Section 6 Technical issues: databases and pronunciation

Chapter 13 **Technical issues: database management**

Pier-Giorgio Zaccheddu

13.1 Introduction

After geographical names information has been gathered in the field, this information needs to be stored and prepared for dissemination. Several methods exist for storing the data, from the paper maps being an old-fashioned but proved medium, to the advanced digital techniques using databases.

For storage of names information in a digital format, one has several options, on the one hand text files or spreadsheets and on the other hand databases.

Some decisions have to be made on the used soft- and hardware side before starting the storage of the geographical names data in databases. It has to be considered that for proprietary database software solutions high license costs might need paying. In contrast, open source software products are free, but some maintenance and updating skills are required.

This section touches on the implications of the geographical names database management used for the output options, e.g. printing a gazetteer from the database or importing the database content into other systems, like Geographic Information Systems (GIS). The publication of geographical names through the internet is not part of the database management and will therefore not be explained in this chapter, but will be tackled in the following section 8.

13.2 Preliminary discussion on structure and content of toponymic databases and gazetteers

What does one have to consider (or to know) before one starts the creation of a database? The following information is a selection of possible issues to deliberate. This information is not intended to be exhaustive. It is envisaged to raise awareness for technical and organizational implications that may result from the decisions made.

For storage of names information in a digital format, one has several options:

- text files (such as Microsoft Word) or spreadsheets (such as Microsoft Excel)
- databases

Text files are easy to handle, but the methods have very limited capabilities in digital processing. Databases might be more complicated for non-skilled people, but the data can be connected with information stored in other databases (such as Geographic Information System repositories), and therefore it can be processed in many ways. That's one of the major reasons to store data in databases.

The advantage of using a database management system (DBMS) is that it can be used to impose a logical, structured organization on the data. A DBMS delivers economy of scale for processing large amounts of data because it is optimized for such operations.

Apart from the technical software support which will be tackled in the following section III, the layout and the structure of the database tables and – which is much more important – the purpose of the database, has to be considered. Examples of purposes are:

- Names database to be used in a Geographic Information System (GIS) for map production
- Names database to be used for the publication of geographical names as gazetteer through the Web
- Names database to be used for Web applications support

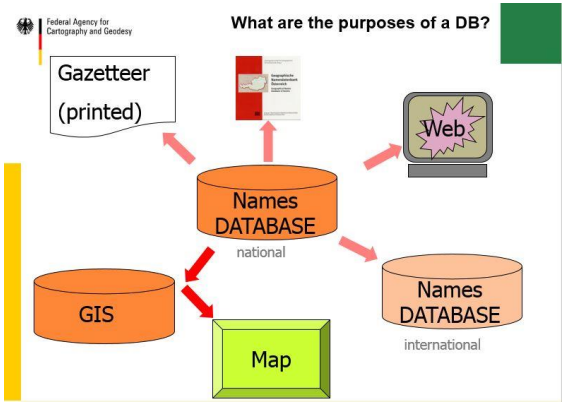

Figure 13-1 Purposes for geographical names databases

The database model design depends on the envisaged complexity of the geographical names content to be maintained, i.e. what object/features are to be captured, which attributes are associated to the object/feature and to the name? Therefore, the objects/features have to be selected and defined. The following object/feature types might be included: settlements, transport, vegetation, water bodies, relief areas, etc. This is related to the decision about the respective attributes associated with the objects/features. Examples are:

administrative division (statistical key-number), stream system (hydrological area code number), population size, status of the geographical name, language of the geographical name, height (height in metres above sea level), etc.

Within UNGEGN's 'Technical reference manual for the standardization of geographical names' a toponymic data transfer standard and format is mentioned in its second section, providing tables of Roman characters used in different languages, and a draft toponymic data exchange standard. The manual was published in 2006 [1]. The draft toponymic data exchange standard has been updated and adapted within several toponymic training courses from 2006 until today. A summary of the discussions can be displayed as follows:

Field Name	Data Type
RecordID	Index
UID	Number (long integer)
Name	Text (50 char.)
Latitude	Number (double)
Longitude	Number (double)
FeatureCode	Text (50 char.)
AdminUnit	Text (50 char.)
Language	Text (50 char.)
Description	Text (255 char.)
VariantName	Text (50 char.)
MapSheet	Number (long integer)
Source	Text (255 char.)
Status	Text (50 char.)
Pronunciation	OLE-Object
Location information	OLE-Object

Figure 13-2 UNGEGN Data model content (status: 2016)

The key for each database model is the inclusion of unique identifiers (UID) to uniquely identify each named feature.

