Last week

- We have discussed indexing and how it helps to speed up searches (selections) in databases
- We have discussed B-Trees as an example of indexing on non spatial data (ordered)
- We have expanded this concept to 2+D data (spatial data):
  - The problem of ordering in 2+ dimensions
  - Spatial indices driven by space or data (Grid/Quad trees, R-trees)
- We have discussed how indexing helps speeding up filtering (select) performance in a filter-refine paradigm

Geo875 | FS25
University of Zürich

6. Lecture Spatial Databases

Raster Data

Rolf Meile

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

Swiss Federal Institute for Forest, Snow and Landscape Research

Esra Suel

Dept. of Geography, University of Zürich

Learning Objectives

- Understand the theory of raster data storage and querying in spatial databases
- Understand what parameters of raster data need to be stored in order to assure efficient data management
- Get a glimpse of how spatial databases are used to manage raster data
- See how raster data services can be delivered with web servers

Overview

1. **Principles of Raster Data Storage**
2. Loading Raster Data
3. Raster Data in Web Services

Exercise 1 – What are raster data

You already dealt with raster data in GIS. Discuss the following points, and come with some answers:

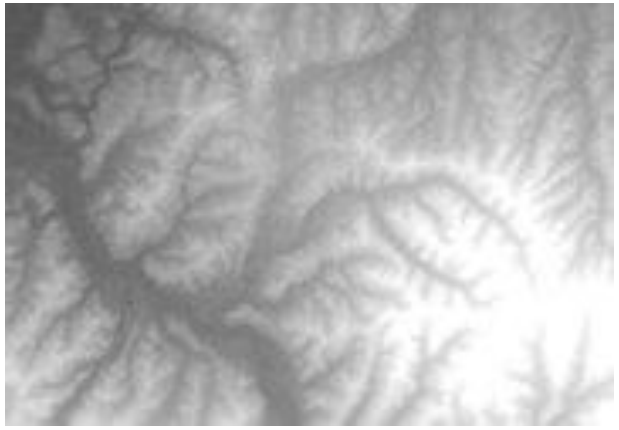
a) Give 3 examples/types of datasets that come as rasters;

b) What technologies are used to collect raster data?

c) What are the common parameters of raster data you may need to know in order to work with them?

Sources of raster data

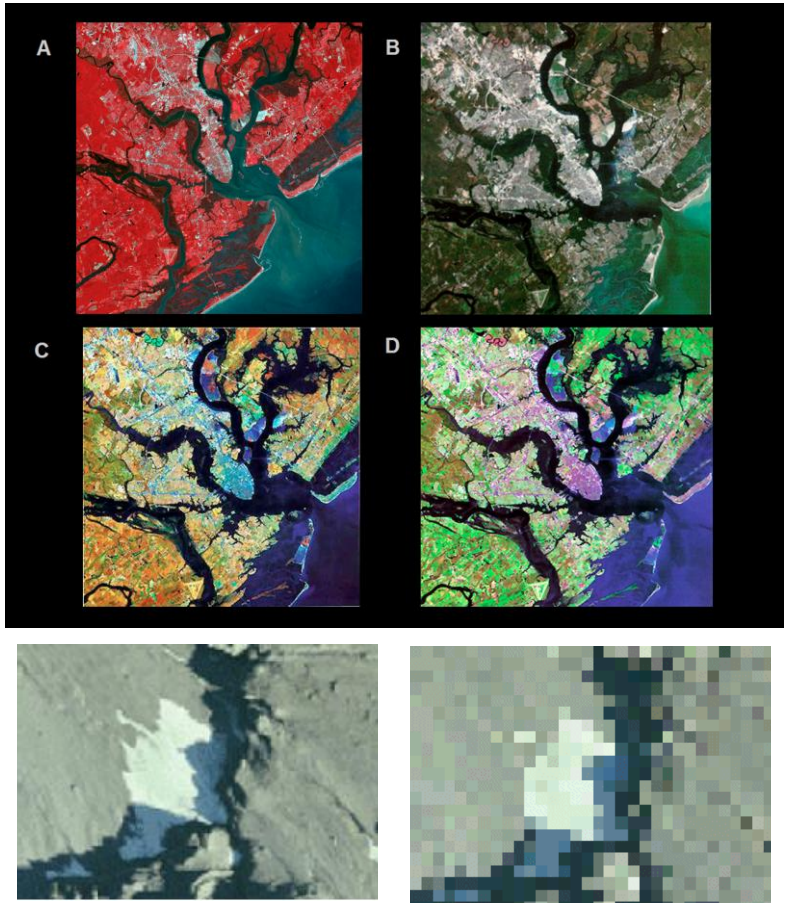
- Raster data are often used to represent *field* data models:
  - A value available for any location in the covered space;
- Raster data are commonly collected by:
  - Air photogrammetry and remote sensing sensors (visible spectrum, near IR, …)
  - Scanning old maps
  - As the result of analysis and derivation from e.g., entity based models (by interpolation)



Illustrations GITTA: http://www.gitta.info/DataCompress/en/html/unit_rastercomp.html

