Document Database Design Using ORM Conceptual Data Model

Wararat Jakawat*

Division of Computational Science, Faculty of Science,
Prince of Songkla University, Songkhla, 90110, Thailand

*Corresponding author. E-mail: wararat.j@psu.ac.th

ABSTRACT

Unstructured data is the most common type of data generated on a daily. Businesses must discover ways to retain such data to analyze and use it to make business choices. Document databases have emerged to the needs of applications. This database has a schemaless feature that allows for more flexibility in data management. Particularly, a document database does not define conceptual modeling. The conceptual data model is one of the essential phases in database development. The original data model makes it difficult to express queries on document databases. Therefore, developers need the conceptual model to understand business users and related data. This work provided a standard design strategy for document representation. ORM is utilized in this approach for conceptual modeling adopted through a document model. This paper defined a set of graphical symbols to document database concepts such as collection and two kinds of relationships. Furthermore, the current challenges in document database modeling were highlighted throughout the discussion of this topic.

Keywords: *Data Model, Conceptual Model, Document Database, Database design, ORM*

INTRODUCTION

The modern challenge of data management is to handle big quality and different types (structured and unstructured). Relational databases (RDB) have become the first choice for storing data. RDB requires support for a schema based on the relational model. There are two concepts of the model: tables and relationships. Data is organized into rows and columns in tables. Between the rows of the tables, relationships are built. There are constraints such as keeping data in a table being difficult, a fixed structure, or many relationships making data access slow. New technology, such as NoSQL databases, has arisen to address these issues in recent years. NoSQL or Not Only SQL databases provided a way for storing data that is different from RDB. Non-relational schema that scales horizontally data, flexible schema that allows dynamic insertion of any form of data, and BASE support are all features of NoSQL. Another problem that NoSQL can solve is the growing volume of data.

A conceptual data model is required for successful database use—it delicates how data in databases can be structured. Schemas can be used to express structure and constraints. Furthermore, the conceptual data model gives semantic objects that are like human reasoning. Many models are commonly used in relational

Article history:
Received 2 May 2022; Received in revised from 24 May 2022;
Accepted 7 June 2022; Available online 25 June 2022

database design, including Entity Relationship, UML, and ORM. As a result, the conceptual data model is essential, as it allows information structure to be expressed without depending on specific databases. In NoSQL databases, there is no such thing as a model. Only a few approaches to the conceptual data model of NoSQL databases have been investigated in the literature. The model is created using ER or UML. Object-Role Modeling (ORM) is not used in any work. ORM is a helpful way for conceptually creating database models. Non-technical users can easily understand the data structure. A conceptual model represents the business users in RDB, and there are rules to translate the conceptual model into an implementation schema. A many-to-many relationship in an ER model, for example, is modeled in an RDB by using a third table called a join table. There are several ways to describe the relationship in document databases, including two-way embedding, one-way embedding, and forming a join collection. As a result, what model can come closest to document database implementation structure?

The goal of this work is to define the conceptual model that will be used for document databases. ORM was used to create the model. The remainder of the paper is structured out as follows. The relevant research of conceptual data model applied to NoSQL are discussed in Section 2. The general notion of document databases was explained in Section 3. Section 4 describes the design of ORM for document databases as well as a data sample. Section 5 presents the conclusion and future work.

RELATED WORK

To conclude the related work about modeling document databases, this paper defines the existing research into two groups as follows:

- Transformation of relational model to document databases

  Many academics are investigating transformation rules to convert models from relational databases to document databases. The RDB rules have been used to present the transformation rules. Aftab et.al (2010) presented a transformation task automatically for MongoDB. Tamas *et.al* (2013) and Stanescu et.al (2016) proposed an algorithm to automatically derive to map from relational model to document model using MongoDB. Their algorithm denormalized to resolves functional dependencies between relations. This paper suggested a method for transforming entities and association relations using RDB metadata. Abdelhedi et.al (2017) formalized a set of transformation rules using the QVT language. The transformation rules were developed by Alotaibi *et.al* (2019) and are applied depending on the sorts of relationships. They focused on association, inheritance and aggregation relationships. At the logical level, their model automatically transformed UML conceptual models into NoSQL models. In the similar way, Roman et al. presented a mapping rule (Čerešňák et.al, 2021).