Associated with the envisaged content of the geographical names database a feature classification has to be determined. Generally, a database should allow for different feature classifications to be used. Usually, a national feature classification is available and can be used for describing and classifying the objects/features in the database. A national feature classification recognizing the situation in the respective country and with feature code names in one (all) official language(s) as well as translations into other language(s) (e.g. English) would be helpful.

Sometimes regional views on and publication of geographical names data are needed, e.g. for regional projects or infrastructures. One good example is the European project "EuroGeoNames (EGN)" (funded by the European Commission from 2006 – 2009). Within EGN the "EGN feature classification" was developed because all other classifications available were deemed to be limited. However, the EGN classification was a compromise as well (the "highest common denominator") and it was satisfactory for the purposes for which it had been required, essentially, query filtering of the web service publishing the data of the different European databases. The EGN classification consisted of 8 classes and 27 sub-classes:

Code		Feature Type	Short Definition	Feature Type Examples
1		**COUNTRIES, ADMINISTRATIVE UNITS AND OTHER AREAS**	Countries, territorial units of a country for administrative purposes and other man-made areas	
	1.1	**Country**	Country of Europe	
	1.2	**Administrative units**	Territorial units of every country for statistics and administrative purposes *Including:*	
			Nomenclature of Territorial Units for Statistics in EU (NUTS 1, 2 and 3)	länder (Germany) autonomous region (Spain) province
			Local Administrative Units (LAU 1 and 2)	municipality
			Other administrative units	
	1.3	**Other non-administrative units**	Other type of man-made areas like economic, cultural, linguistic or tourist areas	
2		**POPULATED PLACES**	Buildings for housing of any category like cities, towns, villages, etc.	
	2.1	**Administrative capitals**	Populated places with capital status *Including:*	
			Administrative capitals of NUTS 1, 2 and 3	capital of country, autonomous region (Spain), province
			Administrative capitals of LAU 1 and 2	capital of municipality
			Other administrative capitals	

Figure 13-3 EGN Feature Classification - extract

Another important item to be touched on in relation to the storage of geographical names data in a database is the definition of a metadata profile/scheme for explaining the content of the database to users, customers, etc. The metadata profile should be created according to common international standards. Many different metadata schemes are being developed as standards across disciplines, such as library science, education, archiving, e-commerce, and arts. Concerning geographical names data as part of geospatial data the ISO 19115:2003 Geographic information -- Metadata standard is one example of how to describe geographical information and associated services, including contents,

spatial-temporal purchases, data quality, access and rights to use. It is maintained by the ISO/TC 211 committee.

13.3 Database design and management

A database management system (DBMS) is a system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data. A DBMS makes it possible for end users to create, read, update and delete data in a database [2]. The DBMS essentially serves as an interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible. The DBMS manages three important things: the data, the database engine/tool that allows data to be accessed, stored and modified -- and the database schema, which defines the database's logical structure. These three foundational elements help to provide concurrency, security, data integrity and uniform administration procedures.

The DBMS is perhaps most useful for providing a centralized view of data that can be accessed by multiple users, from multiple locations, in a controlled manner. A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing many views of a single database schema.

The DBMS can offer both logical and physical data-independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data (storage and hardware). As long as programs use the so-called application programming interface (API) for the database that is provided by the DBMS, developers won't have to modify programs just because changes have been made to the database. For relational DBMSs (RDBMSs), this API is SQL, a standard programming language for defining, protecting and accessing data in a RDBMS [2].

Using a DBMS to store and manage data comes with advantages. One of the biggest advantages of using a DBMS is that it lets end users and application programmers access and use the same data while managing data integrity. The latter is the maintenance of, and the assurance of the accuracy and consistency of data over its entire life-cycle. Data is better protected and maintained when it can be shared using a DBMS instead of creating new iterations of the same data stored in new files for every new application. The DBMS provides a central store of data that can be accessed by multiple users in a controlled manner.

If steps for designing a (geographical names) database were defined, the following ones should be included:

1) Determine the purpose of your database.
2) Determine the tables you need in the database.
3) Determine the fields you need in the tables.
4) Identify fields with unique values.
5) Determine the relationships between tables.
6) Refine your design.
7) Add data and create other database objects.