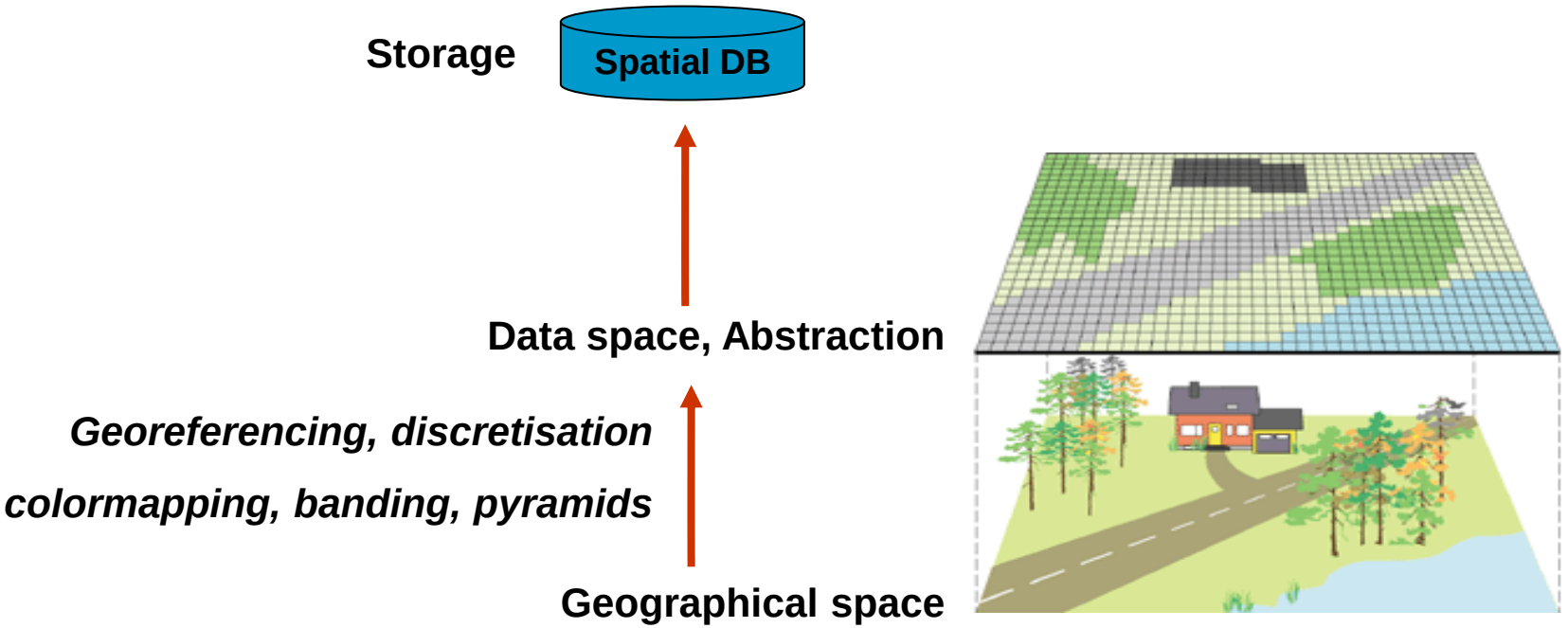
Characteristics of Raster Datasets

1. Cells (pixels)
2. Spatial domain (extent) – specified as bounding box
3. Resolution (x/y ground dimensions of the cell – in m or deg)
4. Potentially, temporal resolution (revisitation interval)
5. Spatial, temporal and band information (e.g., cell indexing direction and origin)
6. Cell values – discrete (pixel maps) or zones (semi-discrete)
7. Additional metadata and supporting data (e.g., masks)



Illustrations: GITTA & http://www.oneonta.edu/faculty/baumanpr/geosat2/RS-Introduction/RS-Introduction.html

