Geo875 | FS24
University of Zürich

Spatial Databases

Rolf Meile

Eidg. Forschungsanstalt für Wald, Schnee und Landschaft (WSL)

Swiss Federal Institute for Forest, Snow and Landscape Research

Zhiyong Zhou

Dept. of Geography, University of Zürich

The Geo875-Team

Rolf Meile, Database- and GIS-Specialist, WSL
CV: M. Sc. Biology (UZH), Monitoring Institute (SG), Cityline AG (ZH)
Interests: spatio-temporal data, data quality, dealing with large datasets, multidimensional GIS, fine cooking and eating.
Contact: rolf.meile@wsl.ch

Zhiyong Zhou, Postdoc, GIUZ
CV: BSc. & MSc. GIS (CUG, China), PhD Geog., (UZH)
Interests: spatial cognition and computation, computational cartography, geospatial artificial intelligence (GeoAI), active mobility analytics
Contact: zhiyong.zhou@geo.uzh.ch,Y25J86

Real world...

The Swiss Forest Inventory

- 80'000 trees measured along with 300 attributes
- Stored in a spatially enabled database
- Analysis of the thematic and spatial characteristics of the forest as well as geostatistic;

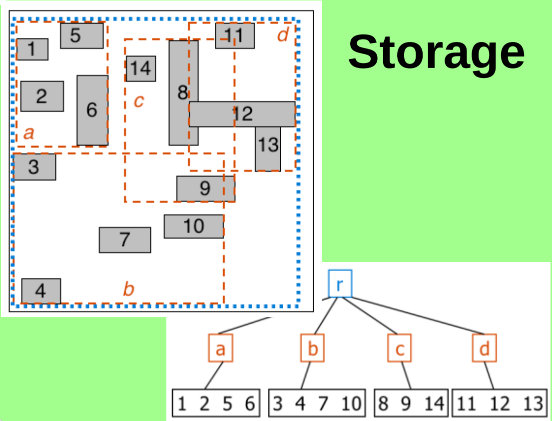
Affective Route Planning

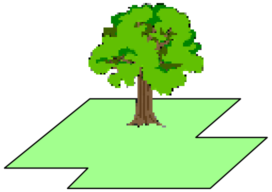
- Integrating and aggregating crowd-sourced datasets
- User based input drives selection of routes
- Routing algorithms suggest 'best' route considering emotions and mood

➔ https://www.lfi.ch/lfi/lfi_film.php

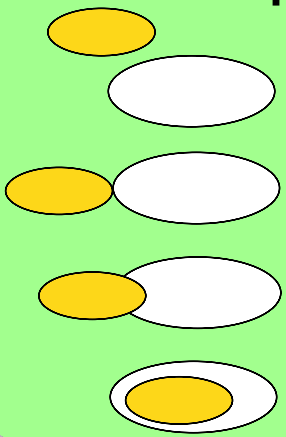
Enabling space in databases

Storage

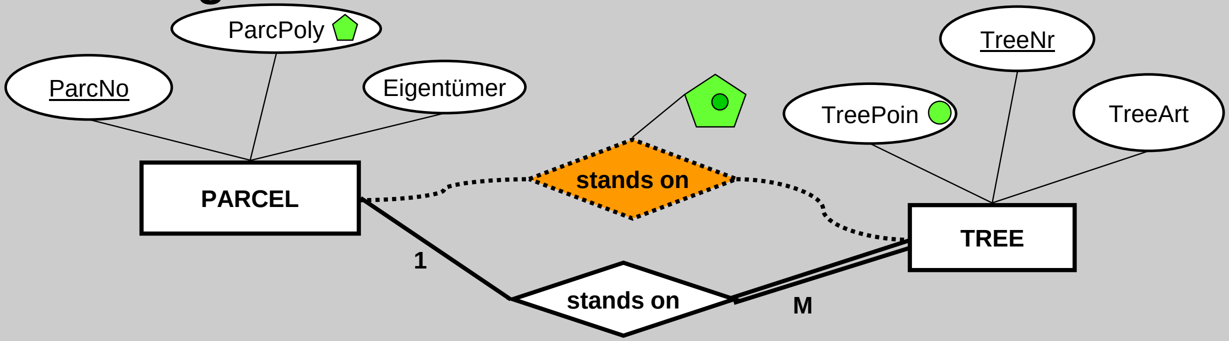




Topology



Modelling



Geo875:
Spatial databases

Geo874:
Introduction to databases

Learning objectives Geo875

- ✓ **"Introduction to spatial databases"** – understand how spatially enabled databases help us better manage spatial data in large, complex projects.
- ✓ Gain proficiency in the fundamentals of spatial data storage and querying in relational DBMS.
- ✓ Master the skills of spatial data modelling and apply them in a project.
- ✓ Understand and correctly apply spatial data types and spatial data storage structures (topology) for the efficient storage and querying of spatial data.
- ✓ Understand spatial indices in spatial databases to achieve efficient storage, modification and retrieval of geometries and rasters.
- ✓ Understand how this principles apply beyond relational DBMS when using web map and geometry servers.

What now?

L1	**Introduction to spatial databases: definitions and concepts**	“What are spatial databases and how are they different to non-spatial DBs?”
L2	**Spatial data types**	“How do we store, query, manipulate, and efficiently manage space in DBs?”
L3	**Topology**	"How can we ensure spatial integrity of our spatial data?"
L4	**Spatial queries**	“How can we define, modify and query a DB?”
L5	**Spatial indices**	“How to speed up answers from a spatial database?”
L6	**Raster data** **Advanded topics** **Recap**	"How can we organise huge or small amount of raster based data?" Recap some interesting bit and pieces on advances in spatial DBs
L7	-	Project delivery

Housekeeping

http://www.krystal-group.com/images/ban-inner/housekeeping.jpg

Housekeeping

https://www.geo.uzh.ch/microsite/geo875/index.html

Online contents:

- Lecture notes
  - Week by week
  - Usually on Thursday noon
- Exercise handouts
- Project description
- FAQ (Problems, Tips and Tricks, Resources)
  1. FAQ-site, project-site
  2. Colleagues
  3. Finally, email us
- Learning objectives and course deliverables

Home Geo875
Program
Project
FAQ
Resources

Spatial Databases Geo875

Lectures, exercises and practical sessions in Fall Semester 2024

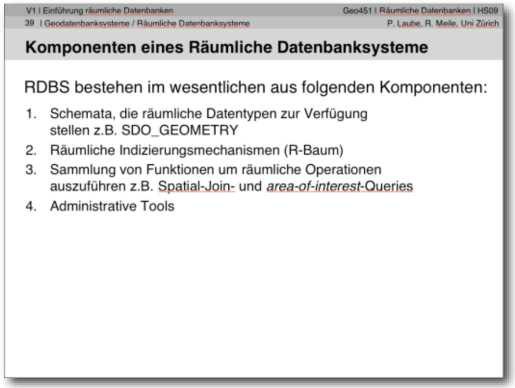
Registration and booking

The course is fully booked for Fall Semester 2024.

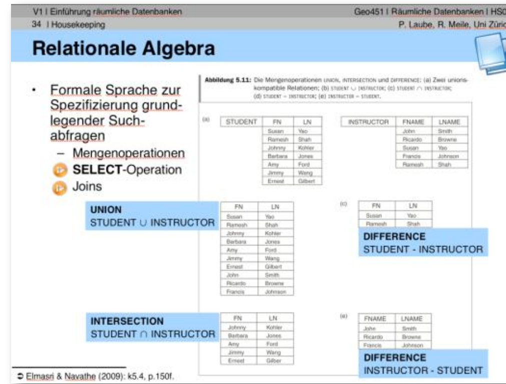
Content and objectives

In this course we extend the knowledge of data management in relational databases acquired in Geo874 (Introduction to Databases) with a focus on spatial data management, usage and analysis. We will introduce spatial indexes, topological data models and spatial queries. Open source software (PostGIS) will be used as an example.

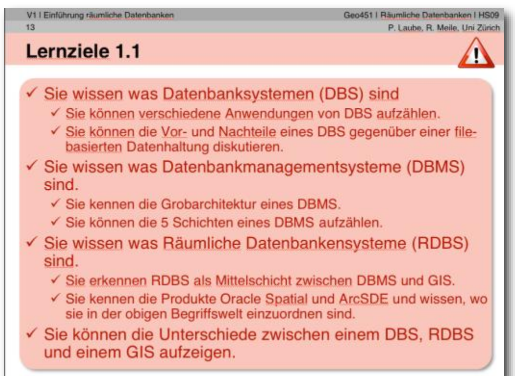
We will use the following colour code



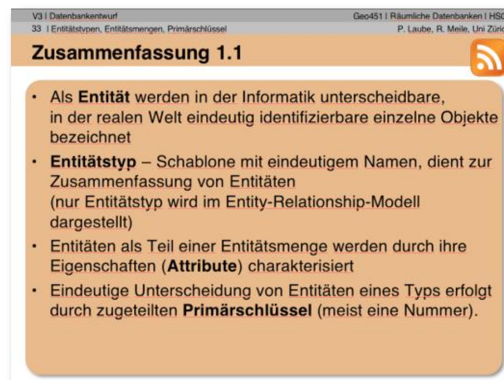
Main contents (just basics, take notes!)