After the purpose of the geographical names database has been determined (step 1), the structure of the database becomes crucial. This means that the database schema or model, the tables and fields have to be selected and their relationships to be set (steps 2 – 5).

A geographical names database schema or model should allow flexibility. Anyway, the database schema should explain whether the geographical names will be used as an attribute of a spatial object/feature ("geospatial-based"), or whether the geographical names entries are the main database entries ("attribute-based"). In the geospatial community the first option is favoured, whereas the geographical names and language community very often model the databases while focusing on the names entries.

Concerning the latter option, the core table will hold one row for each geographical name. The definition of columns (fields) may vary considerably from country to country, but there are general rules that apply to most databases of geographical names. Typical fields associated with a geographical name are feature type, coordinates, variant names, textual description, source of the name information and status of the name. The following simple setup for rows and columns in the database explains the general rules:

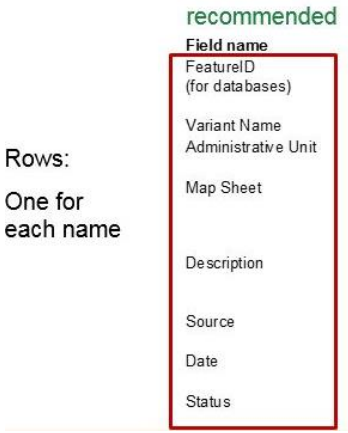

Figure 13-4 General rules for a simple database - rows

Columns:
attribute information to the name

Data type	Description
Integer	A unique identifier assigned to the name. This ID will be used to link the name with other database tables.
Text	Other names assigned to the feature, if any
Text	Name or Code of the Administrative Unit where the name is situated in.
Text	Reference to a map sheet in a topographic map series. The data type may be Integer if the sheet name contains only numbers no letters.
Text	Comments, e.g. on the history of the name, and verbal statements on the extension of the feature.
Text	Source of the name. e.g. captured in the field by interview
Date	Date of the entry to the Database. Other option: date of approval by the Board.
Text	Comment, e.g. name is approved or not approved by the Board.

Figure 13-5 - General rules for a simple database - columns

The possibility of linking more names with the same named place gives the opportunity to integrate minority languages and exonyms, which are an important contribution to multilingualism.

Within the European initiative “Infrastructure for Spatial Information in Europe – INSPIRE” the INSPIRE data specification on geographical names was prepared following the participative principle of a consensus building process, also involving UNGEGN experts. In INSPIRE the concept that the same place can be referred to by several names was reflected. In order to reflect this approach the central element of the INSPIRE geographical names data model is the spatial object “named place” that can carry one or more names. The specifications of geographical names can be used for modelling names in any other INSPIRE theme. Each named place has a unique INSPIRE identifier. It is further characterised by the eventual name(s), geometrical representation and if available, type, local type, indicative scale of usage, and the possibly related spatial objects. The latter helps to preserve consistency between data at different levels of detail. In addition, life-cycle information – i.e. when the named place has been inserted / changed, or eventually superseded / retired in the spatial data set – should be given if available [3].

Within any database design and management it shall be considered as well that geographical names data are used at all levels of resolution. The spatial resolution of a geographical names data set is typically described by the scale of the map where it has been captured from, or for which it has been captured.

The core of the INSPIRE geographical names application schema is described in figure 6 showing its non-voidable elements.

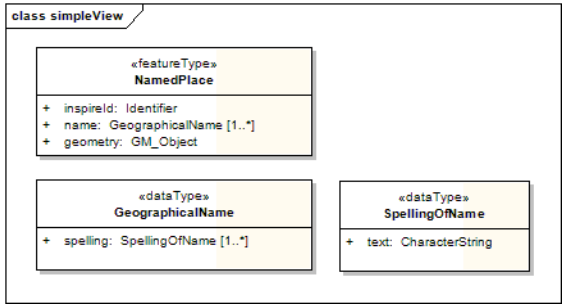

Figure 13-6 - core of the INSPIRE geographical names application schema

The only feature type of the schema is the feature type NamedPlace, representing any real world entity referred to by one or several proper nouns. Each NamedPlace is associated with one or several geographical names, i.e. proper nouns applied to the spatial object, modelled with the data type GeographicalName. The different geographical names of one given spatial object may be for example the names in different languages or in different forms (e.g. complete and short forms of country and administrative unit names). Each GeographicalName may have one or several spellings, i.e. proper ways of writing it, in one or several scripts like the Latin/Roman, Greek and Cyrillic scripts, modelled with the data type SpellingOfName. The following example explains how geographical names are modelled:

- The city of Athens may be modelled in the schema as one NamedPlace.
- The endonym “Athína” (Greek language) and exonym “Athens” (English language) are two different GeographicalName of this unique NamedPlace.
- “Αθηνα" (Greek script) and its standard romanisation "Athína" (Latin script) are two different SpellingOfName of the same GeographicalName “Athína”.