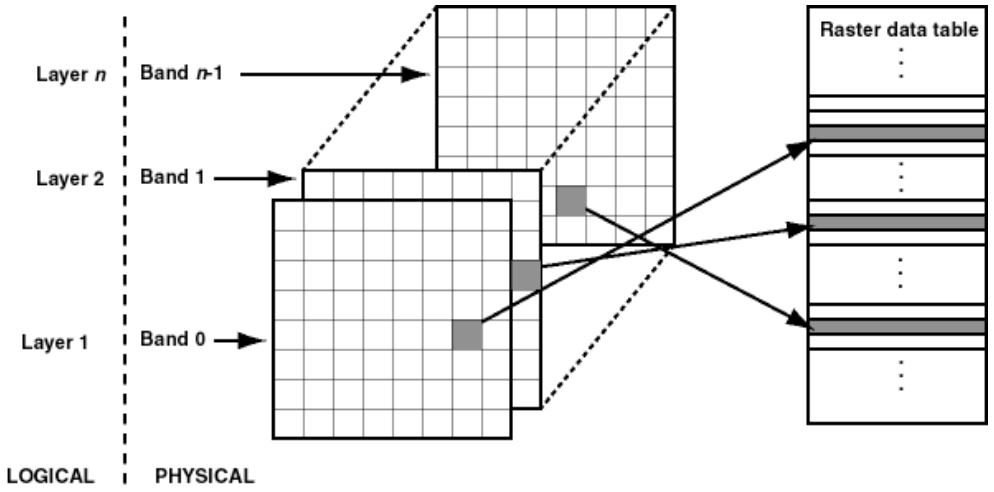
Representing continuous fields

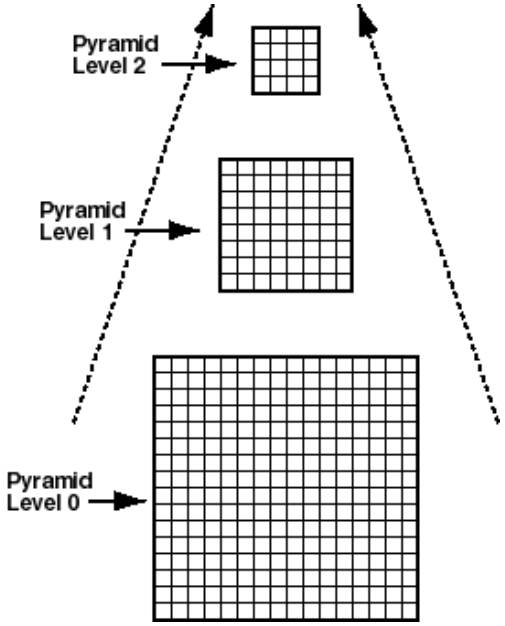


- How are these data stored in the DB?
- How are they managed, indexed, what is the performance and which tools are used?

Conceptual Model of Raster Storage in DB

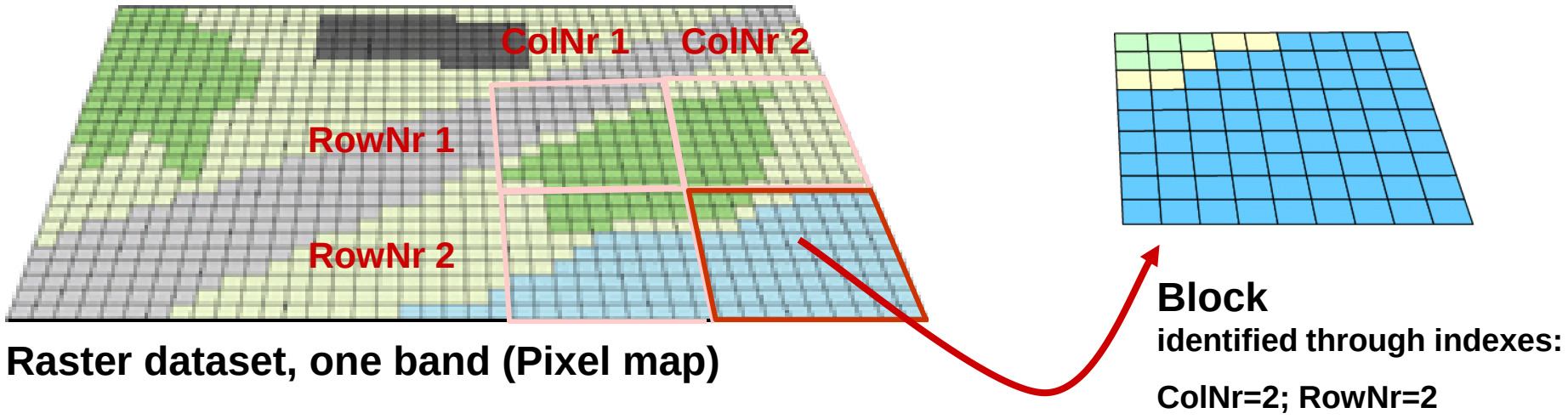
- Cell – element of the grid (=pixel). Each pixel has an assigned value (colour, height, temperature), size
- Block: set of pixels of given size (e.g.128x128 px)
- Layer: logical groupings of cells (e.g., Temperature); a band is a physical concept (e.g., Red band); layer is a logical concept
- Pyramid: **derived** representations for increased performance





