Design considerations for a generic database for dynamic maritime parameters

Anca Atodiresei[1], Marian Cață[2], and Andrei Băutu[3]

[1] "Mircea cel Bătrân" Naval Academy, Romania, anca.atodiresei@anmb.ro
[2] "Mircea cel Bătrân" Naval Academy, Romania, marian.cata@anmb.ro
[3] "Mircea cel Bătrân" Naval Academy, Romania, andrei.bautu@anmb.ro

Abstract. The advancement of technology has made the maritime industry more complex, leading to an increase in the need for effective management systems. In this research paper, we present an analysis of the design considerations for a generic database for dynamic maritime parameters. The objective is to develop a database system that can handle a wide range of maritime parameters and be used in different applications. The paper covers aspects such as data modelling, data storage, and performance. While we focus on meteorological parameters, the proposed database system is designed to be flexible and efficient, allowing for easy integration with other systems in the maritime industry.

1. Introduction

Databases are a fundamental part of any modern computer system. Database management systems (DBMS) are specialized software that play a crucial role, because they enable the creation and manipulation of databases. DBMS are typically categorized based on their data models, which determine the structure in which data is stored. Although relational databases, which emerged in the 1970s, have been the leading solution for many years, newer models have also emerged, such as object-oriented (OO), NoSQL (Not Only SQL), and NewSQL databases. OO databases have an advantage in that they are compatible with OO programming languages, thereby avoiding mismatches between relational databases and programming languages. NoSQL databases prioritize flexibility and horizontal scalability over the rigidity of relational databases. Lastly, NewSQL databases merge the benefits of both relational and NoSQL databases.

To state the database design problem, one must design the logical and physical structure of a database within a particular database management system or paradigm (relational, OO, NoSQL, etc). This design must ensure that the database contains all necessary user information and facilitates efficient behavior across the entire computer system for all users, as well as during application processes and user interaction [1].

2. Database models

2.1. The relational database model

Beginning in the 1970s, the relational database model, first proposed by E.F. Codd in [2], has been increasingly utilized in database modeling. Codd's definition of the "relational model" included the use of tables composed of rows (known as records) and columns (known as attributes) to store data. This model employs connections between records in tables to represent data relationships.

A relational database comprises data elements arranged in tables with defined structures, providing diverse methods for accessing or reconstructing data. Tables, referred to as relations, consist of columns representing various data categories, and each row stores a distinct data instance for the relevant category. The database creation process involves validating the data's potential values and applying constraints. The tables' interdependence creates a "relationship," rendering the model flexible for multiple perspectives of the same database. Therefore, this approach entails minimal assumptions about the database's data retrieval methods.

Relational databases offer several advantages such as having the majority of information stored within the database itself instead of the application, making it self-documenting. Additionally, it allows for effortless manipulation of data by adding, updating, or deleting it. Furthermore, it provides benefits such as data summarization, retrieval, and reporting. The structure of the database is predictable as it is organized into interconnected tables in a tabular form. Finally, any required alterations to the database schema are straightforward to implement.

Almost all database management systems have implemented the relational model, using SQL as the standardized tool for querying and manipulating these databases. Researchers have also created models related to the relational model or that enable relational models to be derived from higher analytic models, using mathematical theories similar to those used by Codd. An example of such a model is the elementary mathematical model of data (MMED), which was developed by Romanian researcher Christian Mancaș [3], [4], [5], [6]. This model can convert MMED schemes into MRD schemes and E-R diagrams while providing additional tools for validation and control.

Relational databases, despite their widespread use, have various disadvantages. These include a lack of high scalability, as beyond a certain point, the database needs to be distributed which can cause significant synchronization issues. The storage of data in tables can also lead to increased complexity if the data cannot be easily encapsulated. Additionally, many of the features offered by relational databases are unused, which raises costs and complexity without providing benefits. SQL, the language used by relational databases, is designed for structured data and can become complicated when working with unstructured data. Finally, when working with large amounts of data, the database needs to be partitioned across multiple servers, which can be challenging when joining tables on distributed servers.

2.2. Other database models

Due to the limitations of the relational database model, over the past decades, new types of database models have emerged, particularly the NoSQL models of databases. Within this family of databases, there are specific models such as document-based and graph-based databases, among others. Each of these models can be more suitable for specific types of applications than the relational model. Some of the types of NoSQL models are:

- Column-based databases store data in columns rather than rows, making them more efficient than row-oriented databases for updating or inserting new values for a column across multiple rows.
- Key-Value stores allow data to be stored without a fixed data model, where data consists of a key represented by a string and associated data that can be any programming language primitive or object.
- Document stores use a document as their basic storage structure, with each document represented by a unique string key and encoded in a standard format such as XML or JSON.
- Object-oriented databases store data as objects, allowing for OOP features like inheritance and reuse.
- Graph databases use graph data structures to represent data and allow for the extraction of meaningful patterns by studying interconnections between nodes, edges, and properties.
- Grid and cloud databases use grid and cloud computing to manage distributed databases and provide easy access to remote hardware and storage resources.
- XML databases store XML data as their fundamental storage format, and there are also hybrid XML databases that combine XML storage with other database models.