The following information shows one example for a characteristic which might be encountered when designing and managing the database. Different computer systems may cause and support different characteristics. For example, the operating system (OS) of the computer may cause displaying errors when geographical names data are exchanged and the settings are not known and not considered properly. An OS is a software program that manages the hardware and software resources of a computer. The OS performs basic tasks, such as controlling and allocating memory, prioritizing the processing of instructions, controlling

input and output devices, facilitating networking, and managing files. The dominant desktop operating system is Microsoft Windows with a market share of around 85%. OS X by Apple Inc. is in second place (9%), and Linux is in third position (1.5%) [4].

The OS uses code pages, i.e. a table of values that describe the character set used for encoding a particular set of glyphs – i.e. graphic symbols that provide the appearance or form for a character –, usually combined with a number of control characters. The Unicode Standard is an effort to include all characters from previous code pages into a single character enumeration that can be used with a number of encoding schemes [5]. If the produced geographical names database is set with the Unicode character set and the receiving operating system supports the Unicode character set, no displaying errors should appear.

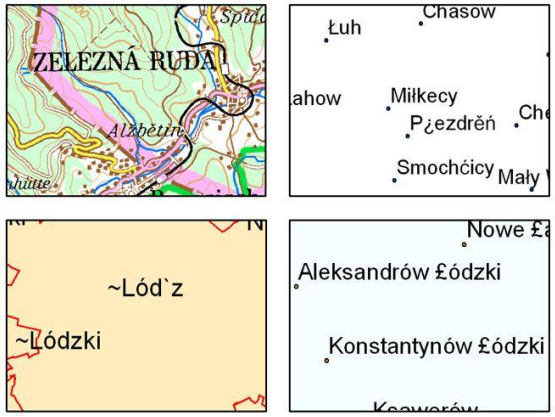

Figure 13-7 - Possible displaying errors of character sets in GIS

Today such displaying errors do appear rarely as the common data ETL (Extract Transform and Load) routines used by GIS or other applications have integrated the Unicode standard and are updated regularly.

13.4 Database open source options vs commercial options: criteria for selection

There is a huge choice of database management systems (DBMS), which includes commercial and open source database products or tools.

This section provides possible criteria for the selection of database software tools. The criteria are the result of an overall comparison of open source and commercial options. It is exemplary only and does not rely on a comprehensive scientific research. Therefore, the following criteria can be reflected and used for the main questions to be answered before extracting geographical names from the database and displaying them in a GIS, on a satellite image or in other applications.

Examples for commercial options are, amongst others, Microsoft Access and Oracle and for open source options MySQL. Microsoft Access is a pseudo-relational database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software development tools. Microsoft Access stores data in its own format based on the Access Jet Database Engine. It can also import or link directly to data stored in other applications and databases [6]. MySQL "The world's most popular open source database is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements.

MySQL is a popular choice for using databases in web applications. PostgreSQL, often simply Postgres, is an open source object-relational database management system (ORDBMS). PostgreSQL is not controlled by any single company — a global community of developers and companies develops the system. PostGIS – as an extension for PostgreSQL – adds support for geographic objects to the PostgreSQL object-relational database. In effect, PostGIS "spatially enables" the PostgreSQL server, allowing it to be used as a backend spatial database for geographic information systems (GIS) [7].

Databases are meant to be interoperable, but they all have their own operational procedures and processes for storing data etc. It should be a last ditch choice to change the database, although there are sometimes good financial reasons for changing, such as licensing issues, acquisition or mergers.