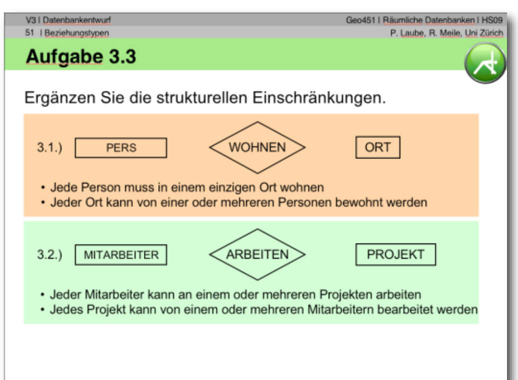
On your own…



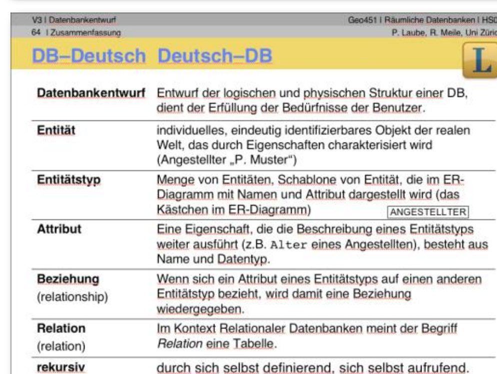
Learning objectives



Summary



Excercises



Terminology, Glossary

Excercises & Practicals

Theoretical Excercises

- Modelling with „Pen and Paper“
- e-learning lessons to expand on lectures' content (GITTA)

Practicals

- Modelling and implementing a spatial project (= group work)
- Application of knowledge in a real-world research project
- Loading and modification of spatial data
- Creating spatial and standard indices
- SQL-Queries (combinations of non-spatial and spatial)
- Apply knowledge from Geo874

Timeplan

Course room

- 8:00 – 8:45 Lecture Part I
- 9:00 – 9:45 Lecture Part II / Theoretical Excercises

Practical room

- 10:15 -12:00 Practicals with demo, exercises, project
- Individual study, work on reports, online tutorials,...

Timetable

20.11. ER-diagram, email pdf proposals to Zhiyong

22.11. Group feedback discussions about ER models

6.12. Feedback discussion with groups on work progress

20.12. Project delivery (pdf) to Zhiyong

Evaluation criteria

See **Geo 875 — Evaluation criteria for FS 2024**
https://www.geo.uzh.ch/microsite/geo875/scripte/geo875_FS24_criteria.pdf

Marking of Geo875

- Group project is graded
- Topics evaluated: **a)** ER-modelling, **b)** realisation in the database and **c)** project final report
- If grade is under 4.0 the whole project can be revised once.

No exam

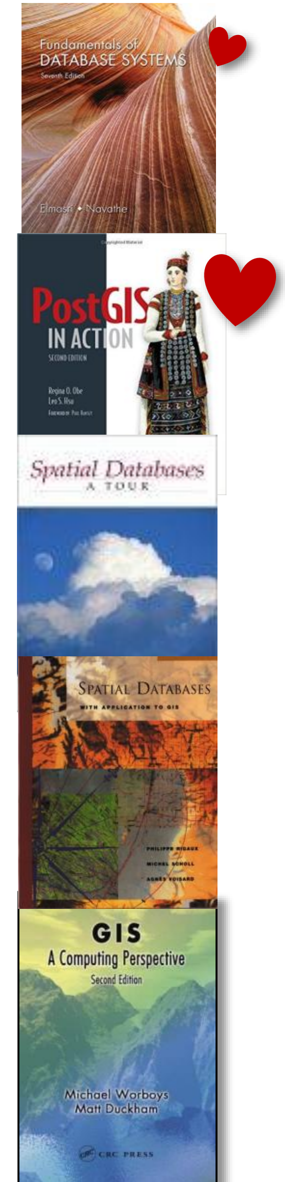
BEWARE OF PLAGIARISM!

Practicals / Exercises

- Not graded. Still possible fail reasons: Regular absences in the exercises; Insufficient participation in group work.

Literature

- Ramez A. Elmasri / Shamkant B. Navathe (2016). Fundamentals of Database Systems 7th ed., Pearson, pp. 1242. ISBN: 978-0-13-397077-7
- Regina O. Obe and Leo S. Hsu (2021). PostGIS in Action, Third Edition, Manning Publications pp. 600. ISBN 9781617296697
- Spatial databases a tour (2003) –
  S. Shekhar and S. Chawla, Prentice Hall,
  pp. 262. ISBN 0-13-017480-7
- Spatial databases : with application to GIS (2002) -
  P. Rigaux, M. O. Scholl and A. Voisard, Morgan Kaufmann Publishers, San Francisco,
  ISBN 1558605886
- Worboys, M. / Duckham, M. (2004). GIS – A Computing Perspective, 2. Issue, CRC Press, Boca Raton FL



e-Learning online

- http://www.gitta.info Swiss virtual campus
- An alternative take on the lectures‘ content
- DE and EN content available
- Links on geo875 Website

- https://postgis.net/workshops/postgis-intro/index.html a comprehensive workshop (originally from the Boundless company)
- we use these samples to build up our exercises in geo875
- datasets available in our database = postgresql with postgis installation
- have a look at schema userdemo in the course database

Geo875 | FS24
University of Zürich

1. Lecture Spatial Databases

Introduction to Spatial Databases: Definitions and Concepts

Rolf Meile
Eidg. Forschungsanstalt für Wald, Schnee und Landschaft (WSL)
Swiss Federal Institute for Forest, Snow and Landscape Research

Zhiyong Zhou
Dept. of Geography, University of Zürich

Learning Objectives

- ✓ Gain a basic understanding of the advantages of managing spatial data in DBMS over file-based spatial data handling;
- ✓ Towards object-relational DB modeling: understand how spatial entity types, their attributes and spatial relationship types are modeled in ER models;
- ✓ Understand how spatial entities, attributes and relationships are implemented in a spatial database
- ✓ Know how to use correct spatial types or subtypes in a spatial database
- ✓ Understand what standardisation is good for in the spatial domain
- ✓ Get a glimpse of simple features

Introduction to Spatial Databases

1. ▶ **Why spatial databases?**
2. Extending relational models to handle space
3. Storing the spatial world (object-relational data storage)
   - Relations with object attributes (geometries)
   - Spatial query methods
   - Implementing spatial relationship consistency
4. Standardisation
   - Why standardise?
   - What to standardise
   - Intro to Simple Features

Why spatial databases?

Recall advantages of non-spatial databases

- *Integrity*: Rules and constraints
- *Independence*: Data is separated from the software
- *Security*: Access control

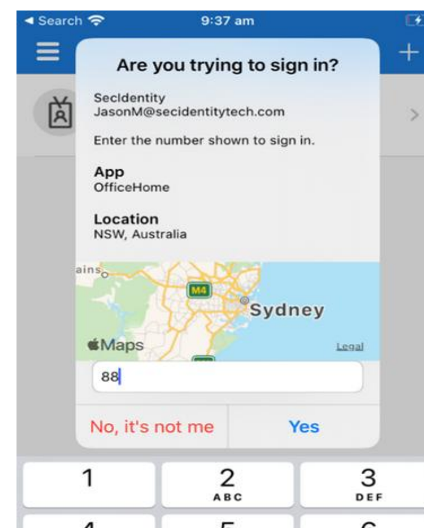
How does this apply to spatial data?

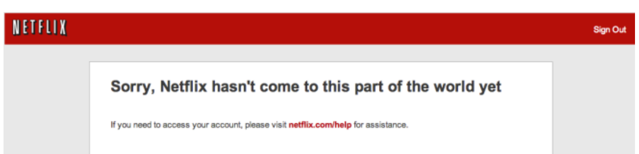
- *Integrity* (conceptual, logical, storage):
  - A specific person unlocks the door of room Y25-H-79; Cars can drive on roads; Parcels do not overlap; Structural: primary keys, foreign keys, unique key, data types, topology, …
  - Storage integrity: attribute and spatial data handled together, not separately (e.g. in different files, different storage locations)
- *Independence*:
  - Spatial database for generic use with many clients: Desktop GIS, Apps, Web, R, Python, …
  - Concurrent use and edits of the same data source

Why spatial databases? cont.

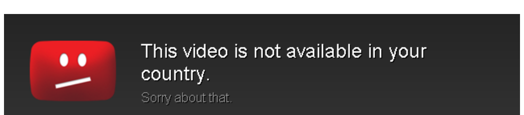
How does this apply to spatial data? cont.

- *Security*:
  - A user can only find out information about object within 50 m
  - Users within a certain space can not have access to certain information (*geofencing*)
  - Users approaching other user beyond a certain threshold are immediately warned
  - Locations of critical infrastructure (company, government) not visible to all user groups
  - Locality of rare Red List species (plants, animals)