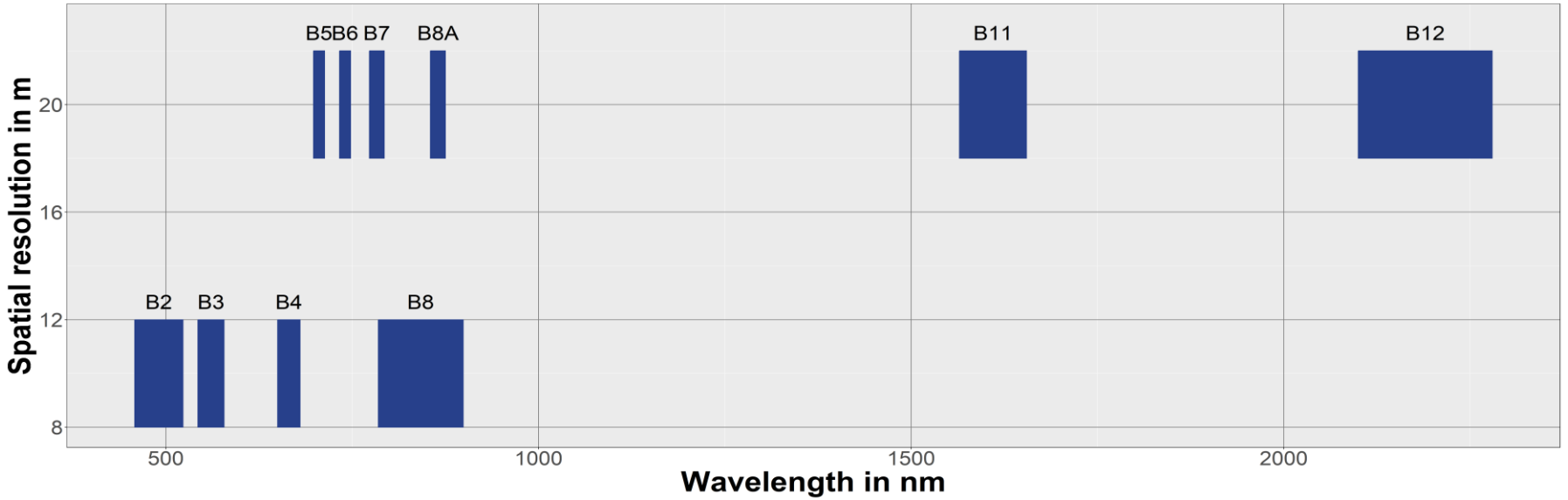
Raster Dataset and Blocks

- Implicit topology – based on cell indexing
- Maintained for performance/handling memory reasons
- Blocks are the basic elements for storage in the DB
- Example of a 9x9 block with 81 cells:



Raster Dataset and Bands - Sentinel-2

- Multispectral
- 13 spectral bands
- 10 & 20 m spatial resolution
- 3-4 days revisiting time in CH

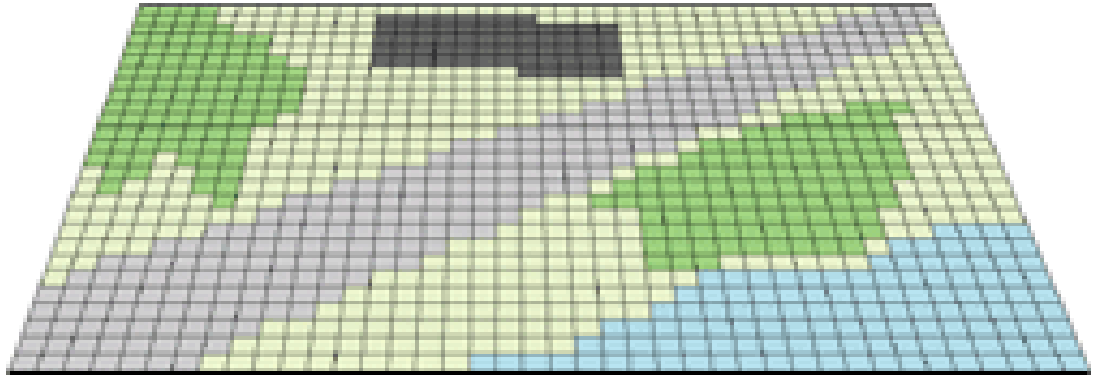


Kudos to Tiziana Koch, WSL

Values of Pixels

Possible approaches to store colour values:

- Colour maps, discrete: colour values are stored as indices pointing to a colour space with discrete colours
  - Good compression
- Colour shades (semi-discrete): e.g., 8 Bit storage of “shades of grey” (256 values) Sentinel-2: 12 Bit storage per channel
  - Better applicable for continuous changes of shades (photography).