A possible list of criteria for selection for both – commercial and open source tools – may be the following:

Requirement	Commercial	Open Source
Data abstraction and independence using the same data	Yes	Yes
The ability to swiftly recover from crashes and errors	Yes	Yes
Data security	Yes	Yes
A locking mechanism for concurrent access	Yes	Yes
An efficient handler to balance the needs of multiple applications	Yes	Yes

Robust data integrity capabilities	Yes	Yes
Logging and auditing of activity	Yes	Yes
Simple access using a standard application programming interface (API)	Yes	Yes
License costs	Yes, eventually very high	No
Technical support	Yes, but additional effort and usually not included	Yes, support by IT companies can be contracted

Figure 13-8 - Possible list of selection criteria

In a nutshell, the most relevant technical and organizational requirements seem to be fulfilled by commercial and open source database tools. Many organizations are turning to open source databases to reduce database management (licensing) costs and avoid supplier lock-in. The maturity of open source databases is at its highest level ever, with more choices, better support and comprehensive ecosystems. Open source DBMS products continue to improve in terms of functionality and scalability, and DBMS tool suppliers are beginning to provide support for these offers. Software companies do provide support to DBMS implementations.

Thus, the big advantage of open source software tools is the licensing cost savings, which might be significantly higher using commercial software tools. It is up to the organization to decide whether the licensing cost savings using open source software tools can be invested in capacity building of the employees or for subcontracts with private companies for providing substantial IT support. However, this support might be needed for commercial software tools as well.

If the geographical names database – be it proprietary or open source – will be imported into a Geographic Information Systems (GIS), the same decisions as for the database have to be made. The main purpose for doing this import is the map production, i.e. to link and overlay the geographical names data with other spatial (topographic) data.

The following information is the result of the overall comparison of most popular open source and commercial options. It is exemplary only and does not rely on a comprehensive scientific research. The most popular commercial GIS software product is Esri® ArcGIS®. It facilitates collaboration and lets one author data, maps, globes, and models on the desktop and serve them for use on a desktop, in a browser, or in the field, depending on the needs of one's organization [8].

ArcGIS support and educational services consist of technical maintenance programs, software releases and updates, technical support, online support services, publications, training, and consulting services. As an open source alternative Quantum GIS (QGIS) can be mentioned. QGIS provides data viewing, editing, and analysis capabilities. QGIS is a user friendly open source GIS licensed under the GNU General Public License. QGIS runs on different operating systems like Linux, Unix, Mac OSX, and Windows and supports numerous vector, raster, and database formats and functionalities [9].

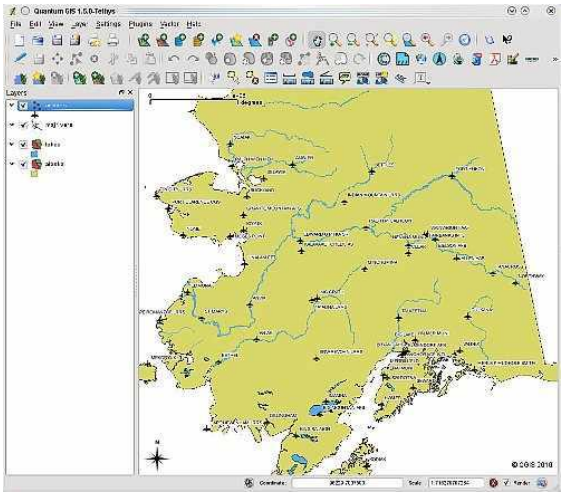

Figure 13-9 - Screenshot from Quantum GIS (QGIS)

Both, proprietary and open source GIS software products do fulfil the requirements for a high-quality spatial data analysis and map production. Again, the big advantage of open source software tools are the licensing cost savings.

13.5 References

[1] UNGEGN Technical reference manual for the standardization of geographical names, http://unstats.un.org/unsd/geoinfo/UNGEGN/publications.html, last accessed 09/2016

[2] What is a database management system (DBMS)? - Definition from WhatIs.com, www.whatis.com, last accessed 09/2016

[3] INSPIRE Data Specification for Geographical Names, http://inspire.ec.europa.eu/index.cfm/pageid/2, last accessed 09/2016

[4] Wikipedia (2016) Operating system, https://en.wikipedia.org/wiki/Operating_system, last accessed 09/2016

[5] Unicode Consortium, codepages, http://www.unicode.org/ , last accessed 09/2016

[6] Microsoft Access, Website: http://office.microsoft.com/en-us/access/ , last accessed 09/2016

[7] MySQL, Website: http://www.mysql.com/ and http://www.postgresql.org/ and http://postgis.refractions.net/ , last accessed 09/2016

[8] Esri® ArcGIS, Website: http://www.esri.com/software/arcgis/index.html, last accessed 09/2016

[9] Quantum GIS, Website http://www.qgis.org/en.html, last accessed 09/2016