- Using conceptual model to document database design

  There are few studies that propose using an RDB model to design document databases. Li et al. proposed the data model based on the data structure and query requirements (Li et al, 2014). There is a model that proposed by using a specify a system independent realization of application data (Bugiotti et al., 2014). Banerjee et al. proposed a conceptual model that consists of a set of constructs, relationships and properties of various relationships (Banerjee and Sarkar, 2016). Their model is a three-layered organization, with Collection, Family, and Attribute being the top three tiers. They proposed NoSQL language validation rules. Shin et al. studied database design for NoSQL databases. They suggested using Peter Chen's UML conceptual model for document database design (Shin et.al, 2017). Their model maps the components of UML class diagram to document data model. The components of the UML class diagram are mapped to the document data model in their model. Class can be a set of documents. Attribute is mapped to columns. Relationships is mapped to reference or embedded documents. Similarly, Benmakhlouf et.al (2018) used UML as a tool for the conceptual model and the simple rules to transform it into the NoSQL model. Vera-Olivera et al (2021) presented a mapping method for determining the level of representation. Based on ER and UML, they propose a new notation.

The studies show that the transformation rules took into account the notions of the conceptual data model. Direct studies have also looked into how to turn the concept of ER into a document model. Researchers have adopted ER or UML for document models in the literature on conceptual models. To the best of our knowledge, none of these models recommend using ORM to create document database models.

DOCUMENT DATABASES

Document databases are adaptable and capable of storing enormous volumes of information. Document databases are made up of three primary components: collection, document, and attributes. A gathering of documents is referred to as a collection. Each document is identified by a set of key-value pairs and stored in a sequential order collection, with a new document being appended to the collection. A document's properties and structure are both contained within it. Different data can be found in an attribute. Each document in a collection can have its own set of fields. Relationships can be defined in two ways. The first is reference, which is the process of linking one document to another. Nested documents are the second type of relationship. The documents are contained in a different document.

Table 1. Terminology between RDB and document databases

RDB	Document databases
Specific schema	Dynamic schema
Table	Collection
Row	Document
Attribute	Field
One to one	Embedded or linking documents
One to many	Embedded, linking, or bucketing strategy for time series (Stanescu et al., 2016)
Many to many	Two ways embedding, one way embedding or join collection
PK and FK	Default key in database

Table 1 shows the terminology of the relationship between RDB and document database.

DESINGING DOCUMENT DATABASES USING ORM

In the database design phase, conceptual modeling is an essential foundation. The goal of ORM is to represent the information semantics of business domains using underlying facts of interest (Halpin Terry, 2010). ORM features a graphical interface for creating conceptual models. Applying ORM in a document database design is the recommended conceptual approach. The data model represents data that specific domains are interested in in a document database and access to a related set of data.

A. Proposed conceptual model

The suggested conceptual model consists of a standard set of collections, collection types, fields, and relationships (embedding or referencing) to unify a conceptual level of document databases. The model's organizational structure is depicted in Figure 1. This work, according to ORM, formalizes the model's components as follows:

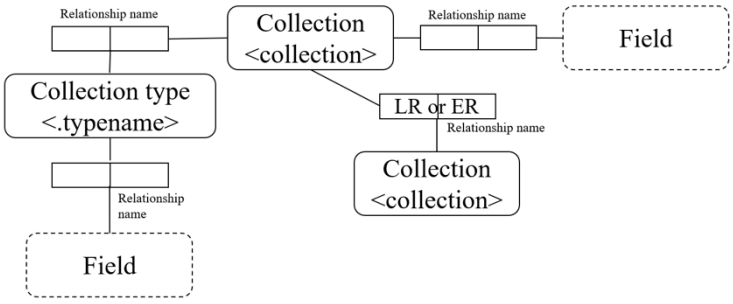

Figure 1 Proposed conceptual model

Collection.

A collection is depicted as a name such as person or product. The use of identification collection provides a symbol as a table symbol in RDB. Formally, a collection C is a triple (C_{CT}, C_C, C_F) where:

- C_{CT} is the name of collection type. It may make a many types. For example, person collection can be categorized into two types: students and teacher. If there is one types, a model does not need collection type.
- C_C is the set of another collection.
- C_F is a set of fields.

