rectangle is given an ordering based on the Morton code of its centroid (i.e. by bit-interleaving the x and y coordinates of the centre point of the rectangle).
- Use the ordering to create successive groups of M rectangles
- In a packed node method of update the lowest level non-leaf nodes are first filled to capacity.
- Nodes at the remaining higher levels are then constructed by grouping the lower levels in order, again using the Morton numbers of the centroids of the MBRs at the lower level,







# R-Trees references

- Searching". Proceedings ACM SIGMOD 1984.

This is the original paper on R-trees. It presents several methods for node splitting strategies for updating the tree. Given a set a rectangles to be allocated to two new nodes, following node overflow, it attempts to find the two subsets of rectangles that are most distant from each other and which form two distinct clusters. It tends to minimise coverage.



- Beckmann, N., Kriegel H.P, Schneider, R., Seeger, B. (1990). "The R*-tree: an efficient and robust access method for points and rectangles". Proceedings ACM SIGMOD 1990.



This paper presents a combination of methods to reduce overlap and coverage. When inserting a new rectangle in leaf nodes, choose an entry with free space that will both minimise increase in size and minimise overlap. Also when inserting, re-insert some of the existing data, as the order in which rectangles were inserted previously affects the quality of the R-tree.

# Spatio-textual (spatial keyword) indexing

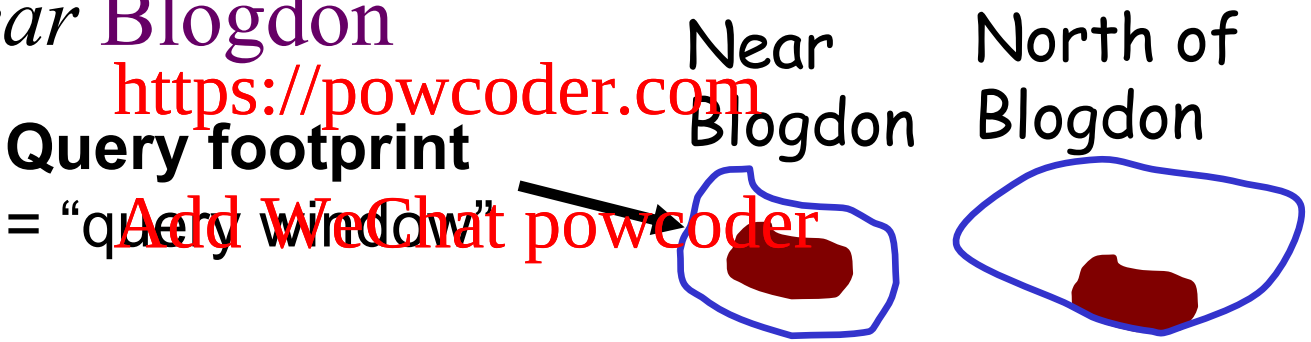
# Motivation: Find geographically-specific resources on the web

Form of query:

Something *sp_related_to* Somewhere

*e.g.* castles *near* Blogdon






Something : Concept terms

Somewhere : Placename

*sp_related_to* = in, near, within Xkm, north_of…etc.

# Matching queries to documents

Document composed of
Concept (text) terms
+ Spatial "document footprint"



•Document footprint determined by geo-locations of places mentioned in document

•Query requires matching :
Query concept terms and Query footprint
to
Document terms and Document footprint

# Geo-Parsing : true & false references

Geo-parsing: Detecting geo-references (mostly toponyms) and resolving them (geo-coding) to coordinates

Actual place names, e.g.

New York, Raleigh, Lower Manhattan, Kansas City, Mo, Washington

- All words highlighted in yellow are names of places *somewhere*

JACK HAGEL, Staff Writer
Redevelopment of the World Trade Center site in New York is getting some input from a Raleigh real-estate maven.

York Properties President Smedes York was chairman of an Urban Land Institute panel at the World Trade Center and Lower Manhattan Summit last month.