Choosing the appropriate database model for a specific application is a significant consideration. Presently, there appears to be a lack of systematic methods that consider the specific needs and features of the sought application when selecting a database model. In [7], a structured approach to selecting a database model is proposed by the authors. This approach takes into account several factors such as data-related requirements, functional requirements, and non-functional requirements. The method suggests the most suitable database models for the corresponding application based on these factors.

2.3. Selection of appropriate database model(s)

The task of database designers is to select the most appropriate model for a specific application, from the numerous database models currently available for use. This selection can be made through different strategies, including the agenda-based strategy, which involves choosing a model based on the latest trends and a desire to learn something new. Additionally, practitioners may opt for the knowledge-based strategy, which relies on personal or organizational experience with previously used databases. Finally, the exploration-based strategy entails selecting a database model based on problem analysis, such as data analysis and objective analysis, to find the best DBMSs that suit the problem at hand.

The third option may seem appealing, but it is unfortunately not widely utilized and lacks structure or establishment. In [7], the authors present a technique for choosing a database model that prioritizes user needs, encompassing data-related, functional, and non-functional requirements. The approach for selecting database models accounts for various types of user requirements and is designed to be implemented at the outset of database development to ensure that the model aligns with user needs. The method involves the following steps:

1. Gather and specify data requirements and express them using a conceptual data model (eg using UML class diagrams).
2. Gather and specify the functional requirements that are related to database operations, that is, data retrieval and update operations (generally called queries).
3. Gather and specify the non-functional requirements that are related to the data requirements and their queries.
4. Based on the above, consider dividing the conceptual data model into fragments, each of which has different characteristics (different performance requirements and different consistency requirements). The result of this step can be many more database models.
5. Select the most appropriate database model for each shard. The choice will be based on a predefined general purpose profile of each database model. A predefined profile consists of a set of non-functional properties associated with each database model.

3. Designing a generic database for dynamic maritime parameters

Our research was funded by the Platmarisc research project (project cod ID / Cod MySMIS: 120201). Within the project, the first research activity aims to identify the components of the systems operating in coastal areas and maritime ports of the Black Sea, with roles in the generation of disaster risks. The project teams will also identify the main dynamic parameters that are required to simulate such systems and disaster scenarios within integrated maritime simulators.

Within the next activities, the values of these parameters will be periodically measured (through various sensors), transmitted (through dedicated communication channels), and stored (in the database created by database experts). This data will be used in the implementation of many activities of the project. Among the parameters identified, which are of interest in the modeling and simulation processes, the research team identified the hydro-meteorological parameters from Table 1.

Table 1. Paragraph styles.

Parameter	Unit of measure	Description
dd	°	Wind direction - average 10 min
ff	m/s	Wind speed - average 10 min
Hw	m	Wave height - visual
iRal	-	Precipitation inclusion indicator warning
ITU	-	Temperature index - ITU
Iw	-	Wind sensor indicator
L	-	Precipitation type
La	°	Geographical latitude
Lo	°	Geographical longitude
N	-	Nebulosity
Napa	m	Water level
P	mB	Reduced pressure at sea level
ppp	mB	Pressure trend value
Pw	s	Wave period - visual
R1	mm	Precipitation 1h
RD10min	KJ/m2	Diffuse radiation - 10 min
Rff10m	m/s	Maximum gust for 10 minutes
RG10min	KJ/m2	Global radiation - 10 min
Rint1h	m/h	Rainfall intensity per 1h
Sal	ppm	Water salinity

Parameter	Unit of measure	Description
SSS10m	mins	Sunshine for 10 minutes
Ta	°C	Air temperature
Td	°C	Dew point temperature
Temp	°C	Average water temperature
UR	%	Air relative humidity
VV	m	Horizontal visibility
VVh	m	Vertical visibility
WHeight	cm	The significant height of the wave
WPeriod	0.1s	Typical wave period
ww	-	Current weather conditions

In order to obtain correct and consistent results for the statistical analysis, modeling and simulation tasks that will take place within the project, it is mandatory that the values recorded in-situ for these parameters are stored and processed appropriately. In this sense, the project team responsible for the implementation of the database analyzed several database models and technical solutions in order to design and implement a dynamic high-performance database for these parameters.

Due to the dynamic characteristics of the specifications of identified parameters, the database must have a flexible structure that can be easily adapted to changes in parameter specifications (their type, invalid/missing values, data resolution, etc.), as well as the emergence of new parameters that could later be added to the list of those of interest for project implementation.

At the same time, due to the nature of the measured and recorded processes, the database must allow the storage, processing and analysis of large amounts of data, with a high rate of transfer, both on the writing side (input) and on the reading side (output). The design of the database must take this aspect into account, so that the implementation of the database ensures a high degree of availability and high throughput.

Given the information provided by the experts in the project, the structure of the database proposed by the database experts is described by the entity-relationship diagram from Figure 1.

4. Conclusions

As part of the research carried out during within Platmarisc project, our study on database models was carried out in order to identify a database model suitable for the collection and processing of dynamic parameters of maritime systems, a study that led to the conceptual design of a database for hydro-meteorological parameters.

