IIAI Open Conference Publication Series
IIAI Letters on Institutional Research
Vol. 005, LIR333
DOI: https://doi.org/10.52731/lir.v005.333

Categorical Database for Ensuring Data Integrity in Institutional Research

Tsunenori Inakura [*], Shotaro Imai [†], Kunihiko Takamatsu [*], Sayaka Matsumoto [*], Masao Mori [*]

Abstract

Institutional research deals with large and different datasets from various departments, and this can make it hard to keep the data accurate. In this paper, we present categorical databases, which is a method based on category theory, that helps to maintain data integrity. By organizing the database as a category, we can see how the data elements are connected. This makes it easier to do meaningful and precise data analysis. The connections between data can be shown as simple sentences that still make sense, even when the data is updated. This way ensures that data remain consistent in both databases and data warehouses through natural transformations, which means that references to the data stay trustworthy. Categorical databases provide a solid way to manage complex data structures, and they make sure that data integrity is kept.

Keywords: Categorical database, category theory, database, institutional research.

1 Introduction

Institutional Research (IR) refers to the investigation and research that supports decision-making in universities by collecting and analyzing data on university activities, such as education, research, and finances. Since IR requires vast amounts of data, various data warehouses have been constructed. These data warehouses are based on databases utilized across different departments, and it is common to convert the data formats of database for inclusion to data warehouse due to the wide variety of data formats involved. At the same time, databases are often expanded and modified to meet the flexible needs of day-to-day operations. As a result, issues arise, such as bloated data tables and increasingly complex data relationships, transforming the databases into complex and massive black boxes. Consequently, data warehouses derived from these databases also become more complex, resulting in data inconsistencies that hinder data analysis and processing. Ensuring and maintaining data integrity is crucial for the construction and operation of databases and data warehouses, and there is a demand for simple methods to build such databases.

Moreover, to conduct IR analysis, it is essential to understand the precise meaning of the data. However, understanding the concrete meaning of data stored in data warehouses and databases has become increasingly difficult. A contributing factor is that data integrity is not always maintained within the database. When the relationships between data in a database become inappropriate, it becomes challenging to interpret the meaning of the data. If data integrity is en-

[*] Institutional Research Section, Office of Institute Strategy, Institute of Science Tokyo, Tokyo, Japan
[†] Center for Information Infrastructure, Institute of Science Tokyo, Tokyo, Japan

sured within the database and appropriately transferred to the data warehouse, then data integrity will also be maintained in the data warehouse. A better understanding of the database enables the identification of underlying issues, making it easier to grasp the meaning of data within the data warehouse.

Categorical databases, which apply category to databases, provide a method for guaranteeing data integrity. When databases are built following specific rules, they can form a category. Such databases satisfy some mathematical ideas from category theory, such as the data integrity. In this paper, we will give a brief explanation of category theory and present a simple way to build categorical databases. We also discuss how to change databases into data warehouses with maintaining the data integrity.

2 Category Theory

Category theory is a mathematical theory that expresses and studies the properties of diagrams composed solely of objects (●) and morphisms (→) from a high-level perspective (see Figure 1). In mathematical concepts, a collection that satisfies the following conditions is called a category:

- **(Objects)** There exists objects c.
- **(Morphisms)** There exists morphisms $f: c_1 \to c_2$ connecting objects. c_1 and c_2 are referred to as the domain and codomain, repspectively.
- **(Composite morphisms)** If there are two consecutive morphisms $f: c_1 \to c_2$ and $g: c_2 \to c_3$, then there exists a composite morphism $fg: c_1 \to c_3$.
- **(Identity morphism)** For every object c, there exists an identity morphism $1: c \to c$, and the properties of composition hold: $1f = f1 = f$.

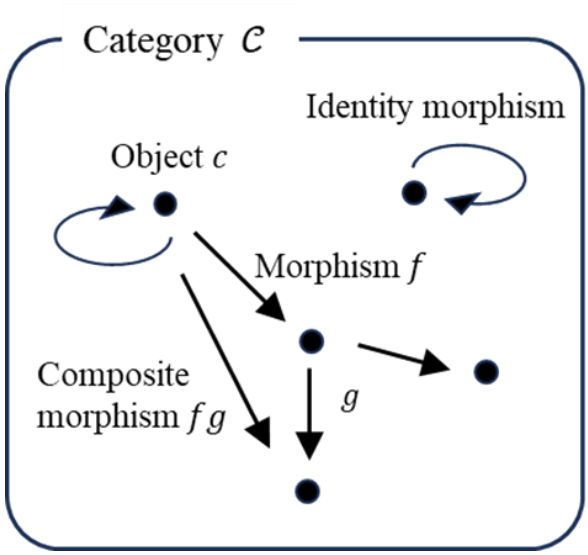

Figure 1: Schematic image of category.