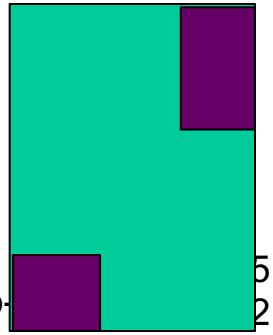
The group heard presentations on how the area surrounding the site of the Sept. 11, 2001, terrorist attacks should be redeveloped. It suggested retail be a central focus for developers. The institute will issue a report based on the recommendations before the end of the year.

York was chairman of the Urban Land Institute, a Washington nonprofit organization, from 1989 to 1991. His dad, J.W. "Willie" York, joined the Urban Land Institute in 1947. That's where he met J.C. Nichols, the developer of Country Club Plaza in Kansas City, Mo. -- the center that inspired Willie York to build Raleigh's Cameron Village, the Southeast's first shopping center.

# Document Footprints

- Document footprints often approximated by one or more minimum bounding rectangles (MBR) or by points
- One MBR could cover all geo-references in a document - but poor results if geo-references in a document are far apart
- Could represent each document as a set of MBRs for the main geo-references





# Text Indexing: Inverted File Index

- The set of terms is referred to as a Dictionary
- The Dictionary (lexicon) is sorted alphabetically
- The list of DocumentIDs is called a Postings List (sorted by ID)



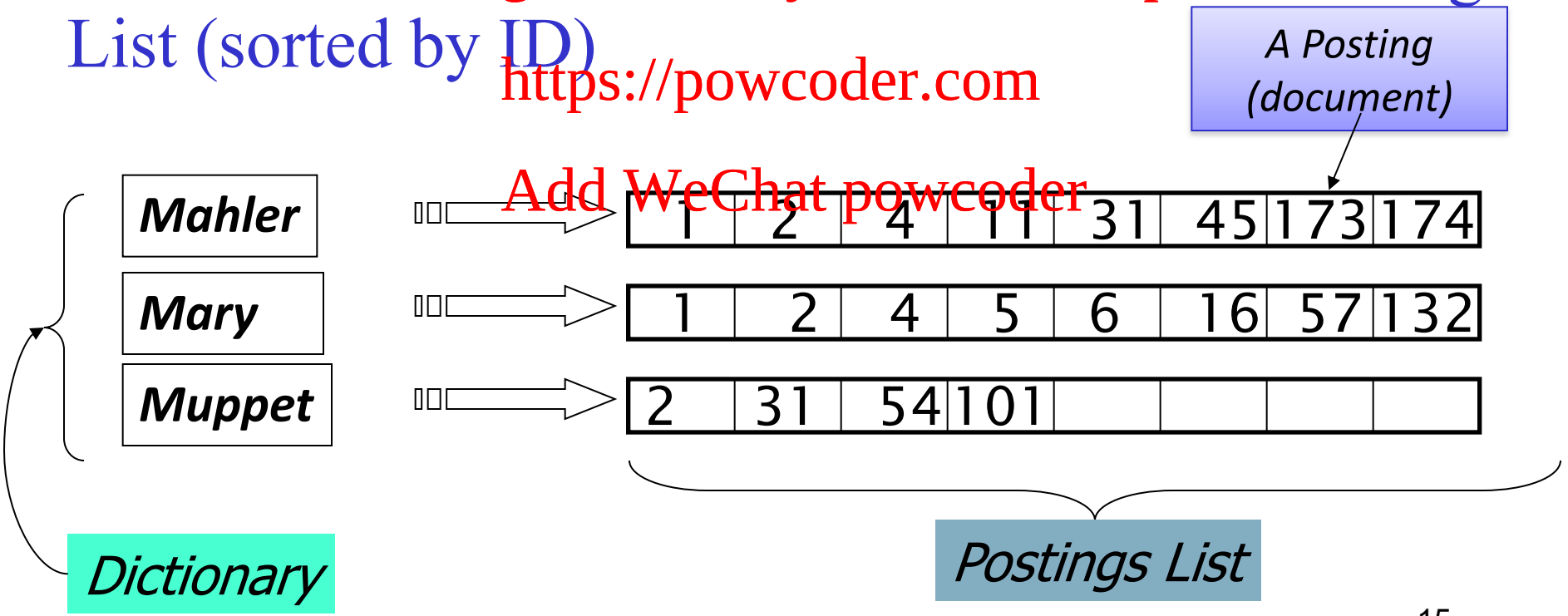



Figure based on Manning et al 2008

# Inverted File Index Simple Example

## Documents Ti = DocIDs

T1: The cow jumped over the moon

T2: How now blue cow

T3: Now the moon shines on the blue cow



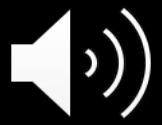
To find the documents that contain all the terms "blue" "moon" "cow" intersect the respective postings lists of