Streaming

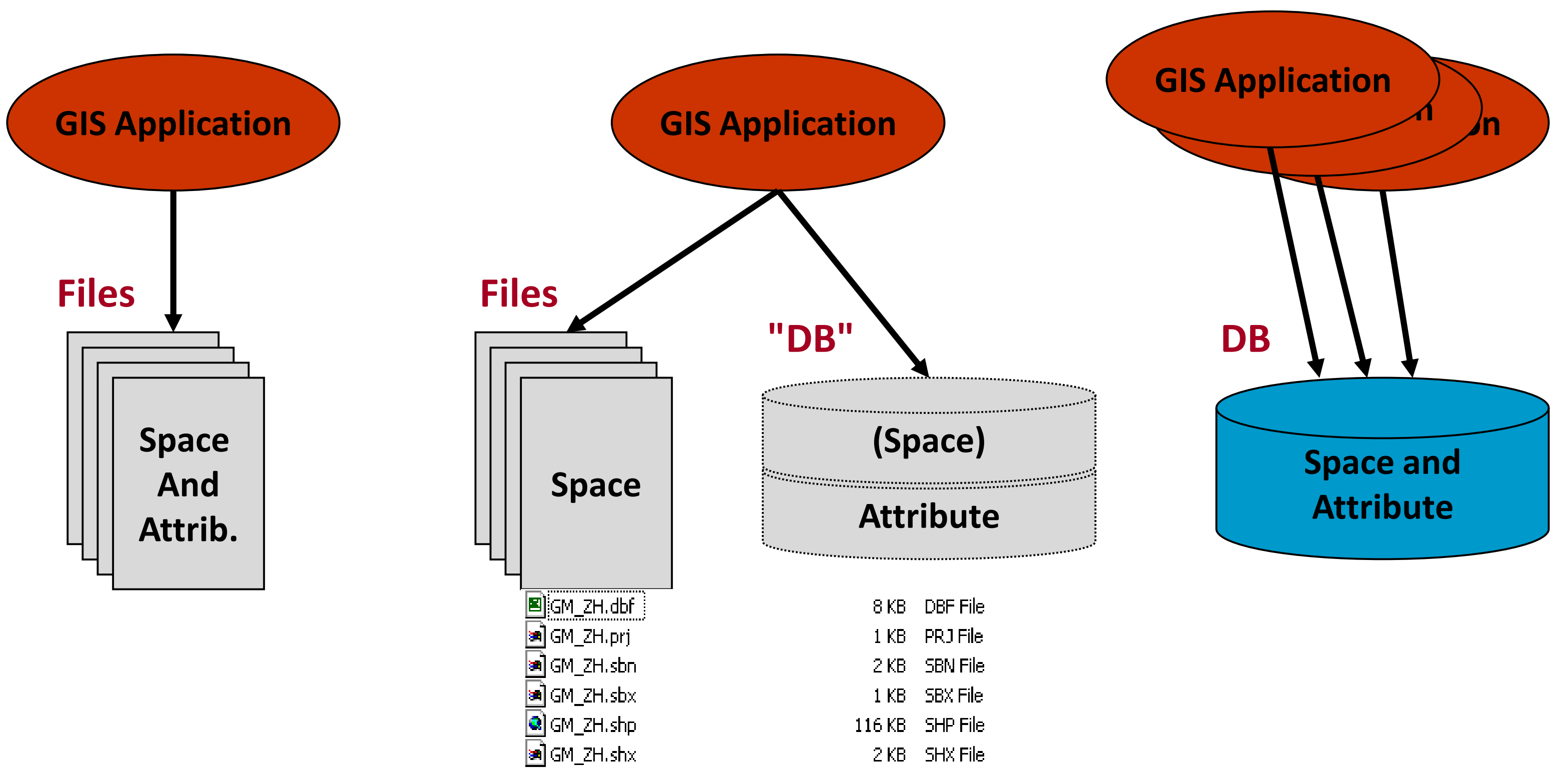


2-Factor authorization incl. location

Why spatial databases (cont.)?

Storage of spatial and non-spatial attributes

1. Files: geojson, GML, Interlis (geometers CH), Geopackage...
2. Files with "DB format" for attributes (e.g. Shapefile), and so-called DBs (e.g. FileGeoDatabase)
3. Real DBs (PostgreSQL, Oracle,…), SQLite



Why spatial databases (cont.)?

Persistent storage of ‘layers’ (=spatial datasets)

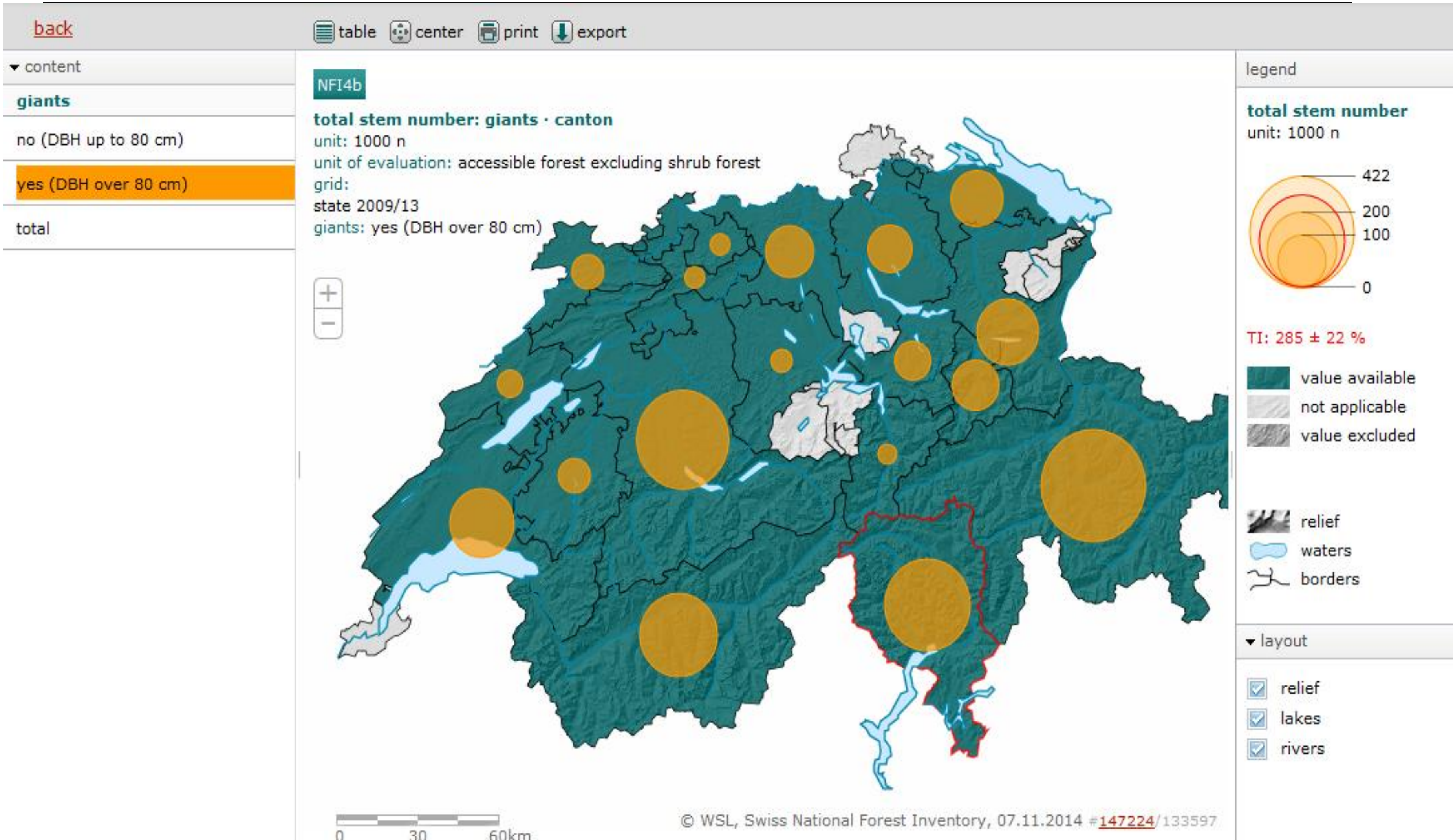
- Altogether with all connected data and metadata chains
- Spatial and non-spatial data is formalized and logically connected
- Reliable data sources; one piece of information at one place;
- On the fly calculating of all kind of relationships

Multiple applications and users access the same spatial data

- Web applications, R analysis software, SQL analysis, GUIs, GISs - all can query at the same time
- Consistent state for all users through their applications
- Editing (means update, insert, delete) is consistent. Troubles can arise with long ongoing transactions.