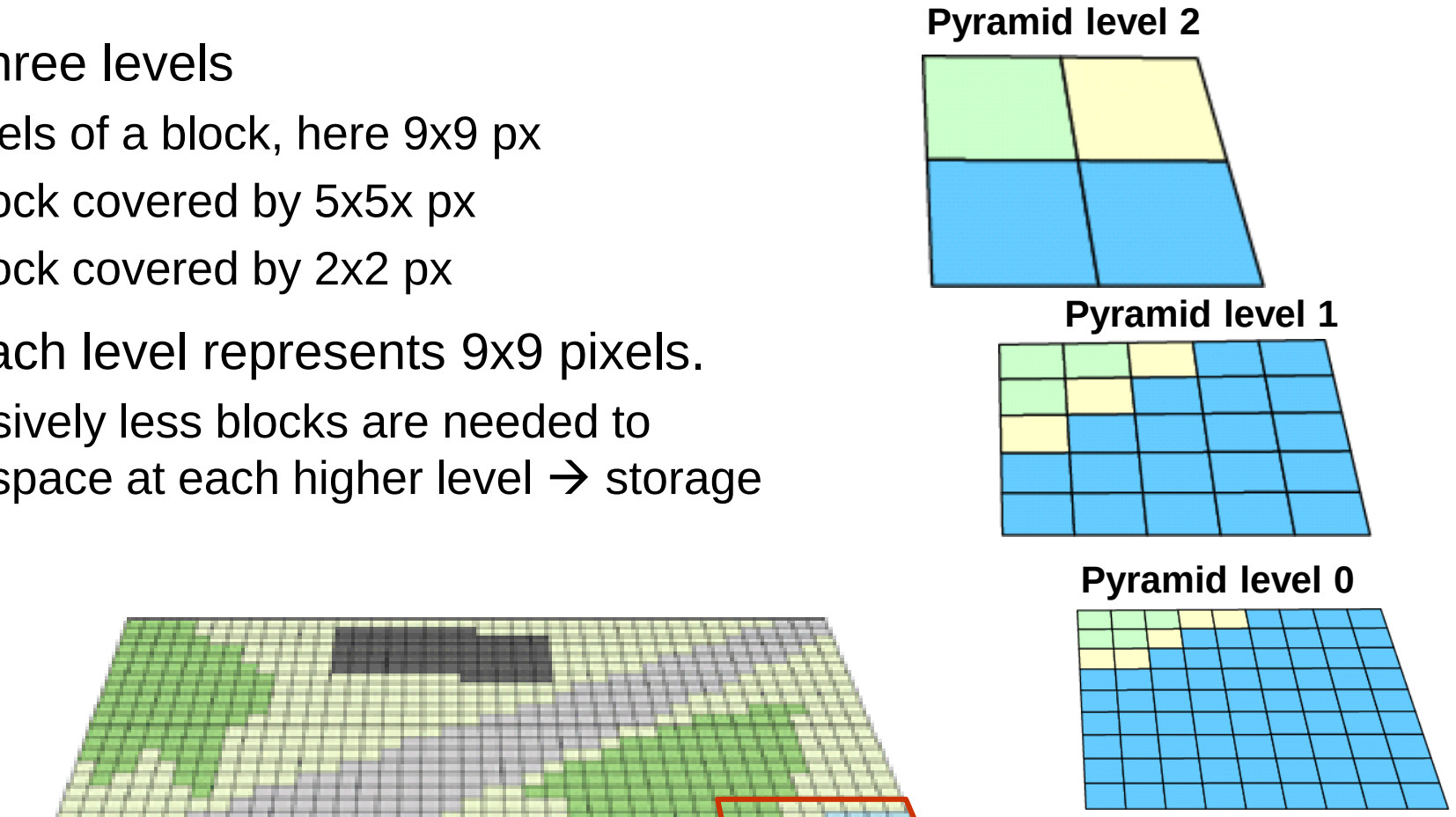


Band (Layer) for which a Colour map with 5 indexed values

1 – Green (Forest)
2 – Grey (Road)
3 – Dark Grey (Building)
4 – Yellow (Field)
5 – Blue (Water)

Raster Data and Derived Pyramids

- A Pyramid with three levels
  - Level 0: all pixels of a block, here 9x9 px
  - Level 1: L0-Block covered by 5x5x px
  - Level 2: L0-Block covered by 2x2 px
- Each Block on each level represents 9x9 pixels.
  - Thus, progressively less blocks are needed to represent the space at each higher level → storage efficiency



Raster dataset: Pixel map with a single Band (with Colour map)

Overview

1. Principles of Raster Data Storage
2. **Loading Raster Data**
3. Raster Data in Web Services

Exercise 2 – Storing raster data

Your friend studying physical geography has a drone allowing to take aerial photographs. She would like to access these data through a Web-based application. She sent you her *.jpg files to help her store them in a spatial DB or in a file geodatabase mosaic dataset.

Discuss:

a) Which **parameters** do you need to know so that you can even start with the task?

b) What steps do you think you have to take to store these raster data into a Geo-DB?

c) Some **algorithms** will have to be executed to process the data during the process. Can you think of what these will be?

Raster Model (Example: PostGIS with PostgreSQL)

- Raster **data type** through postgis_raster extension
- Can be applied as a column with type raster

```
CREATE TABLE my_rasters
(
  rid         serial primary key,
  name        varchar(128),
  rast        raster
);
```

- This table then stores all the tiled (chunked) rasters
- Different bands stored in the same table
- Storage internal or external (out of db = file based)
- Blockwise binary storage of pixels
- Implementation hidden behind the raster type
- Constraining of this table through a function

Raster Model - Constructors (PostGIS with PostgreSQL)

- Convert PostGIS geometries to rasters with ST_AsRaster.
- Load rasters with the raster2pgsql.exe loader
- Create rasters from scratch using ST_MakeEmptyRaster and ST_AddBand, and then set pixel values using various other raster functions
- Build rasters from existing rasters using processing functions such as unioning, tiling, map algebra, reclassifying, resizing, reprojecting, resampling, and so forth
- Use the PostGIS function called ST_FromGDALRaster, which allows you to convert rasters in various formats to PostGIS rasters
- No loading functions implemented in QGIS