The morphism is functional, namely, it requires that it is defined for all elements of the domain (totality) and that each element of the domain maps to a unique element in the codomain

(uniqueness). Any mathematical concept that satisfies the above conditions is a category. Although category theory is highly abstract, it reveals a wealth of theorems and properties.

It has been demonstrated by Spivak that a database can form a category if it satisfies the above conditions [1,2]. A database that forms a category is called a categorical database. For categorical databases, various theorems and properties from category theory can be applied to. Data integrity is one of those properties.

When a database forms a category, the correspondence between the database and the category is as follows:

- Database schema: Category
- Database tables: Objects in the category
- Foreign keys in the tables: Morphisms in the category
- Data update: Natural transformation in the category

The entire database schema is treated as a single category. The tables correspond to the objects of the category, and the foreign keys in the tables correspond to the morphisms. Totality and uniqueness of morphisms must be satisfied:

- **(Totality)** All data in the foreign key columns of the parent table (domain) must have valid corresponding data in the referenced child table (codomain).
- **(Uniqueness)** Each piece of data in a foreign key column must refer to only one target. It is acceptable for multiple pieces of data to be aggregated into a single reference.

These guarantee many-to-one or one-to-one relationships between tables. The reference target for a foreign key must be the primary key of the child table. Additionally, identity morphisms correspond to the primary keys of the same tables.

Any mathematical concept that satisfies the above conditions is called a category. A famous example of a category is the category of sets, denoted as "Sets," where the objects are sets and the morphisms are functions defined on those sets. Additionally, there is a category whose objects are categories themselves, known as the "category of categories." In this category, the morphisms between categories are called functors. Although highly abstract, category theory reveals numerous theorems and properties.

3 Categorical Database

In this section, we introduce a simple method for designing a categorical database, using a course management database as an example. In order to clarify, the final result is shown in Figure 2.

The course management database is designed according to the following steps.

Step 1. List the concepts needed to explain "course management."

First, list the concepts (entities) that seem necessary to explain the operations related to "course management." It is not necessary to list all the concepts at this step, and more can be added later. The concepts chosen depend on the creator, but here, eight concepts are listed: "Grade," "Score," "Instructor," "Course," "Student," "Department," "Subject," and "Country."

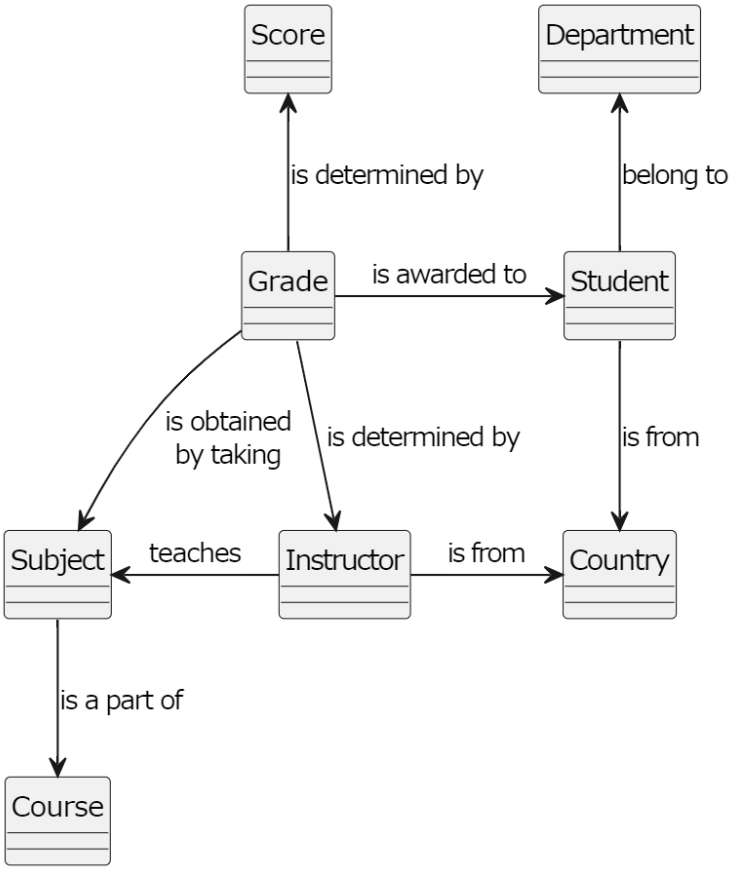

Figure 2: Ontology log of a mock course management database.

Step 2. Connect related concepts with lines.

The listed concepts are connected with lines where there is a perceived relationship between them.