Data analysis Swiss NFI

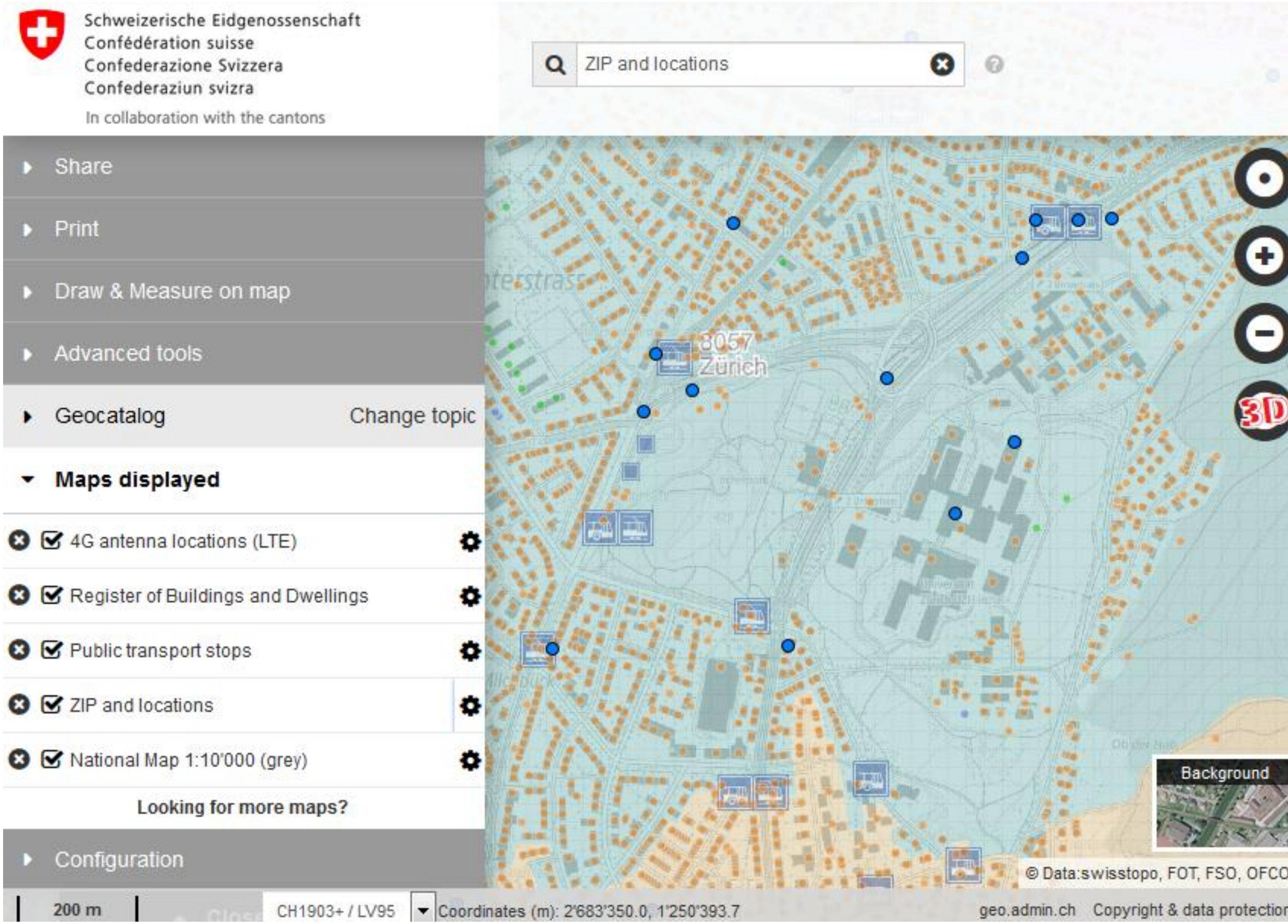
lfi.ch



JU	15257	7	39	52	15296	7

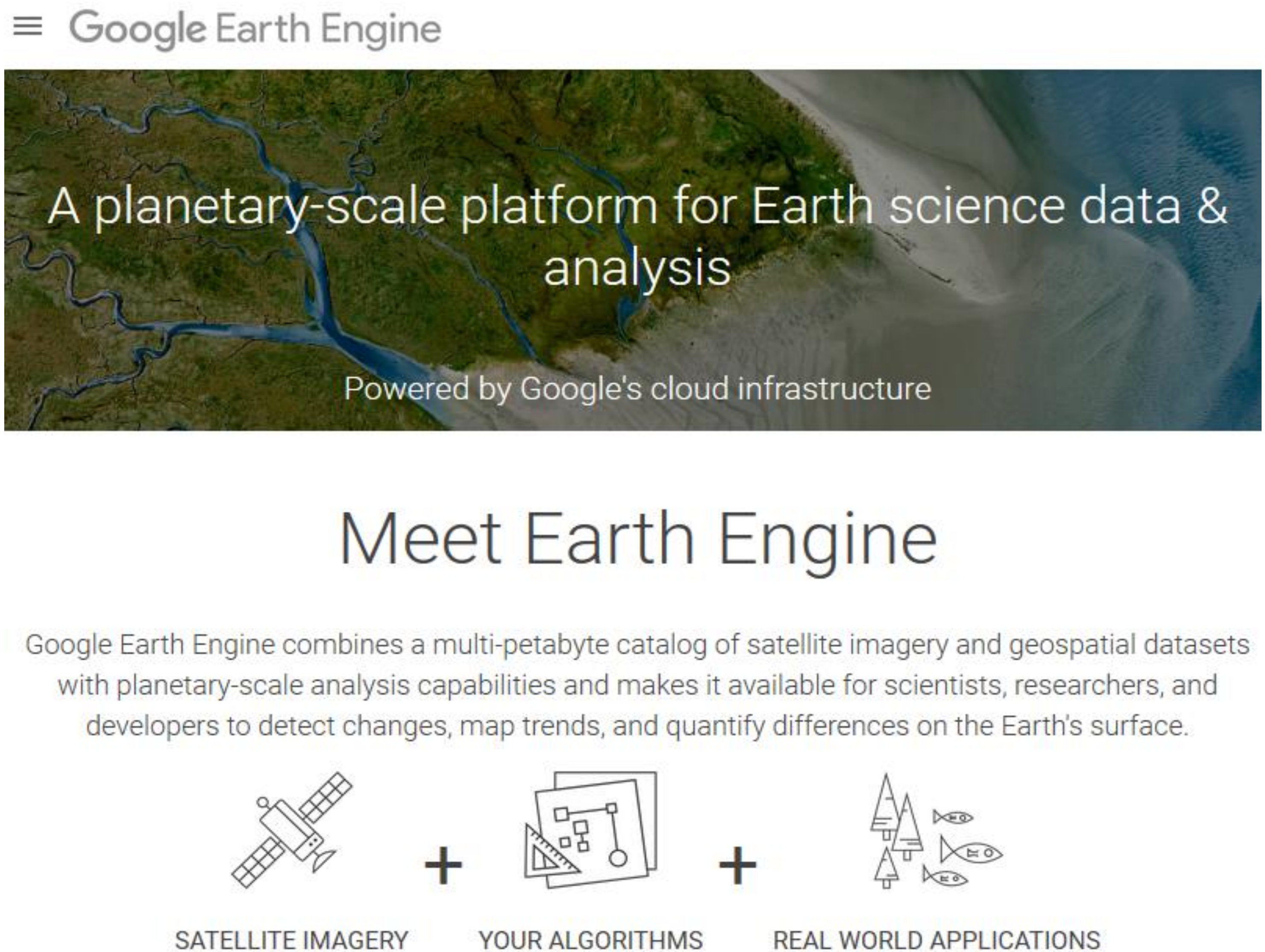
Sharing large data collections

geo.admin.ch



Massive spatial data analytics

earthengine.google.com



Spatial databases @ job

e.g. www.wsl.ch

Specialist in data management (80%)

You will be responsible for the user-friendly recording of forest inventory data as well as their long-term storage within the framework of the projects "Experimental Forest Management" and "Forest Reserve Research". Your tasks will include the development and support of spatial and relational databases (Post-greSQL/PostGIS) as well as the development of tools for data management. These are a data collection software used in

Web-/ Software-Entwickler/-in (80%)

SwissLichens & SwissFungi sind die nationalen Kompetenzzentren für Flechten und Pilze. Die Datenbestände umfassen über 750‘000 aktuelle und historische Fundmeldungen von über 10‘000 Arten. Die Aufgabe der Datenzentren umfasst das Sammeln, Validieren und Publizieren von Verbreitungsnachweisen und Artinformationen. Sie stellen die bestehenden Verbreitungskarten und webbasierten Eingabetools von SwissLichens & SwissFungi auf moderne Technologien um. Sie entwickeln neue Werkzeuge für den Datenimport und programmieren Schnittstellen zur Anbindung von Partnerorganisationen. Für das Datenzentrum Natur & Landschaft entwickeln Sie webbasierte GIS-Applikationen für die Umsetzung der Biodiversitätsstrategie der Schweiz und integrieren Daten zu den Natur- und Landschaftsinventaren des Bundes. Unterhalt und Pflege der Applikationen und Datenbanken gehören ebenfalls zu Ihren Aufgaben.

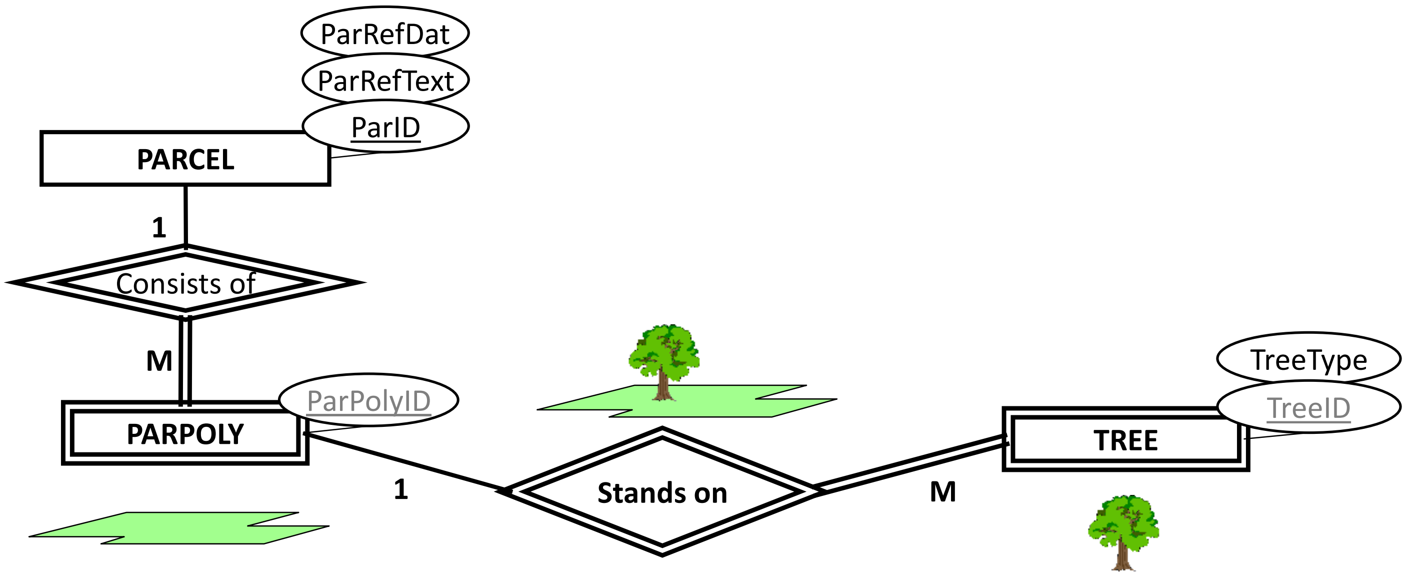
Introduction to Spatial Databases

1. Why spatial databases?
2. **Extending relational models to handle space**
3. Storing the spatial world (=object-relational data storage)
   - Relations with object attributes (geometries)
   - Spatial query methods
   - Implementing spatial relationship consistency
4. Standardisation
   - Why standardise?
   - What to standardise
   - Intro to Simple Features

Exercise 1.1

How would you extend **traditional relational** databases with specific capabilities for storing spatial data? Discuss using the ER example below:

- What is missing in these databases? focus is database
- Which things we need for modelling (ER, Attributes, Relations)? focus is ER modeling



Modeling and storing data about space

Problem levels: ER model -> relations -> DB tables…

1. An entity type has attributes of homogeneous basic values (numbers, strings, dates). But how do you model (ER) and later store (DB) their location, incl. shape & size?
2. Relationships between normal entity types are in one or another way deterministic (each company can have one or more managers). But do a **spatial relationships** behave the same way (each tree can stand on one or more parcels)?
3. Querying: How to do queries based on spatial attributes? How do you (efficiently) find entities with a certain spatial relationship?

Modeling Geometry Data Type

Pure relational: A complex (and complicated) solution; feasible for points only;

Example with country-boundary-contour-point:

Country

name	*capital*	*population*	*id-boundary*
Germany	Berlin	78.5	B1
France	Paris	58	B2
...	...	...	...

```
SELECT   b.id-contour, p.x, p.y
FROM     Country cou, Boundary b, Contour con, Point p
WHERE    cou.name = 'Germany'
AND      cou.id-boundary = b.id-boundary
AND      b.id-countour = con.id-contour
AND      c.id-point = p.id-point
ORDER BY b.id-contour, con.point-num
```

Boundary

id-boundary	*id-contour*
B1	C1
B2	C2
B2	C3
B3	C4
B3	C5
...	...

Contour

id-contour	*point-num*	*id-point*
C1	2	P1
C1	1	P2
C1	3	P3
C1	...	...
C2	1	P4
C2	2	P5
C2	...	...
...	...	...