Raster Data Model (Example: Oracle Spatial)

e.g., map sheets of the 1:25'000 maps of Switzerland

Raster info table

RASTERTEST

	Column	Type	Nullable	Default	Comments
123	**ID**	**NUMBER(3)**			
ABC	TEXT	VARCHAR2(128)	Y		
OBJ	GEORAST	MDSYS.SDO_GEORASTER	Y		

	Key	Column(s)	Type
	GEORASTERTEST_PK	ID	P

MDSYS.SDO_GEORASTER

	Element	Type
	RASTERTYPE	NUMBER
	SPATIALEXTENT	MDSYS.SDO_GEOMETRY
	RASTERDATATABLE	VARCHAR2(32)
	RASTERID	NUMBER
	METADATA	SYS.XMLTYPE

Indexable

Metadata

Block table (per Band, per Pyramid)

RATEDATA

	Column	Type
123	**RASTERID**	**NUMBER**
123	**PYRAMIDLEVEL**	**NUMBER**
123	**BANDBLOCKNUMBER**	**NUMBER**
123	**ROWBLOCKNUMBER**	**NUMBER**
123	**COLUMNBLOCKNUMBE**	**NUMBER**
OBJ	BLOCKMBR	MDSYS.SDO_GEOMETRY
101	RASTERBLOCK	BLOB

	Key	Column(s)
	RATE_PK	RASTERID, PYRAMIDLEVEL, BANDBLOCKNUMBER, ROWBLOCKNUMBER, COLUMNBLOCKNUMBER

Pyramid Level

Band

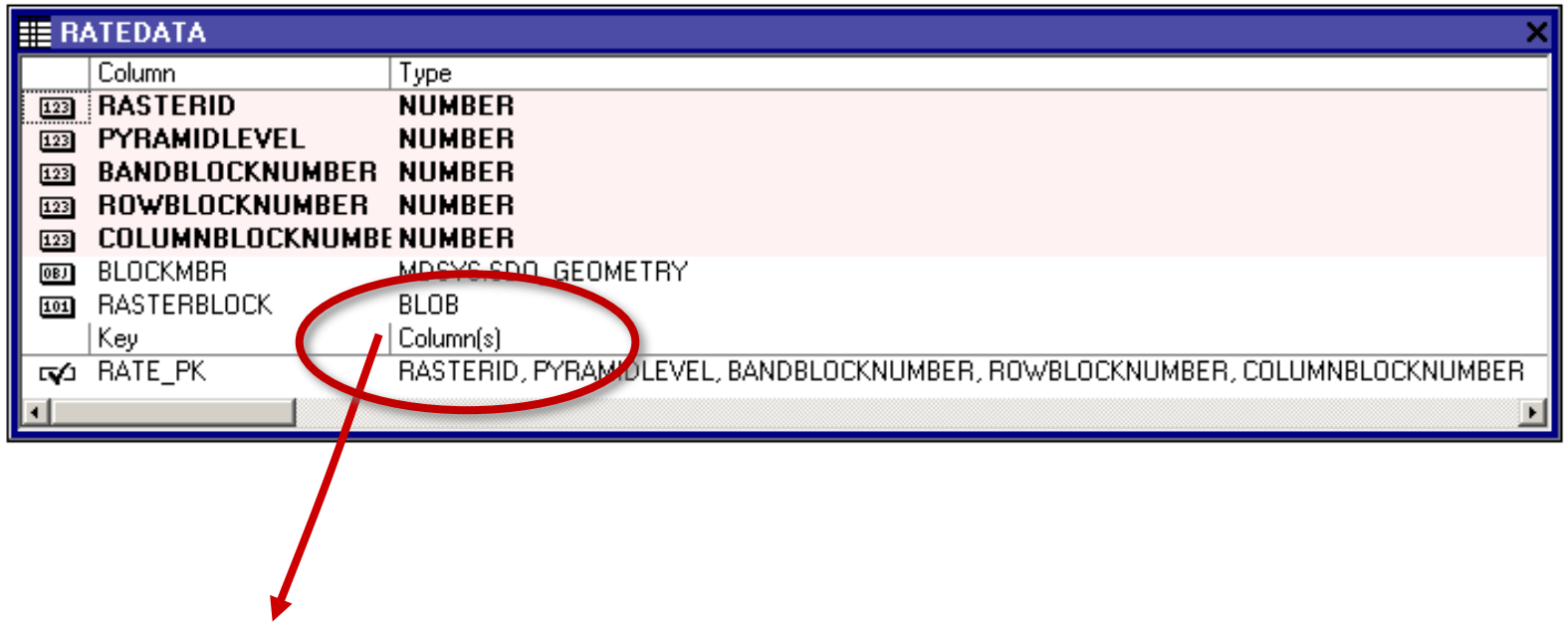
Identified location of Block

Block → all the pixels

Schema/User is MDSYS with SDO_xxx types !