$$\{2, 3\} \cap \{1, 3\} \cap \{1, 2, 3\} = \{3\}$$

Thus document 3 is returned

| Terms (Dictionary) | DocIDS Postings List |
|---|---|
| blue | {2, 3} |
| cow | {1, 2, 3} |
| how | {2} |
| jumped | {1} |
| moon | {1, 3} |
| now | {2, 3} |
| on | {3} |
| over | {1} |
| shines | {3} |
| the | {1, 3} |

# Combining Text Indexing with Spatial Indexing

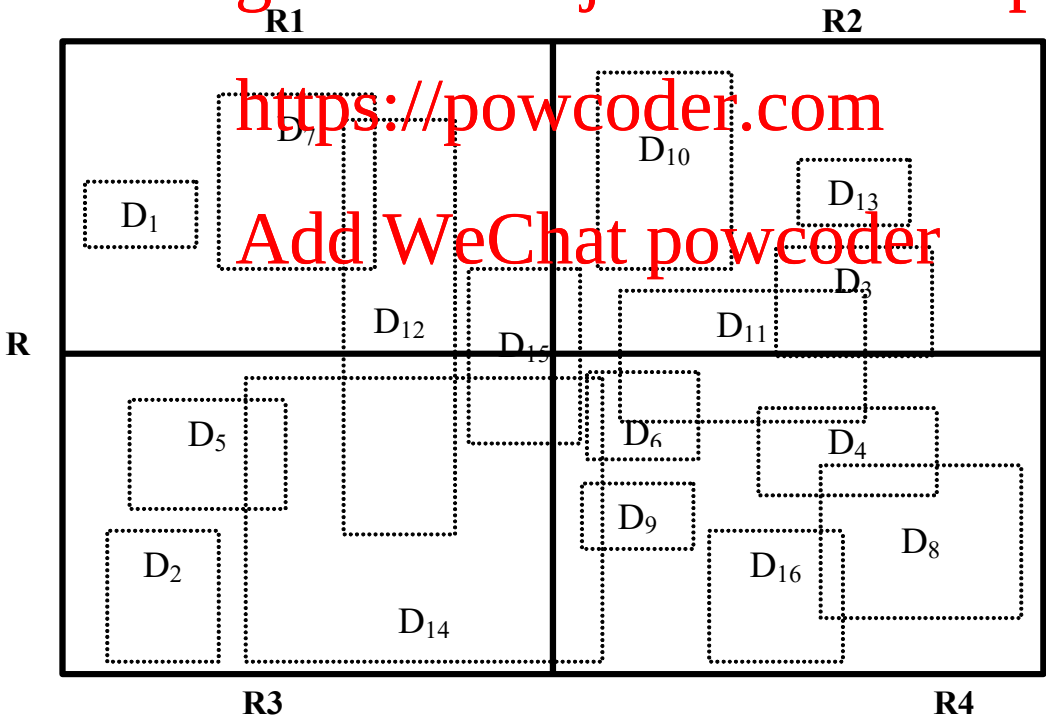


1. Separate text and spatial indexes
2. Space first / Spatial primary index
3. Text first / Text primary index

# Spatial Indexing of Web Documents with a Regular Grid

- Derive document footprint(s) $D_i$ for each document
- For each cell, record documents whose footprints intersect it