Point

id-point	*x*	*y*
P1	452	1000
P2	365	875
P3	386	985
P4	296	825
P5	589	189
...	...	...

Foreign-Keys

- each point ever being part of a vertex is explicitly stored in a table
- "models" geometry the "ER way"

Solutions

Prerequisites on the DB side for ER modeling

- Introducing an abstract **geometry** type
- Enabling the **computation** and **storage** of spatial relationships
- Support **efficient spatial querying** and storage with data structures (spatial indexes) and spatial methods.
- all this is implemented in a spatial database (no GIS client/application);

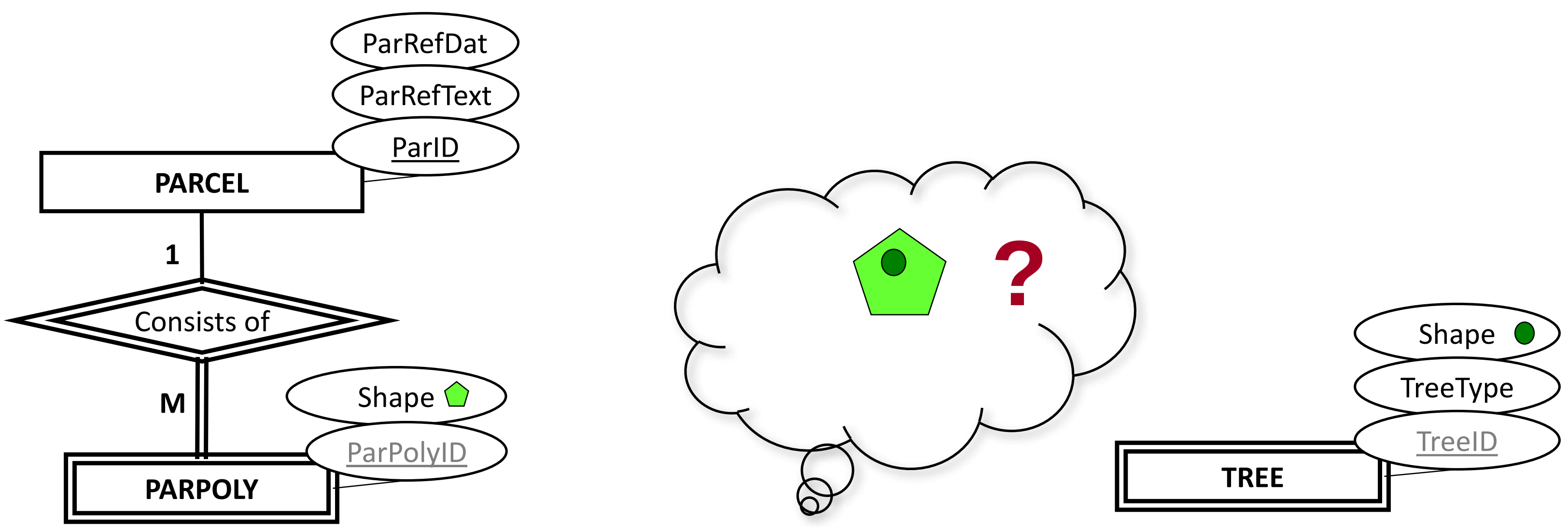
Now we can on the ER side

- model a **spatial entity type** with its geometry attribute
- model a **spatial relationship type** between two spatial entity types (option C in the following slides; -> hardly ever used this way; leads to separate topology implementation)

ER models of the spatial world

Solution in detail:

- **Spatial entity types** are modeled as having a combination of attributes of simple data types and of **one** single geometry data subtype;
- **Geometric attributes** are tagged with an icon of type Geometry;
- **Relationships** – explicit ones or none (see next slides)



Modeling spatial relationships

If two spatial entity types (with their spatial attribute) have a mutual spatial relationship

A) The nature of the relationship is that of a '**strong participation**' or a '**part of**' or '**weak entity type**'.
-> model this with standard ER techniques using cardinalities and participation
-> later in the DB this leads to foreign keys pointing to primary key attributes; spatial querying possible as well;

B) The relationship is **loosely spatial** in the topological sense (e.g., lies in, close to, adjacent to,...) ; no topology needs to be enforced while gathering new data
-> nothing needs to be modeled; leads to standalone spatial table (this is the 'shapefile' or 'layer' paradigm in the database)
--> recall: spatial querying fully available;

C) The relationship is **loosely spatial** in the topological sense; some topology rules, if possible, should be applied to newly inserted data
-> model a spatial relationship without cardinality (-> rather implement topology)
-> could lead to a trigger (little checking program) in the DB

Modeling spatial relationships

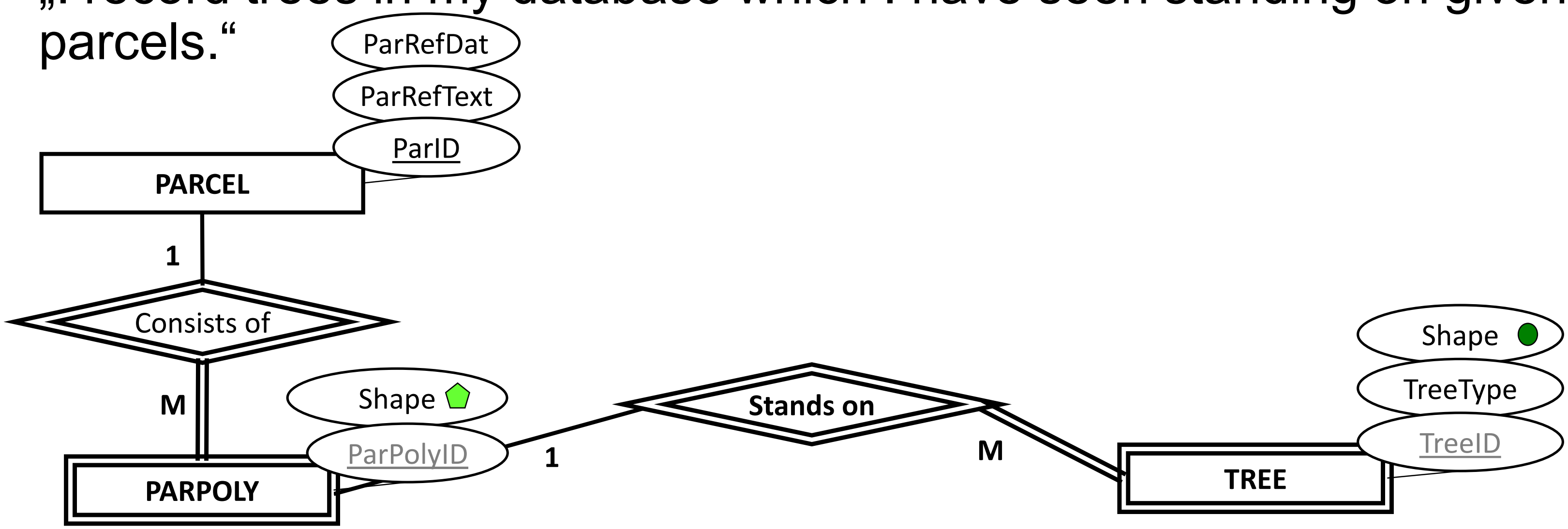
Option A: Strong participation

Relationship type reading as we know from geo874

- „Each tree must stand on exactly one parcel."
- „Each parcel can have one or more trees."

Further descriptions that fit the context

- „A tree can only exist if there is a given parcel polygon on which it can stand."
- „I record trees in my database which I have seen standing on given parcels."



Excercise 1.2

Option A - modelling strong participation

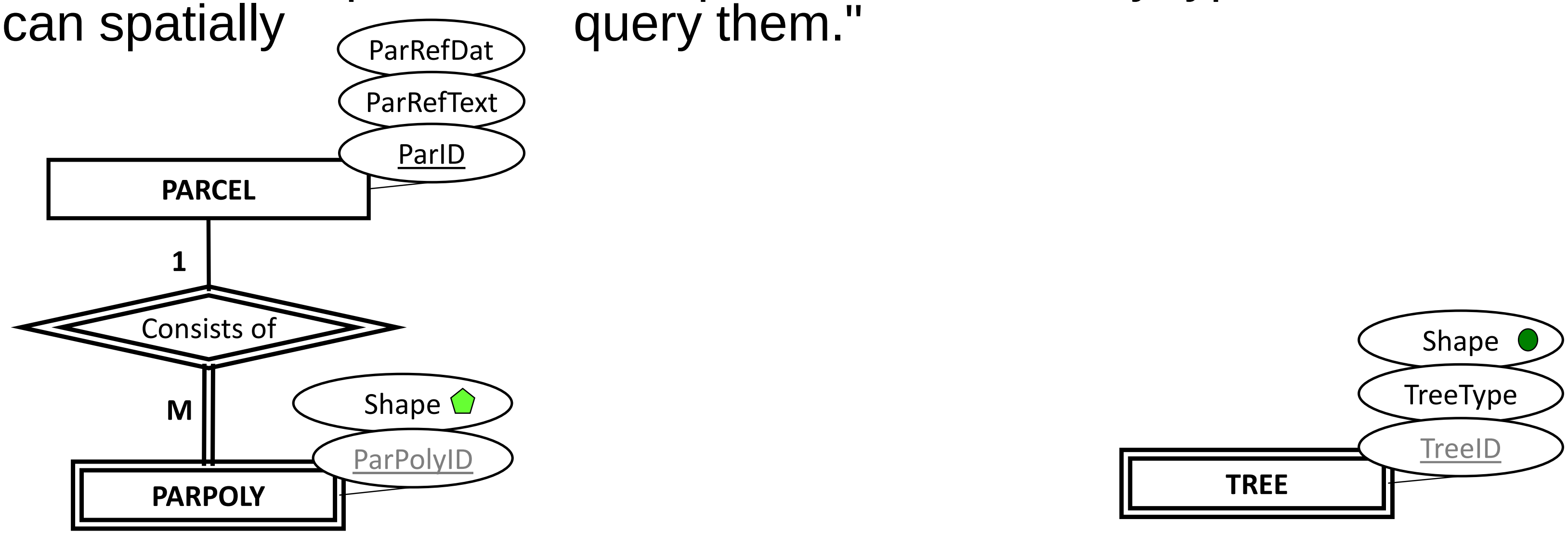
Which main problem could arise when modelling like this? Think of real world data that would be filled into the resulting empty spatial tables. Or if data has to be changed at some point in time.