Collection types.

Formally, a collection type CT is a pair of (CT_N, CT_F) where:

- CT_N is a name of collection types.
- CT_F is a set of fields.

Fields.

A field is an attribute or a column in term of RDB. Formally, a simple field SF is a couple (SF_n, SFv) where:

- SF_n is the name of simple field.
- SF_v is the value of the field.

Relationships.

There are two types of relationships: linking relationship LR and embedded relationship ER. Formally, each relationship R is a pair if (R_T, R_n) where:

- R_T is the types of relationship.
- R_n is the name of a relationship.

B. Transform ORM into document data model

This section describes the graphical representation of document model lists of the main graphical symbols by adapting ORM notation to document notation. The collection represents a soft rectangle, and figure 2(a) is room table. The proposed symbol includes displaying the collection mode in parentheses for a collection. The room collection is shown in figure 2(b). The collection type symbol is shown in figure 2(c). Field symbol is depicted as a rectangle, and shape has dash lines. An example of a field is room number, as shown in figure 3. ORM displays a primary key to an entity symbol by writing its name under an entity's name with a bracket. For the document model, this paper uses the same way.

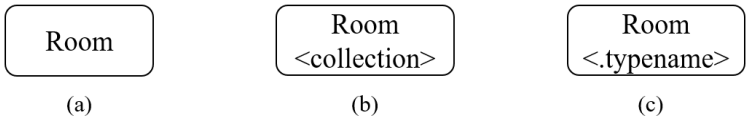

Figure 2 Entity in ORM (a), Collection (b) and Collection type (c)

Figure 3 Column in ORM (a) and field (b)

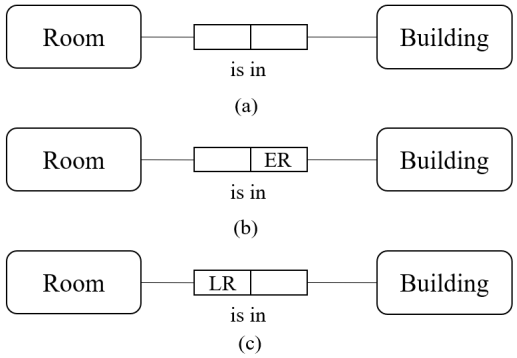

Figure 4 Relationship in ORM (a), and embedded relationship (b) and referencing relationship (c)

To talk about relationships in RDB can play a role. Roles are given role names. Graphically, the objected relationship is enclosed in linking between collections. Role name displays on the top of the symbol. Types of relationships are displayed in the symbol. This paper defines that LR represents a referencing relationship and ER represents an embedded relationship. Figures 4(b) and 4(c) show two types of relationships. For example, ER closes to building collection in figure 4(b). Building collection is a collection that is inside room collection. While LR is close to room collection, it shows that a room's location concerns a building.

C. Running example

Figure 5, an ORM used as an example, depicts the conceptual data model utilized in document database design. For example, the model employed is one of room activity, in which a room is used for a meeting or a lecture. The room number and the building in which it is located are used to identify a room. The combination of room number and building refers to only one room. Types of rooms can be rooms for meetings or lectures. A room is

reserved for an activity at a given time. For each activity, only one room can be reserved.

According to the proposed concept, the result of changing the ORM conceptual data model into a document data model is shown in figure 6. The model gets closer to the implementation model, making it easier for a database designer to comprehend the implementation structure.