Step 3. Express the relationships between the concepts in sentences.

The relationships between the connected concepts are written as sentences. The related concepts become the "subject" and "object," and their relationship becomes the "predicate." For example, the relationship between "Grade" and "Student" can be expressed in two sentences: One is "Grade is awarded to student", the other is "Student receives grade." Both sentences express the relationship between "Grade" and "Student," but having two relationships is redundant. By assigning a direction to the functional relationship, only one needs to be used. In the case of the relationship between "Grade" and "Student," the direction "Grade is awarded to student" is functional. Functional relationships satisfy totality and uniqueness. Applied to the relationship "Grade → Student," these properties mean:

- Totality: Every "Grade" has an associated "Student."
- Uniqueness: Each "Grade" belongs to only one "Student."

This ensures a one-to-one relationship. The reverse direction, "Student → Grade," is not functional, because:

- Not all students receive grades (e.g., due to taking a leave of absence), so totality is not satisfied.
- One student may receive grades for multiple courses, so uniqueness is not satisfied.

The determined relationship is represented as ⌈Grade is awarded to student.⌋

Step 4. Express all relationships as sentences.

All relationships connected are expressed as functional sentences. Nonfunctional relationships such as the relationship between "Instructor" and "Student" are excluded. The resulting set of functional relationships between concepts, expressed in sentences, is called the ontology log (olog) [3].

Step 5. Ensure that concatenating successive relationships results in meaningful sentences.

A connection through consecutive morphisms is referred to as a path. For example, take the successive relationships "Grade → Student → Department" from Figure 2. These relationships can be concatenated into the following sentence: ⌈Grade is awarded to Student⌋, and ⌈Student belongs to Department. ⌋ This can be further summarized into: ⌈Grade is an educational outcome of Department. ⌋ This sentence expresses the relationship between "Grade" and "Department," and whereas the concept of "Student" does not explicitly appear, it is implicitly included in the summarized sentence. If a single sentence representing the path retains the meaning and interpretation of the successive relationships expressed individually, then the path forms a composite morphism of consecutive morphisms. This ensures that complex data relationships remain coherent. Conversely, if the sentence lacks coherence, it suggests deficiencies in the concepts or relationships. The last step is concatenating any successive relationships can still result in a single meaningful sentence.

The ontology log created for the course management database is shown in Figure 2. Whereas the database instance itself has not been explicitly displayed, this ontology log forms a categorical database, where data integrity is ensured. The conditions for constructing a categorical database are verified by checking that each sentence makes sense, thereby maintaining mathematical rigor. This approach enables the design of a database with ensured data integrity, even without specialized knowledge in databases or mathematics.

In database design, by expressing foreign keys in sentences and creating an ontology log, one can intuitively determine whether the integrity of data references is maintained simply by reading the sentences. Moreover, by expressing paths in sentences, the flow of data references can be understood solely from the data schema. For example, consider two paths shown in Figure 2: "Grade → Instructor → Subject" and "Grade → Subject." Both paths have the same starting and ending points, and when expressed in sentences, they are as follows:

- ⌈Grade is obtained by taking Subject. ⌋

- [Grade is determined by Instructor,] and [Instructor teaches Subject.]

It is clear that these two paths refer to the same data when considering the instance data. However, two paths that share the same starting and ending points do not necessarily refer to the same data.

When we express two paths in Figure 2, "Grade → Instructor → Country" and "Grade → Student → Country," we have:

- [Grade is determined by Instructor,] and [Instructor is from Country.]
- [Grade is awarded to Student,] and [Student is from Country.]

It is evident that the former references the nationality data of the instructor, while the latter references the nationality data of the student, so the data being referenced is different. Therefore, even though the starting and ending points are the same, these two paths does not refer to the same data. In this way, by expressing paths in sentences, it becomes easy to determine whether they refer to the same data based solely on the database schema.

4 From Databases to Data Warehouses

Concatenating any successive relationships into meaningful sentences guarantees data integrity. This means that even when foreign keys are followed in a database, data can be referenced accurately. As is mentioned, for example, in Figure 2, the successive relationships "Grade → Student → Department" are summarized into [Grade is an educational outcome of Department.] This sentence implicitly includes the concept of "Student" and indicates that analyzing grades by department is meaningful from an IR perspective. Similarly, concatenating "Grade → Student → Country" suggests that analyzing the relationship between students' home countries and their grades is also meaningful. The sentence that expresses the meaning of a path reveals the relationships between the data, making it easy to understand what insights can be gained from analyzing that data.