- Quick discussion with your colleague(s)

Modeling spatial relationships

Option B: Loosely spatial - no relationship

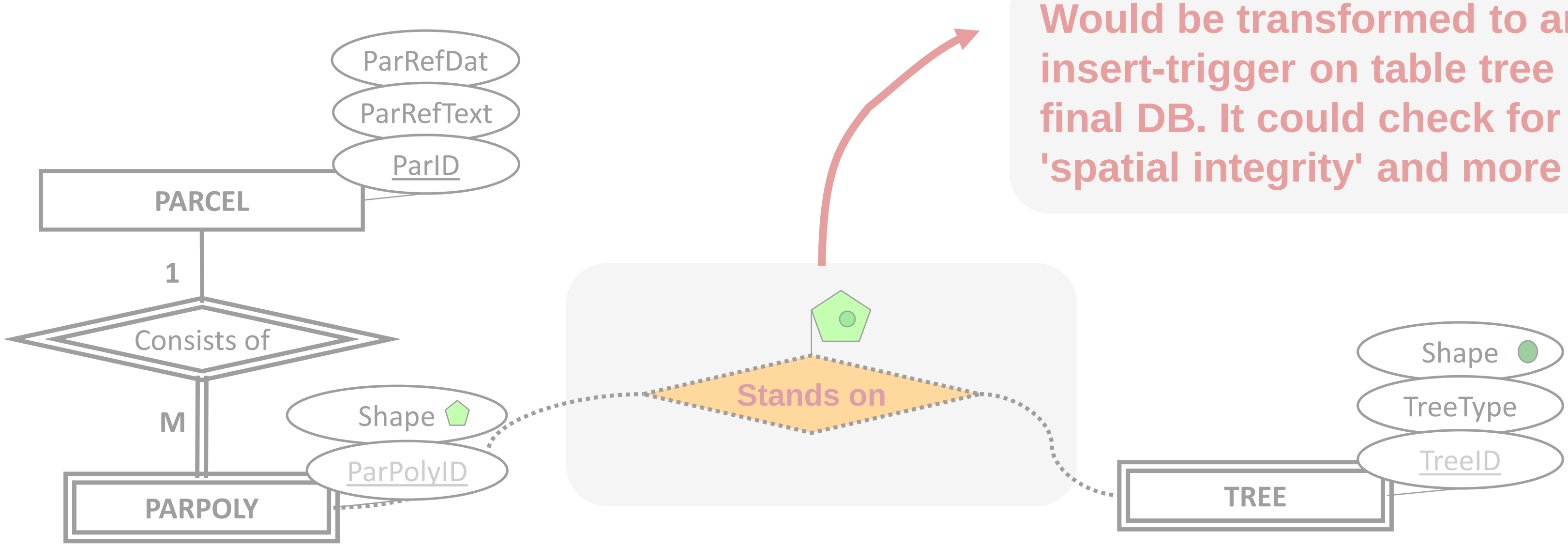
- "Locations of trees were collected independently from parcels spatial and non-spatial attributes"
- "New incoming tree data is not checked regarding to parcels or polygons of parcels"
- "Tree datasets could come from whole Switzerland - parcels could be from the city of Zurich, only"
- "The datasets are loosely spatial because they are spatial! This is implicit and the only common property."
- "There is no explicit relationship between the entity types. But still, we can spatially query them."



Modeling spatial relationships

Option C: Loosely spatial - spatial relationship

- „Each tree must stand on some (existing) parcel.“
- „A tree can only exist when there is some parcel polygon which it can stand on.“
- „I record trees with their coordinates. Tree data is not actively connected with parcel information. Only when entering new tree data into the database we check wheter the trees stand on some parcel. This is done by a spatial comparison with a spatial operator.“
  -> topology implementation (lecture 3)



Would be transformed to an insert-trigger on table tree in the final DB. It could check for 'spatial integrity' and more

Introduction to Spatial Databases

1. Why spatial databases?
2. Extending relational models to handle space
3. **Storing the spatial world (Object-Relational data storage)**
   - Relations with object attributes (geometries)
   - Spatial query methods
   - Implementing spatial relationship consistency
4. Standardisation
   - Why standardise?
   - What to standardise
   - Intro to Simple Features

Storing the DB spatial world – Abstract Data Types

The object-relational model

- **Spatial tables** (spatial relations) have a combination of columns of simple data types and a **single** column of the **geometry data type**
- Displayed in a mapping application a spatial table is called *feature class* or *layer*
- Metadata is specified as part of the geometry type, e.g., spatial reference systems for coordinate storage**.**
- Attributes of type geometry are filled with **object** values
- These spatial objects are **supported by all spatial DBMS** (e.g PostgreSQL with PostGIS)
- **Query methods** and **spatial indices** are supported, too

Geometry as Abstract-Datatype ADT

treepoint	
treepointid (PK)	**bigint**
parzid (PK)	**int**
parzpolyid (PK)	**smallint**
geom	**geometry**
treespecies	varchar(256)

Storing the spatial world – Abstract Data Types

More details on geometry data type and subtypes

- Base geometry type can accept different 'geometries'
- points, lines, polygons up to 3D space and/or metrics
- multipoints, multilines, multipolygons with multiple geometries per record
- geometrycollection with possibility to store heterogeneous geometries in the same spatial attribute of a spatial table; try to avoid;
- Cartesian plane as a base for all calculations

Subtyping, constraining the base geometry type

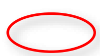
- Check on imported datasets in a DB! Don't expect anything. Check it.
- Modeling fits to database
- Transparency & consistency
- Search for: casting, typmod, type modifier

Subtyped geometry attribute with point and CH1903+ / LV95

treepoint	
treepointid (PK)	**bigint**
parzid (PK)	**int**
parzpolyid (PK)	**smallint**
geom	**geometry(point,2056)**
treespecies	varchar(256)

Storing the spatial world – Abstract Data Types

Keep in mind

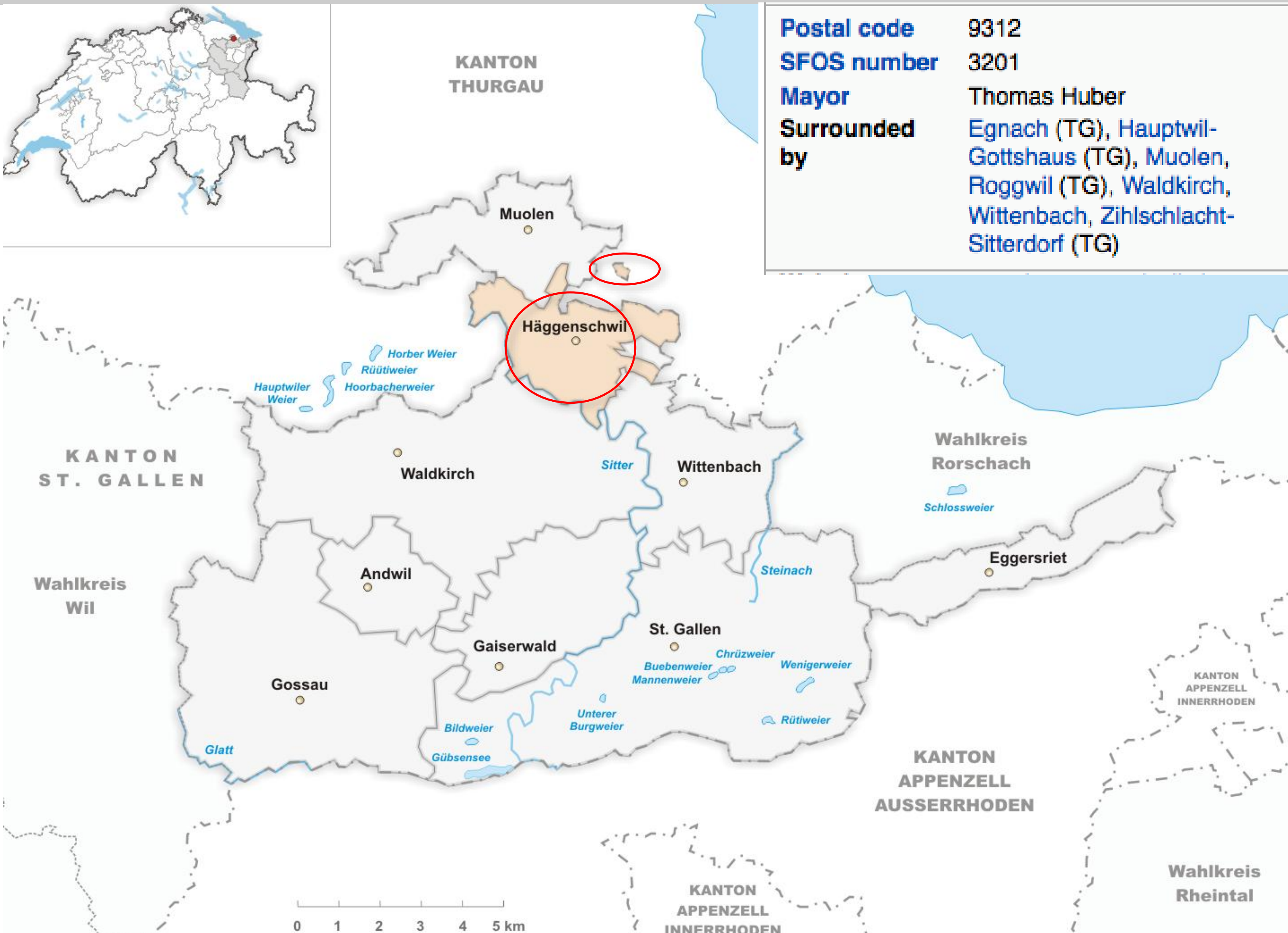
- Geometries are queried through functions (hence, *object*-relational):
  ST_Area(geometry), ST_Centroid(geometry), ST_Distance(geometry,geometry)...
- Geometry data type exposes many functions about the contained geometries: ST_SRID(geometry), ST_Envelope(geometry)...
- Multi geometries behave just as geometries
  ST_Area() works as well as ST_Area()
- Efficient evaluation of some topological queries (ST_Touches, ST_Inside,...) requires ordering of geometries **–> spatial indices**
- Abstract data types and functions can be used in **pure SQL**:

```
SELECT ST_GeometryType(t.geom) FROM treepoint t;
```