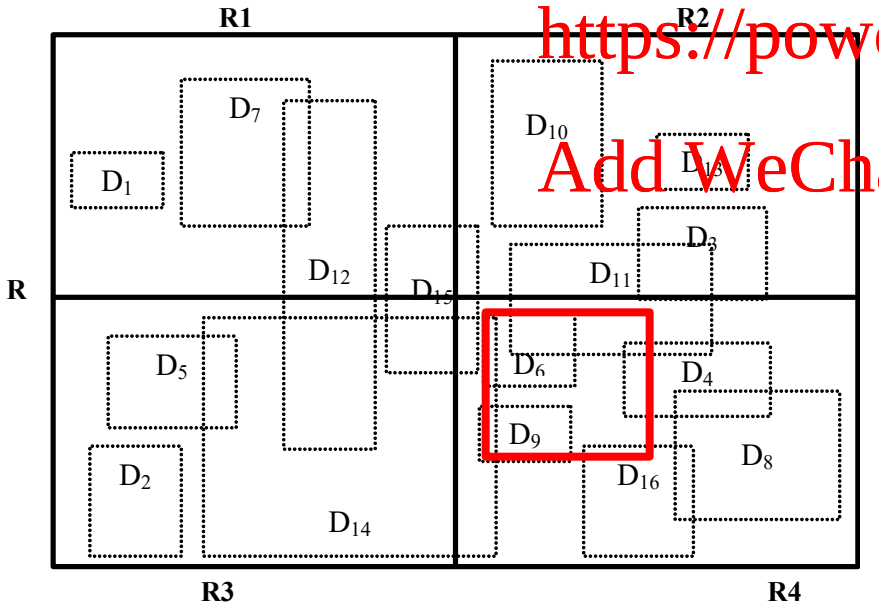




# Separate text and spatial indexes with query <terms + query footprint>

| | |
|---|---|
| Term1 | D1, D2, D6, D23, … |
| Term2 | D2, D6, D9, D11, … |
| Term3 | D27, D85, … |



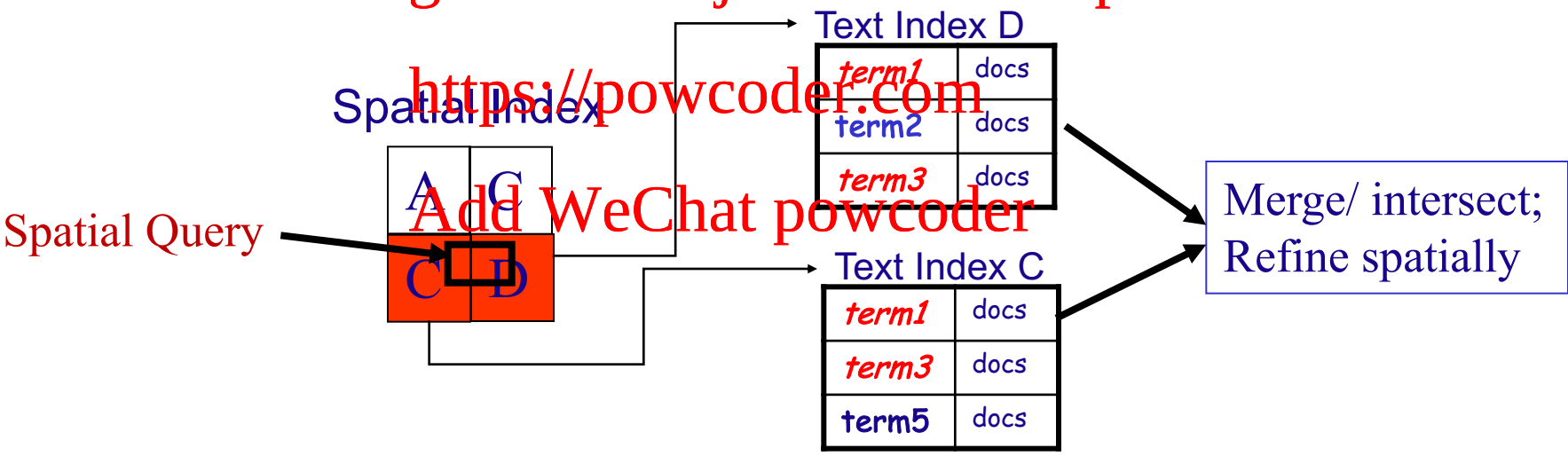



1. Find documents containing query term(s)
2. Find documents intersecting query footprint
3. Find intersection of the two sets (here D6, D9, D11)

# Integrated approaches: Space-first methods

- Spatial cells of index point to inverted file (text) indexes of documents whose footprints intersect the cell
- Below illustrated with a regular grid
- [See next slide for an R-tree (with more detail)]






- Find cells that intersect query footprint / window;
- For each such cell, search associated inverted index to find documents that contain all query terms;
- Test (filter) resulting document footprints against query footprint.