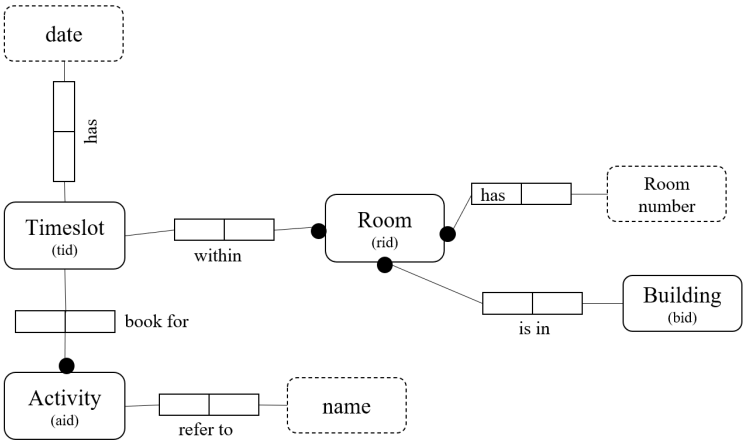

Figure 5 ORM conceptual model

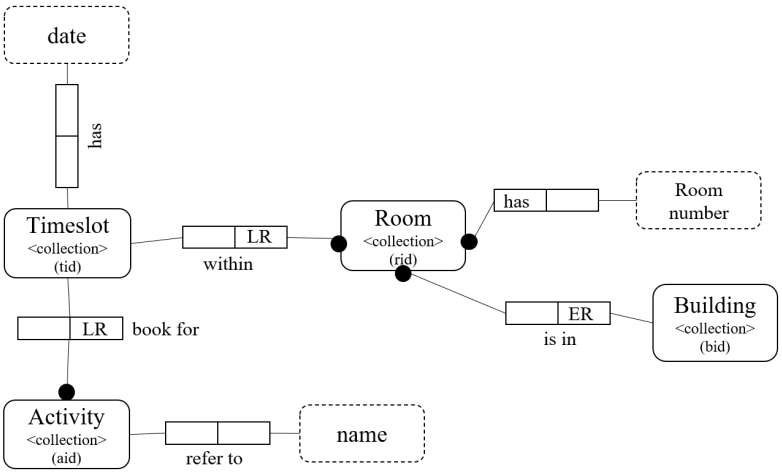

Figure 6 Document model using ORM

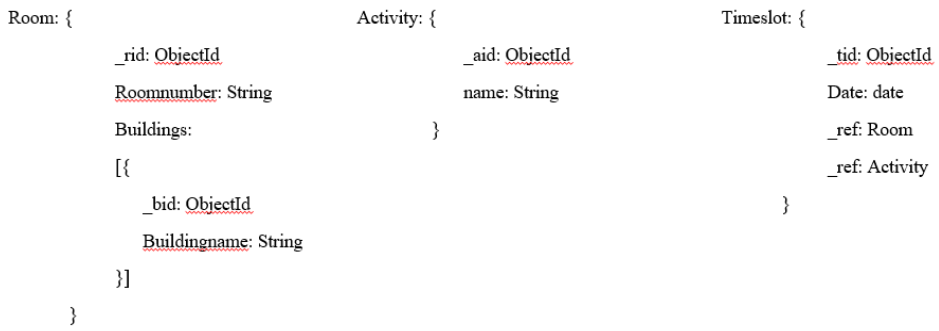

Figure 7 The physical model

D. Transformed document data model into physical model

The structure of document databases represents all relationships that are linking or embedding. This study aids database designers in their work. Figure 7 shows a physical model of the application in conceptual model from figure 6. So, all relationships, which are linking or embedding that depends on query patterns, are represented into "LR" for timeslot relationship linked to room and "ER" for building embedded into room in figure 6 above.

VALIDATION METHOD

This section aims to study the understanding of the database's semantics and to write the queries. This paper applied the model to four different case studies to perform the proposed conceptual model. All case study models allow displaying to two student developers a conceptual model and physical model, and the student developers do not know the data model of concerned applications. Each model, the result indicates the average time of writing the queries according to (1) with the conceptual model of ORM, (2) with the conceptual model of ORM and physical model, (3) with the proposed conceptual model, or (4) with the proposed conceptual model and physical model.

Each student developer writes three queries for a model: each query concerns different types of relationships. The first query applies a filter to a collection and returns fields in other collections. The second one applies to a collection and returns filed in the embedded collection. The last query applies to a filter collection and returns fields in other collections and embedded collections. Each database is associated with a set of queries. The calculated average time of writing queries in each situation is shown in Table 2. The result is that the semantic information of a database and data structure helps the student write queries faster. The average time of the proposed model is better than the ORM. Note that the uses of models and applications are probably with students' experience. The purpose of

using the data model is to communicate to the developers quickly. Therefore, the model must be simple enough for developers to grasp the semantics and structure.