Databases are integrated into data warehouses. Since databases used across various departments often have diverse formats, they are reformatted when brought into a data warehouse. If the meanings and interpretations of the sentences describing the data reference paths in the database align with those describing the paths in the data warehouse, the database can be integrated into the data warehouse with preserving data integrity. This is illustrated in Figure 3. In the database, data reference relationships are represented for paths such as “Grade → Subject → Course,” “Grade → Instructor,” “Grade → Student → Department,” and “Grade → Student → Country,” which will be brought into the data warehouse. Similarly, the data warehouse includes data reference paths such as “Grade → Course,” “Grade → Instructor”, “Grade → Department,” and “Grade → Country.” If the corresponding sentences describing these references convey the same meaning and interpretation, even if the wording differs, the database can be integrated into the data warehouse without compromising data integrity. If, however, the sentences describing data references have different meanings, the data references in the database should be reorganized to match the meaning of the sentences in the data warehouse. In this way, the database can be integrated into the data warehouse without disrupting data integrity.

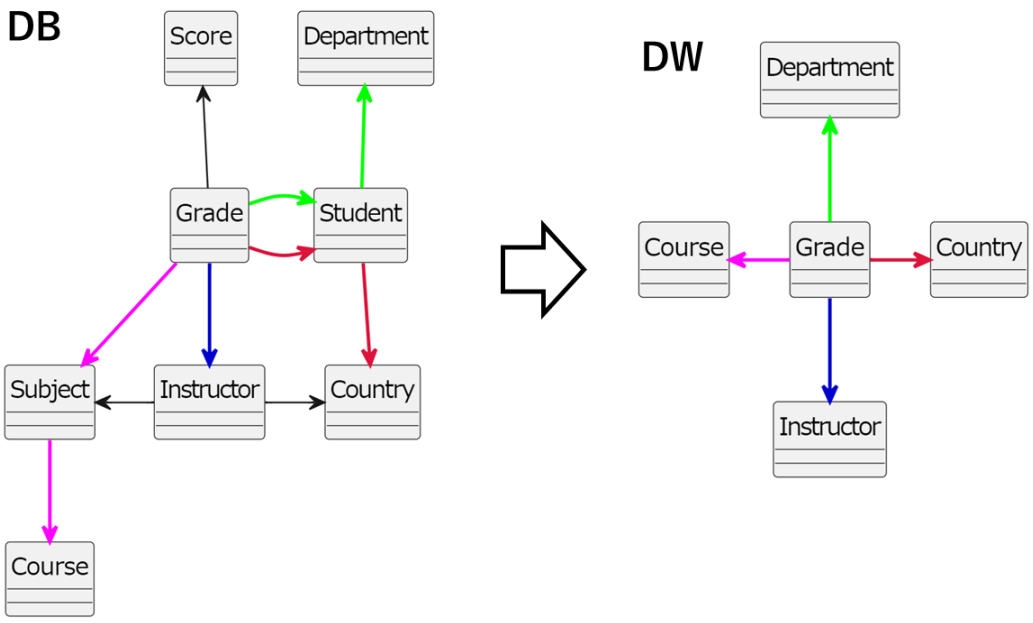

Figure 3: Schematic Image of integration of database to data warehouse.

Database instances change over time, and the data in data warehouses is updated accordingly. Because updates are made with preserving data integrity, it is guaranteed that data references in the data warehouse will match whether they were made before or after the updates. In other words, data can be accurately referenced in the data warehouse regardless of when updates occur. In category theory, this consistency is called “natural transformation." This property ensures accurate data references before and after updates, supporting reliable data management across the database and data warehouse.

5 Conclusion

This paper presents a method for constructing categorical databases to ensure data integrity in institutional research. Institutional research frequently relies on large and complex data sets drawn from various departments, which makes it difficult to maintaining data integrity. Using category theory, especially categorical databases, helps keep data relationships precise and mathematically consistent. This approach allows databases to be easily integrated into data warehouses without compromising data accuracy.

This study shows that when data is organized as categorical databases, it is possible to express relationships through paths that are easy to understand and can be checked mathematically. As a result, the sentences that describe these paths ensure the data integrity. This kind of dynamical data integrity is also supported by the natural transformation property, which keeps data references consistent, no matter when data updates happen. Therefore, categorical databases provide a practical and effective way to manage complex data structures while ensuring that data references are reliable in both databases and data warehouses.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Numbers JP22H00077.

References

[1] D.I. Spivak, "Functorial data migration," Information and Computation 217 (2012) pp. 31-51.

[2] D.I. Spivak and R. Wisnesky, "Relational Foundations for Functorial Data Migration," arXiv:1212.5303v7 [cs.DB] 24 Jul 2015.

[3] D.I. Spivak and R.E. Kent, "Ologs: a categorical framework for knowledge representation," arXiv:1102.1889v2 [cs.Lo] 7 Aug 2011.