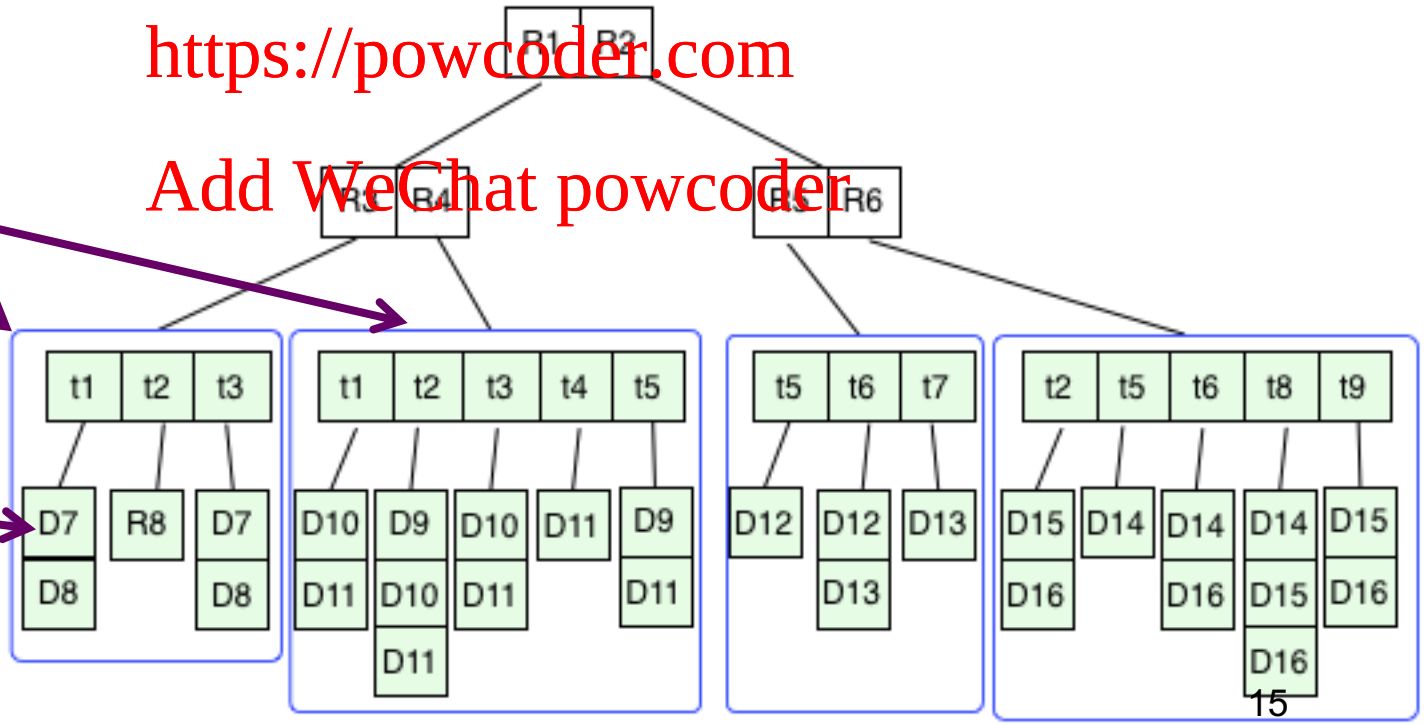
# Space First with R-tree

Each entry in a leaf node points to an inverted file index of documents whose footprints are inside the corresponding rectangle



Inverted (text) indexes

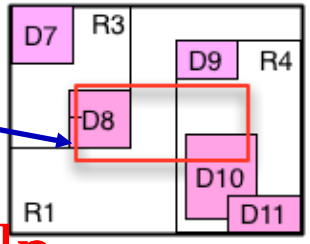
Documents with footprints inside parent rectangle