BLOB field content

Block table (per Band, per Pyramid)



- **BLOB -** Binary Large OBject data type field stores the block's pixels and their values
- As always somewhat hidden - but we have implicit topology of the pixels in the BLOB field (!)
- Pixel values accessible through GUI methods or DB SQL methods

Overview

1. Principles of Raster Data Storage
2. Loading Raster Data
3. **Raster Data in Web Services**

Accessing raster data through Web Services

Large data collections in redundant File Systems

- Petabytes (PB) collections are not unusual
- Transactions and Backup with such quantities of Raster data may not be optimal in a database (BLOBs, non-atomic values)

Web Server (raster server) approach:

- E.g., ArcGIS Server, Rasdaman or UMN Mapserver;
- Web-based endpoints (REST) like Google, Open Street Map, leaflet lib;
- Rasterdata standards of Open Geospatial Consortium OGC: WMS, WCS;
- Useable from any web-enabled client, desktop or web app;
- Datasources – filesystems (common ones, cloud based like HDFS Hadoop Distributed File System); DBs for metadata (Oracle, PostgreSQL and others)
- Functionality extended beyond data services e.g., handling time series, aggregating, colour transformation.

Accessing raster data through Web Services

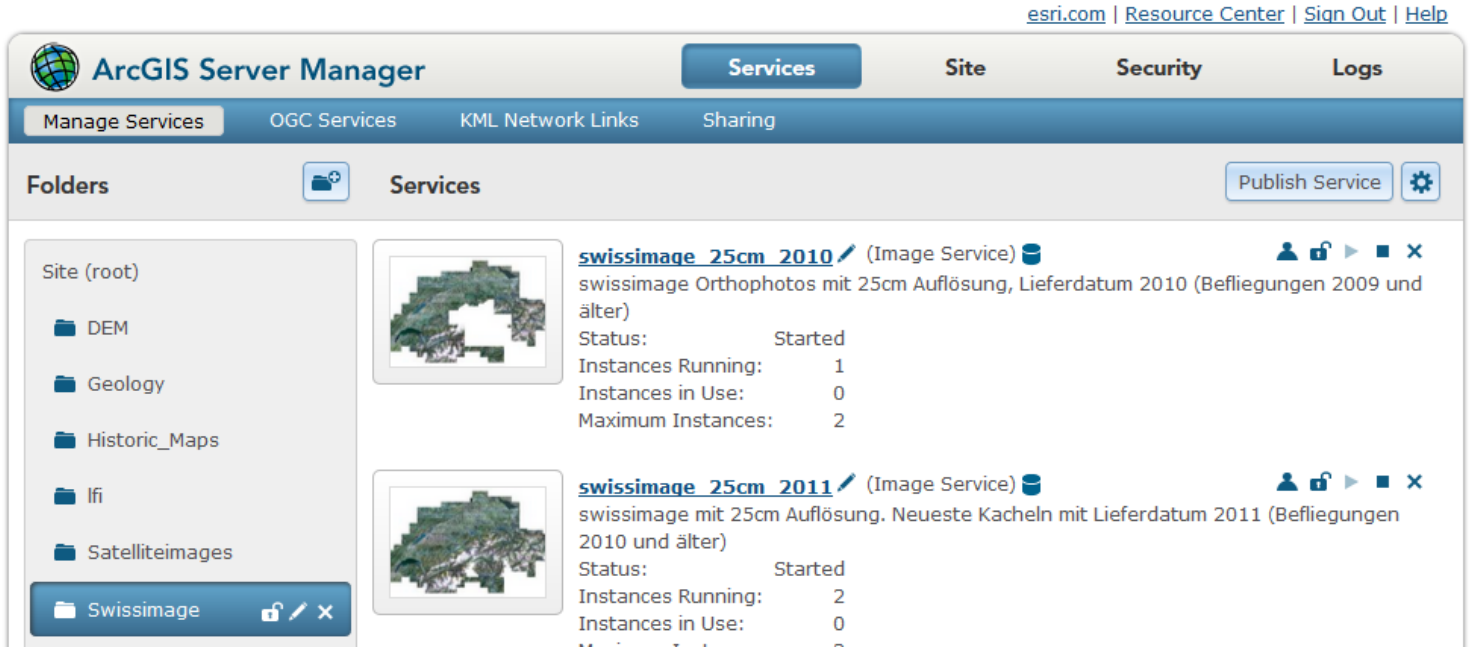
Management of Geo-Rasters in a File System

- Files are georeferenced (*.geotiff → tiff + header with georeferencing info);
- Redundant file systems are backed up;
- Support for Mosaic datasets through georeferencing files
- File lists are managed in a DB
- Scripts check for corruptions on regularly base

Advantages:

- Distributed (clustered) system possible;
- Supports clustered Web servers;
- Access through HTTP(S) protocol;
- Web based administration possible;
- Visualization and analysis simplified through wider range of tools.