More on implementation details of ADTs

- see later in this and next lecture and practicals
- https://postgis.net/docs/reference.html

Community Häggenschwil SG – a MultiPolygon



Postal code	9312
SFOS number	3201
Mayor	Thomas Huber
Surrounded by	Egnach (TG), Hauptwil-Gottshaus (TG), Muolen, Roggwil (TG), Waldkirch, Wittenbach, Zihlschlacht-Sitterdorf (TG)

➲ [4] https://de.wikipedia.org/wiki/Häggenschwil

Modeling Multi or Single (-Polygons, -Lines, -Points)

Method A: Use singlepart geometry subtype

- Each geometric polygon is planned to be saved as a single record for the 'multi-case' depicted in the previous slide
- Leads to **two entity types**: Example Häggenschwil with entity type *community*, relationship type *consistsof*, entity type *communitypoly*.
- Add normal attributes to the appropriate entity type
- Advantage: This **separates semantic meaning** for community **from spatial meaning**!

community	
commid(PK)	**bigint**
commname	varchar(512)

communitypoly	
commid(PK)	**bigint**
commpolyid(PK)	**bigint**
geom	**geometry(polygon,2056)**

Foreign key on communitypoly pointing to community

Modeling Multi (-Polygons, -Lines, -Points)

Method B: Use multipart geometry subtype

- ER model reflects a 1:m relationship in one single spatial entity type; circumstances somewhat hidden in the geometry subtype - but **intuitive**;
- Simply one entity type for community; one has to 'know' that we have stored more than one polygon in one record (!)
- Not all GIS visualisation and calculations work with multipart:
  E.g. ST_Split(geometry1, geometry2)
- If attributes only apply to some polygons being part of a multipolygon record -> use method A

community	
commid(PK)	**bigint**
commname	varchar(512)
geom	**geometry(multipolygon,2056)**

Subtyped geometry attribute with polygon and CH1903+ / LV95

Storing the spatial world – rasters

- **An Abstract Data Type model as well:** multiple tables to reflect raster pieces and metadata; the raster itself is stored in a column of (behind the scenes) type BLOB (Binary Large Object)
- Esri raster type: SDE.ST_RASTER
- Oracle raster type: SDO.SDO_GEORASTER
- Postgis raster type: raster.
- Rasters not necessarily stored in DBs
- More details later in the semester. Rasters are not easy!

Excercise 1.3

ER modeling with spatial data

You get two spatial datasets from the national map agency. One is the community dataset of Switzerland and the second one the cantons of Switzerland. How would you model and finally store these two datasets in a database?

- Discuss with your colleague(s).
- Suggest ER/Relation/Table solution(s)
- What are possible pitfalls or implicit assumptions we have here?

Introduction to Spatial Databases

1. Why spatial databases?
2. Extending relational models to handle space
3. Storing the spatial world (Object-Relational data storage)
   - Relations with object attributes (geometries)
   - Spatial query methods
   - Implementing spatial relationship consistency
4. **Standardisation** ▶
   - Why standardise?
   - What to standardise
   - Intro to Simple Features

Spatial schemas – the subject of standardisation

- **Spatial capabilities** are often an optional "schema" that can be loaded on top of a pure relational database
- Oracles spatial option is included in the SDO schema
- ESRIs spatial extension (on top of many DBs) goes to the SDE schema
- PostgreSQL with PostGIS (spatial extension) is available from the schema named **public**
- These schemata share their types, subtypes, operators, functions, views, tables with us; these are the spatial capapilities of a spatial DB;

```
SELECT PostGIS_full_version();  --identical to
SELECT public.PostGIS_full_version();
```

```
POSTGIS="3.2.1 5fae8e5" [EXTENSION] PGSQL="140" GEOS="3.8.0-
CAPI-1.13.1 " SFCGAL="1.3.7" PROJ="6.3.1" LIBXML="2.9.10"
LIBJSON="0.13.1" LIBPROTOBUF="1.3.3" WAGYU="0.5.0
(Internal)"
```

Standardisation

- Spatial data storage and handling standardised by the Open Geospatial Consortium (https://www.opengeospatial.org/)
- ISO group TC211 is very active on geography/geomatics standards (https://www.isotc211.org/)
- Standardisation contributes to **interoperability;**
- **Meaning 1:** export your DB into an SQL dump, and load into the RDBMS of a different vendor;
- **Meaning 2**: apply your skills with whichever RDBMS;
- Standardisation covers:
  - Definition of Abstract Data Types (ST_Geometry…) and their parts (coordinate pairs, coordinate sequences)
  - Definitions of spatial reference systems, units, etc
  - Formats and notation
  - ST_ method nomenclature as an example. Your scripts should be portable from one RDBMS to another
- …in reality, differences exist. Beware!

Standardisation

- Geodata standards:
  - **Open Geospatial Consortium (OGC)**: One of the most important ones -> Simple Features Specification.
    Parallelled by ISO-19107 and 19125;
  - **ISO**: SQL/MM Part 3: Spatial ISO-13249. Multimedia and Spatial Data;
    Defines abstract data types;
- For DB interfacing (=query language):
  - ISO:SQL-92 up until ISO:SQL-2011 and beyond

Published standards [edit]

The list of standards by the ISO/TC 211 committee:[10]

ISO 6709:2008 *Standard representation of geographic point location by coordinates*

ISO 19101:2002 *Geographic information -- Reference model*

ISO/DIS 19101-1 *Geographic information -- Reference model - Part 1: Fundamentals*

ISO/TS 19101-2:2008 *Geographic information -- Reference model -- Part 2: Imagery*

ISO/TS 19103:2005 *Geographic information -- Conceptual schema language*

ISO/TS 19104:2008 *Geographic information -- Terminology*

ISO 19105:2000 *Geographic information -- Conformance and testing*

ISO 19106:2004 *Geographic information -- Profiles*

ISO 19107:2003 *Geographic information -- Spatial schema*

ISO 19108:2002 *Geographic information -- Temporal schema*

ISO/CD 19109 *Geographic information -- Rules for application schema*

ISO 19109:2005 *Geographic information -- Rules for application schema*

ISO 19110:2005 *Geographic information -- Methodology for feature cataloguing*

ISO 19111:2007 *Geographic information -- Spatial referencing by coordinates*

ISO 19111-2:2009 *Geographic information -- Spatial referencing by coordinates -- Part 2: Extension for parametric values*

Incomplete list of standards by the ISO group TC211

Example: Simple Feature Specification for SQL

- Standardises:
  - Main aspects of spatial data access in a database;
  - Focuses on 0-2D Geometries (not 3D), and only on geometries with linear interpolation between vertices;
  - Defines what are **valid and invalid** geometries
  - Categorises geometry types and **defines their methods and operations**
  - Standardises topology and raster access;
- Thus, enables the development of new methods, exchange of data, sharing of know-how
- Does not prescribe internals – only aspects pertaining to access and exchange!

Very useful: OGC Simple Features Spec 1.2.1:
https://portal.opengeospatial.org/files/?artifact_id=25355
https://portal.opengeospatial.org/files/?artifact_id=25354

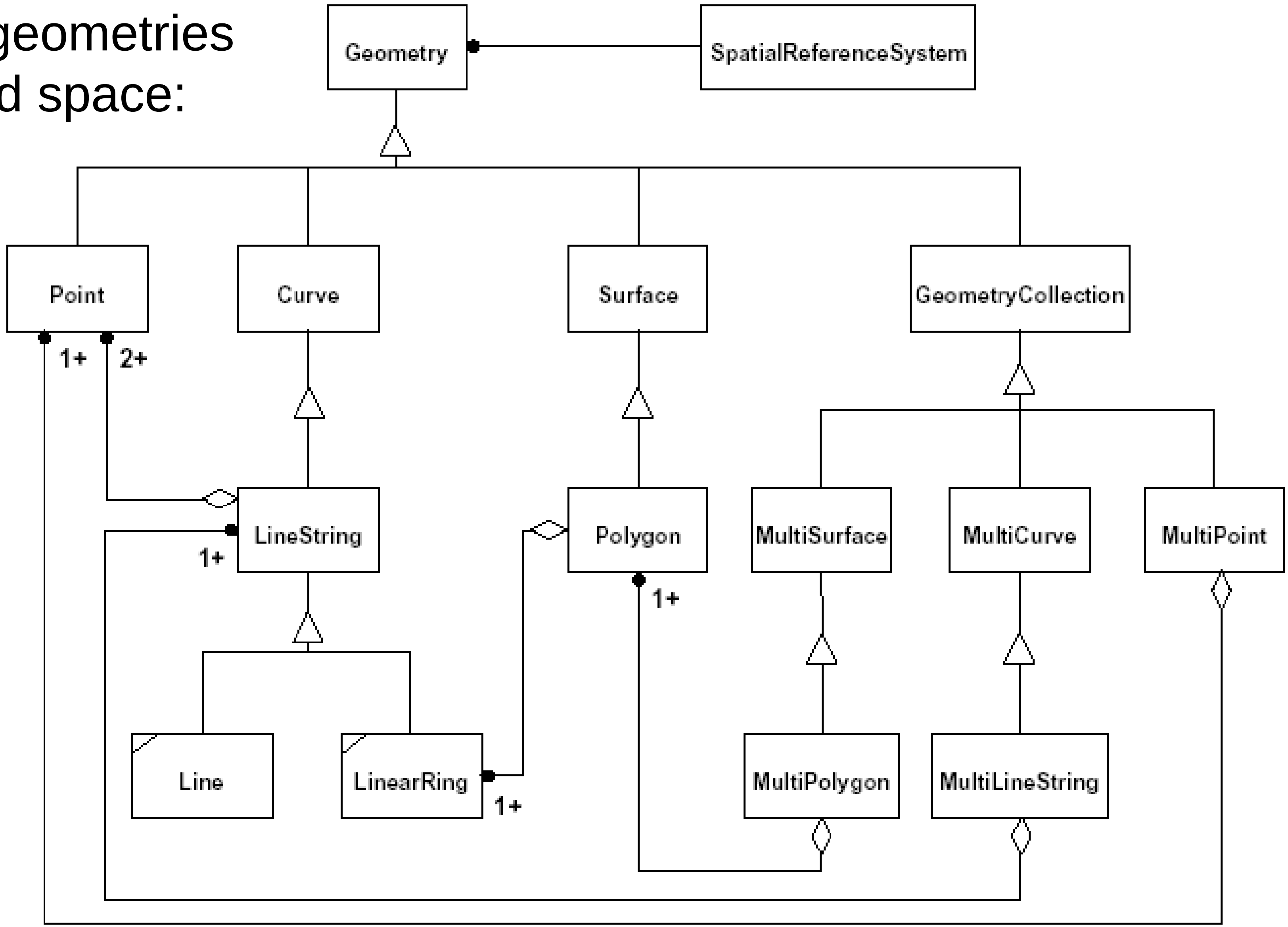
Example: SQL-MM Part 3 (ISO/IEC 13249-3:2016)