# Space First with R-tree (*cont.*)

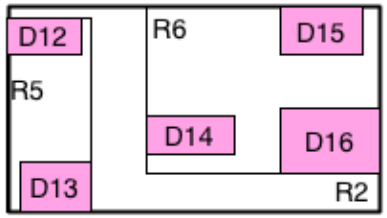
Query text: t1, t3

Query footprint



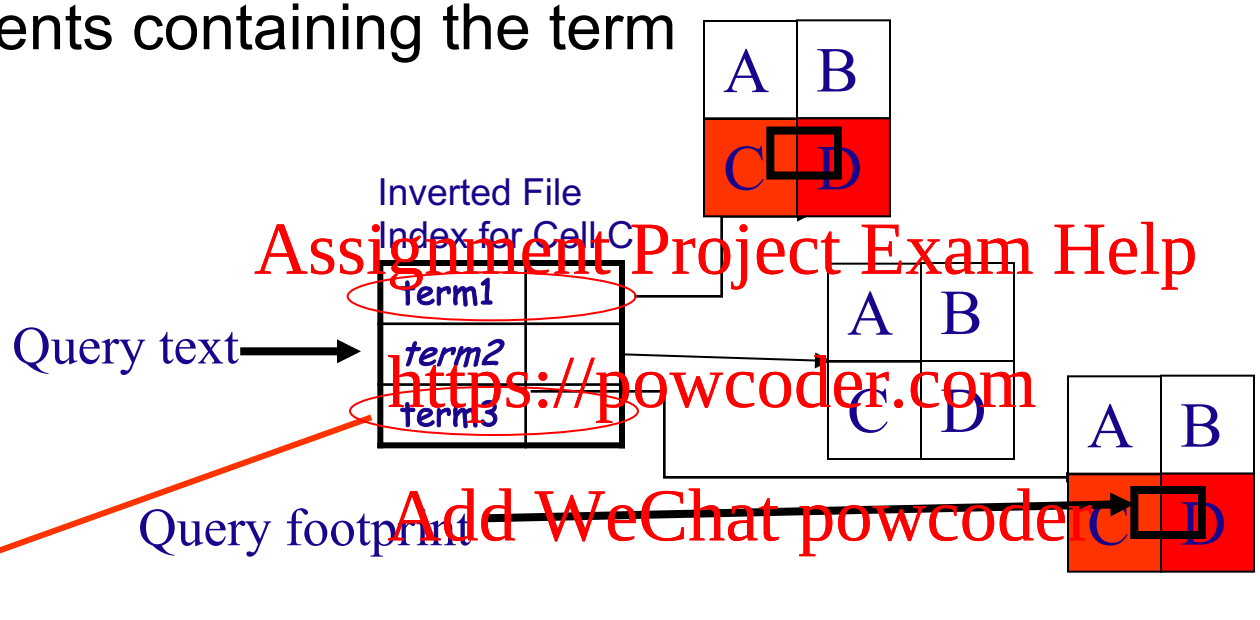

- Find leaf cells that intersect query footprint: R3, R4
- Find documents that contain all query terms: D8, D9, D10, D11
- Test document footprints against query footprint
- -> D8, D10





# Text first spatio-textual indexing

For each term in inverted file index: store spatial index of documents containing the term






Index Entry (for regular grid/quadtree): term3 : cellC($D_1$, $D_7$); cellD($D_3$, $D_{11}$, $D_{13}$)…

Retrieval:

- For each query term, access associated spatial index to retrieve ids of documents in spatial cells intersecting the query footprint
- Intersect results to find documents that contain all query terms
- Filter those documents against the query footprint

# References

Fernando Melo and Bruno Martins. Automated geocoding of textual documents: A survey of current approaches. Transactions in GIS , 21(1):3–38, 2017.

Dingming Wu, Gao Cong, and Christian S. Jensen. A framework for efficient spatial web object retrieval. The VLDB Journal, 21(6):797–822, December 2012.

Yinghua Zhou, Xing Xie, Chuang Wang, Yuchang Gong, and Wei-Ying Ma. Hybrid index structures for location-based web search. In CIKM, pages 155-162, 2005.

S. Vaid, C.B. Jones, H. Joho, and M. Sanderson. Spatio-textual indexing for geographical search on the web. In SSTD, pages 218-235, 2005.