Table 2. Queries writing time

Student	ORM	ORM and physical model	Proposed model	Proposed model and physical model
1	40 mins (DB1)	28 mins (DB3)	20 mins (DB2)	16 mins (DB4)
2	34 mins (DB2)	22 mins (DB4)	18 mins (DB1)	15 mins (DB3)
Average	37 mins	25 mins	19 mins	15 mins

CONCLUSION AND FUTURE WORK

The conceptual document model is the topic of this paper. The model is based on the object-role model. This paper presents a collection of concepts for creating a conceptual document model using an ORM model as a starting point. The ORM concept demonstrates that it can be used to construct document databases. The model includes a document database's collection, collection type, filed, and relationships concepts. In addition to relationship kinds, building document structure is considered for the implementation model. Future work plans to complete the model to consider the constraints, validation rules, and semantic relationships when the conceptual model is transformed into the logical model.

REFERENCES

Halpin, T. (2010). Object-role modeling: Principles and benefits. International Journal of Information System Modeling and Design (IJISMD), 1(1), 33-57.

Tamas, V., Péter, F., Krisztián, F., and Hassan, C. (2013). Denormalizing data into schema-free databases. International Conference on Cognitive Infocommunications, 747–752. doi:10.1109/CogInfoCom.2013.6719198.

Li, X., Ma, Z. and Chen, H. (2014), QODM: A query-oriented data modeling approach for NoSQL databases. IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), 338-345.

Bugiotti, F., Cabibbo, L., Atzeni, P. and Torlone, R. (2014), Database Design for NoSQL Systems. International Conference on Conceptual Modeling, 223-231.

Banerjee, S. and Sarkar, A. (2016). Modeling nosql databases: From conceptual to logical level design, The 3rd International Conference on Applications and Innovations in Mobile Computing, 12(5), 632-636. http://www.ripublication.com.

Stanescu, L., Brezovan, M., and Burdescu, D. D. (2016). Automatic mapping of mysql databases to nosql mongodb. Federated Conference on Computer Science and Information Systems (FedCSIS), 837–840.

Abdelhedi, F., Ait Brahim, A., Atigui, F., and Zurfluh, G. (2017). MDA-based approach for NoSQL databases modelling. Big Data Analytics and Knowledge Discovery, eds. L. Bellatreche and S. Chakravarthy (Cham: Springer International Publishing), 88–102.

Shin, K., Hwang, C., and Jung, H. (2017). Nosql database design using UML conceptual data model based on peter chen's framework, International Journal of Applied Engineering Research, 12, 632–636.

Benmakhlouf, A. (2018). Nosql implementation of a conceptual data model : UML class diagram to a document oriented model. International Journal of Database Management Systems 10, 1–10.

Vera-Olivera, H., Guo, R., Huacarpuma, R. C., Silva, A. P. B. D., Mariano, A. M., and Holanda, M. (2021). Data modeling and nosql databases a systematic mapping review. ACM Computing Surveys, 54, 1–26. https://doi.org/10.1145/3457608.

Aftab, Z., Iqbal, W., Almustafa, K. M., Bukhari, F., and Abdullah, M. (2020). Automatic nosql to relational database transformation with dynamic schema mapping. Scientific Programming, 2020. https://doi.org/10.1155/2020/8813350.

Alotaibi, O. and Pardede, E. (2019). Transformation of schema from relational database (RDB) to NoSQL databases. Data, 4(4),148. https://doi.org/10.3390/data4040148.

Abdelhedi F., Brahim A. A., Ferhat, R. T., Zurfluh, G (2020). Discovering of a Conceptual Model from a NoSQL Database. 22nd International Conference on Enterprise Information Systems, INSTICC: Institute for Systems and Technologies of Information, Control and Communication, 61-72.

Čerešňák, R., Dudáš A., Matiaško K. and Kvet M. (2021) Mapping rules for schema transformation : SQL to NoSQL and back. International Conference on Information and Digital Technologies (IDT), 52-58. 10.1109/IDT52577.2021.9497629.