- SQL Multimedia Applications Spatial specification
- Standardises concerning geometries:
  - circularly interpolated curves;
  - 3d and 4d geometries; still no srid - as in simple features....
  - what does 3d and 4d mean... **only coordinates**! not geometries
- most parts of that are integrated in Postgis/Postgresql; additional cool stuff, too!

http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=60343

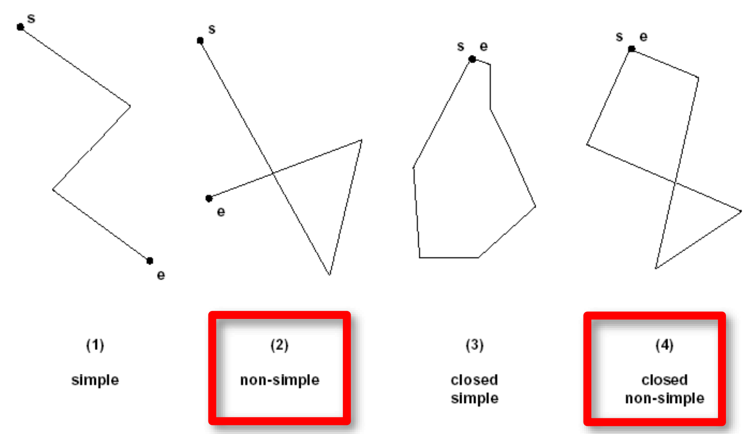
Simple Features

All geometries in 2d space:

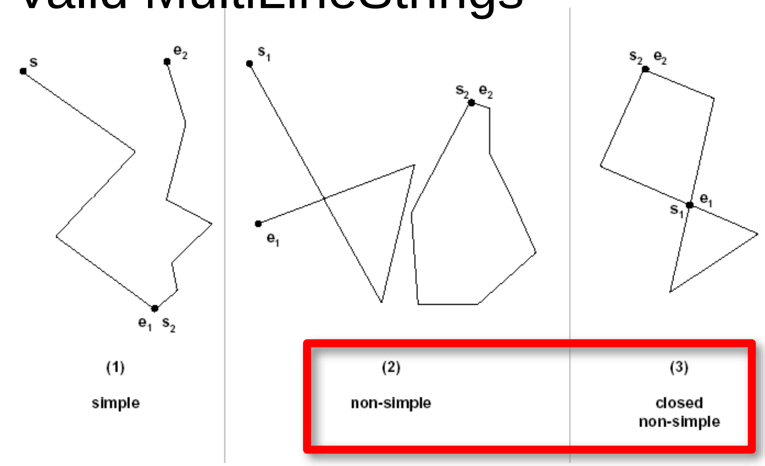


Valid geometries - examples (also simple vs. non simple)

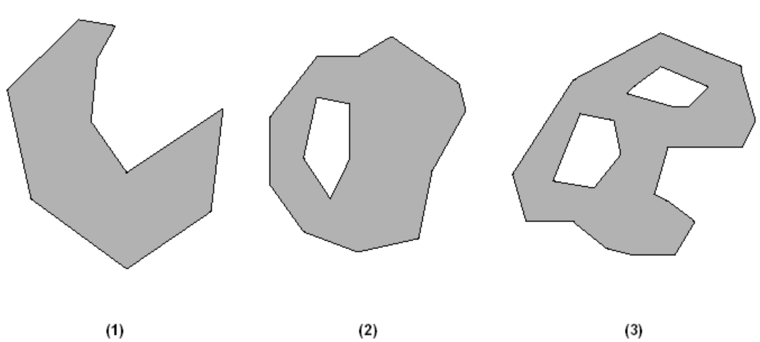
Valid LineStrings



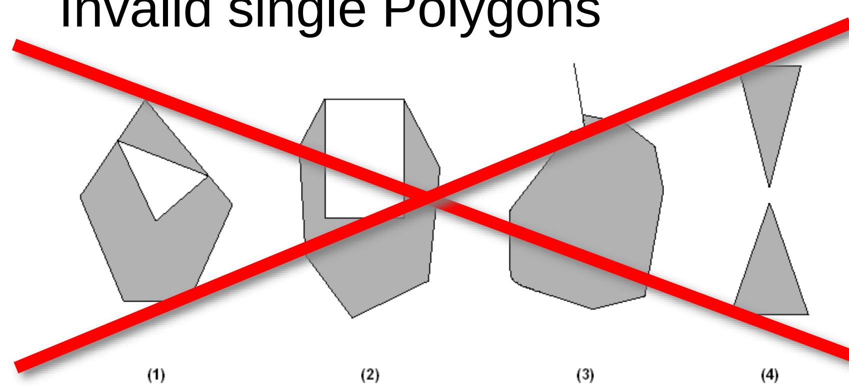
Valid MultiLineStrings



Valid single Polygons

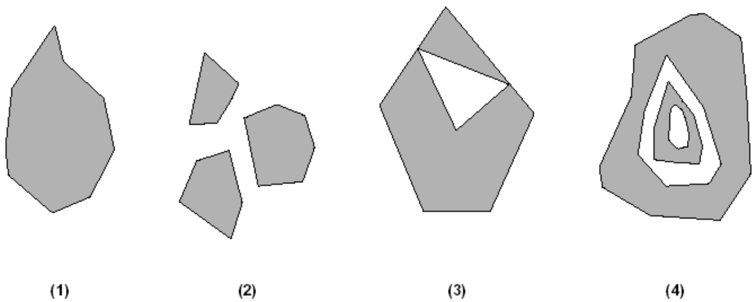


Invalid single Polygons

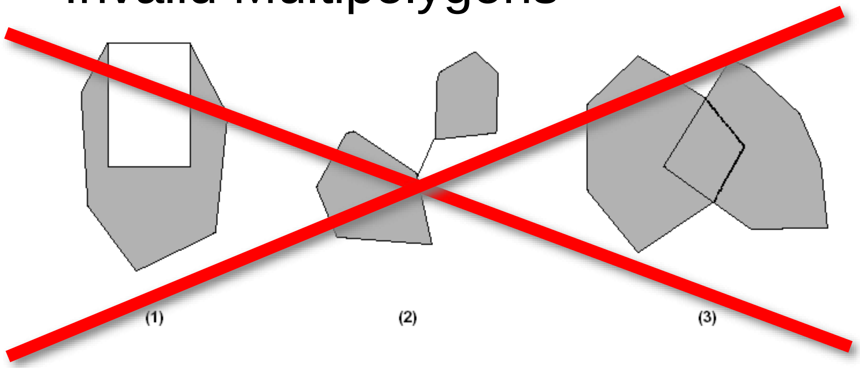


Valid geometries - examples

Valid Multipolygons



Invalid Multipolygons



Standardized Methods (Simple Features)

Geometry constructors

`ST_GeometryFromText(geomtxt)`

`ST_GeomFromGeoJSON(geomjson).....`

Spatial relationships and measurements

`ST_Contains(geomA, geomB)`

`ST_Distance(geomA, geomB).....`

Geometry accessors

`ST_Envelope(geom)`

`ST_GeometryType(geom).....`

Geometry processing

`ST_Difference(geomA, geomB)`

`ST_Union(geomA, geomB).....`

Standardized Methods (Simple Features)

Geometry editors

```
ST_RemovePoint(linestring, offset)
ST_SetSRID(geom).....
```

Linear referencing

```
ST_AddMeasure(geom_mline, measure_start, measure_end)
ST_Line_Substring(a_linestring, startfraction,
endfraction).....
```

Validity

```
SELECT ST_IsValid(ST_GeomFromText('LINESTRING(0 0, 1
1)')) As good_line,
       ST_IsValid(ST_GeomFromText('POLYGON((0 0, 1 1, 1
2, 1 1, 0 0))')) As bad_poly;
```

Summary 1

- We need to extend the relational model to handle space
  - Geometry attributes (Geometry objects, reference systems)
  - Spatial relationships (topological, range,...)
  - Spatial methods for (efficient) spatial queries (supported by spatial indexes)
- Spatial relationships can be stored either by:
  - Explicitly storage - strong participation, weak entity (with known cardinality);
  - no relationship
  - spatial relationship
- Spatial data handling in DBs is standardised by OGC or ISO
- Simple Feature specification
  - Valid and invalid geometries
  - 0D,1D,2D (Multi)geometries

Deadlines

Timetable and deadlines

- 14.11.2024 (Thu) - Deadline for cancellation of module booking
- 20.11.2024 (Wed 11:59 pm) - ER-Diagram proposals to **Zhiyong Zhou** (see the above Communication rules)
- 22.11.2024 (Fri) - Group feedback discussions about ER models
- 06.12.2024 (Fri) - Short oral report and discussion about current state of the project
- 20.12.2024 (Fri 11.59 am) - Project report delivery (digital PDF version to **Zhiyong Zhou** (see the above Communication rules) and finalization of database modeling