Chengyuan Zhang, Ying Zhang, Wenjie Zhang, and Xuemin Lin. Inverted linear quadtree: Efficient top k spatial keyword search. In ICDE, pages 901-912. IEEE Computer Society, 2013.

# Large Scale Databases



# Spatial Databases Revision

# Class Test structure

4 Questions (with sub-parts)

- **ANSWER ALL QUESTIONS**



10 marks per question



2 questions on Alia Abdelmoty's material

2 questions on Chris Jones's material

# Spatial Databases Questions in the Class Test

Focus for ***Spatial Databases*** questions is on problem solving





This requires understanding of concepts but look especially at examples of questions in exercises that require solving a problem of some sort

# databases

Subject of questions can relate to:

- Construction of spatial indexes given a set of objects
  Requires understand principles of different spatial indexing methods
- Assignment Project (spatial keys) :
  how they are derived from coordinates;
  how they represent points and how they represent cells



- 9-intersection model for representing topological relations with area objects and linear objects
- SQL queries given a set of data tables
  generic spatial SQL as illustrated in lecture slides is fine,
  i.e. not necessary to memorise the exact names

# Spatial Database Topics Covered

- Geographical information characteristics
  Spatial data models, Locations,
  Spatial relations between spatial objects
- Object Relational DB for spatial data
- Spatial SQL
- Spatial indexing
  - Space vs object directed
  - Regular grid, quadtrees & location codes, R-tree
  - Spatio-textual indexing

# Geographical information characteristics

- Location
  - Qualitative: names, postcodes, absolute vs relative
  - Coordinates: lat/long *vs* map grids. Datums
- Geometric objects
  - Points, lines, areas…. OGC Simple Feature geometry model
  - Topological primitives
- Spatial relations between geometric objects
  - Topological :
    - Inside, contains, overlap, touch (meets), equal, disjoint, covers, covered by
    - 9-intersection model of each topological relation
  - Proximal (distance) : qualitative vs quantitative
  - Direction / orientation : qualitative vs quantitative

# Spatial data models

- Raster / location-based

– record what is present within regular grid cells

- Vector / object-based



– Geometric primitives : point, line (lineString, linearRing), polygon (linearRings)…





– Topological primitives: node, arc/link (start node / end node), ring (arcs), face (rings), polyhedron (3D)

– Topologically structured maps : polygon map, network map, triangulated irregular network

# Object-Relational technologies

- Complex data types to represent geometry
  - SQL Create : geometry objects from basic data types (numbers, arrays etc)
- Spatial queries with Spatial SQL
  - Spatial properties : area, length, distance..
  - Spatial relations : topological, distance
  - Geometric intersection (returns geometry)
- Spatial indexing

# Spatial Indexing

- Space-directed vs Object-directed
- Spatio-textual



## Space-directed



– Regular grid



- Location codes based on lower-left corner of cells : concatenation of x/y (vs bit-interleaving)
  - Index location codes with B-tree
- Use for processing queries and updates
- Limitations of regular grid

# Space-directed indexing cont.

- Quadtrees
  - Recursive subdivision of space into quadrants
  - Create location codes for cells using Morton codes / Z-numbers
    - Bit-interleaving of x,y coordinates
  - Issue of unique codes (base 4 vs base 5)
  - Query processing with ‘block’ location codes

Space-filling curves

- desirable properties of location code numbering systems

# Object-directed indexing with R-trees

R-trees extend/modify some aspects of B-trees

Surround geometry objects with minimum bounding rectangles (MBRs) and Create hierarchy of MBRs

– Entry in tree node = MBR + pointer to another node

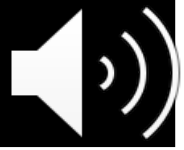
  or (from leaf node) to geometry

– Search by comparing query rectangle with the MBRs in the node entries

– Desirable properties of R-Tree

– Dynamic vs static update methods (static → packed)

# Spatio-textual / Spatial keyword indexing



## Spatial indexing of documents

- Geoparsing → document footprints



## Inverted file indexing of text documents



- dictionary + posting lists



## Combining spatial and inverted file indexing

- separate indexes and merge results
- space first
- text first