The research carried out also has an important contribution to the implementation of the Platmarisc project, by providing a tool for the collection and processing of maritime parameters, measured by the sensors of the integrated marine platform, data which will be essential for the modeling and simulation activities planned in the next stages of the project.

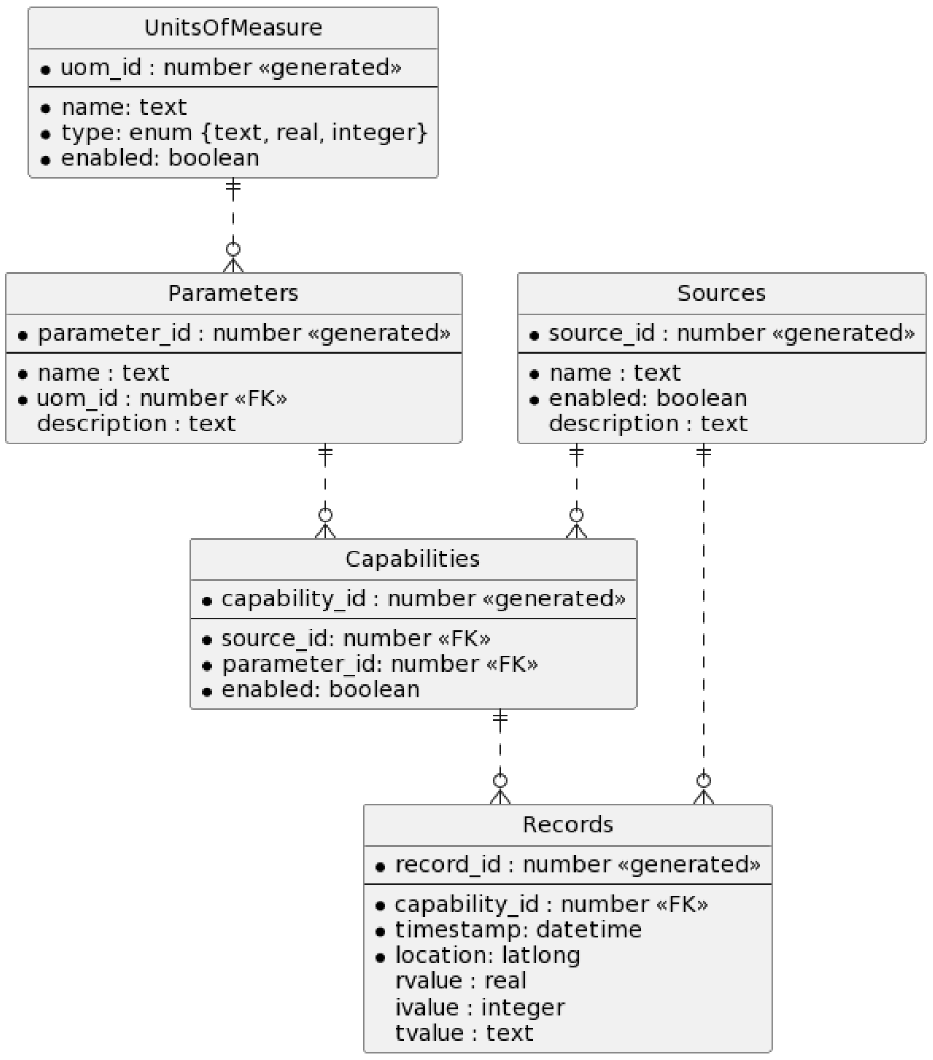

Figure 1. E-R diagram of the database for dynamic maritime parameters

Acknowledgements: This work was supported by grant no. 383/390059/04.10.2021, project cod ID / Cod MySMIS: 120201: Innovative integrated maritime platform for real-time intervention through simulated disaster risk management assistance in coastal and port areas – PLATMARISC.

References

[1] Thalheim, Bernhard. Entity-relationship modeling: foundations of database technology. Springer Science & Business Media, 2013.
[2] Codd, E. F. (1970). "A Relational Model of Data for Large Shared Data Banks". Communications of the ACM. 13 (6): 377–387. doi:10.1145/362384.362685.
[3] Mancas, Christian. "On enforcing relational constraints in MatBase." London Journal of Research in Computer Science and Technology (2017).
[4] Mancas, Christian. "MatBase E-RD Cycles Associated Non-Relational Constraints Discovery Assistance Algorithm." Intelligent Computing-Proceedings of the Computing Conference. Springer, Cham, 2019.
[5] Mancas, Christian. "MatBase constraint sets coherence and minimality enforcement algorithms." European Conference on Advances in Databases and Information Systems. Springer, Cham, 2018.

[6] Mancas, Christian. Conceptual Data Modeling and Database Design: A Fully Algorithmic Approach, Volume 1: The Shortest Advisable Path. CRC Press, 2016.
[7] Roy-Hubara, N., Shoval, P., & Sturm, A. (2019). A Method for Database Model Selection. Lecture Notes in Business Information Processing, 261–275. doi:10.1007/978-3-030-20618-5_18