Accessing raster data through Web Services



Alternatives for serving raster (and vector) data

- MapServer
- GeoServer
- QGIS Server

https://where2b-conference.com/fileadmin/where2b/resources/documets/archive/2018/Where2B-2018-Serververgleich-thildebrand.pdf

Raster data management @WSL (Esri products so far)

Raster dataset:

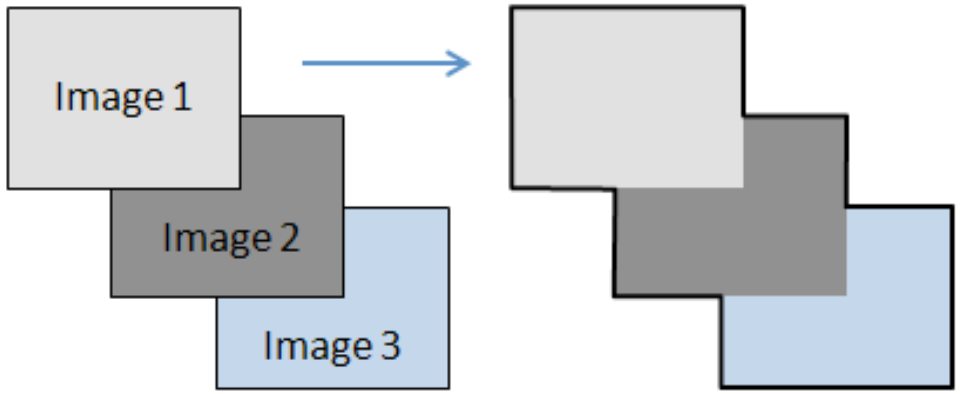
- Multiple pictures are stitched together during loading into the database;
- Edges must fit – or some method must be used to match overlapping pixels;
- Parameters of the rasters must be identical: Colormap, pixel size, pyramids, band numbers, spatial reference system;

Mosaic dataset (+ raster catalog):

- A collection of pictures or independent raster datasets;
- May overlap;
- May store multiple attributes;
- Each dataset in the catalog

  may have different ref. system

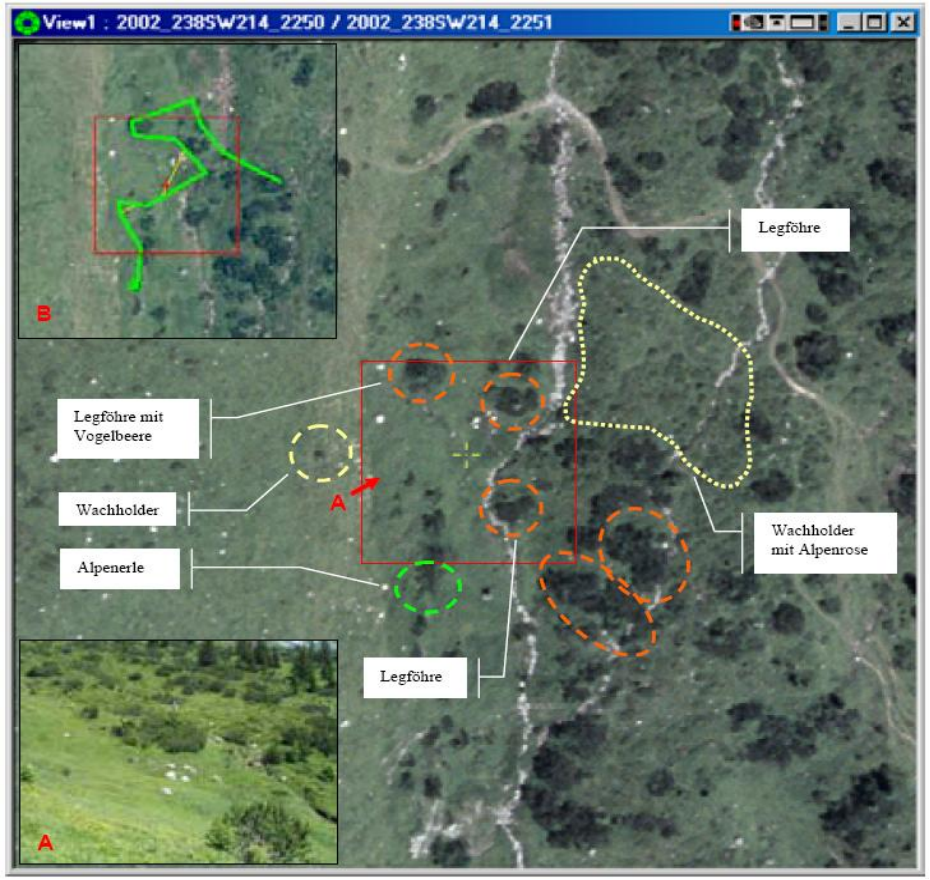
  and other parameters.



Raster data @WSL - almost no DB anymore

Managing large data quantities...

- Swissimage imagery
- State maps for different time periods and generalisations
- Special areal imagery for Forest inventory
- Clients: Erdas, ArcGISPro, ArcSDE Api, 3-D aerial imagery interpretation
- Multiuser and Multi-project-Systems
- Reached the Petabytes



Summary

- Raster storage in the DB;
  - Parameters (pixel values, Colour maps, Blocks, Bands, Pyramids);
  - Principles of Storage
- Approaches: Raster dataset vs. Raster Catalog
- Discussed examples of access through Web services;