Relational Data Modeling on the Document-Based NoSQL

Muhammad Riza Alifi[1], Transmissia Semiawan[2*], Djoko C.U. Lieharyani[3], Hashri Hayati[4]

***Abstract*—Data management technology that continues to develop and boost the popularity of document-based not only structured query language (NoSQL) has become the most-used data model. Behind its popularity, data management technology offers an intriguing advantage, namely flexible data storage, whether in terms of data forms and sizes or structured and unstructured data. However, this data modeling flexibility has its challenge due to its impact on more complex scheme creations, without being accompanied by any need-based design patterns. This study aims to model relational data on the document-based NoSQL at its conceptual, logical, and physical levels. The conceptual design was developed based on processes, rules, and business requirements. The logical and physical designs were developed based on the extended references and computed design patterns determined from the operating workload. The relational data model design on the document-based NoSQL was successfully formed using the entity relationship diagram (ERD) with Chen notation for the conceptual, and collection relationship diagram (CRD) for both logical and physical levels. The conceptual design focused on the representation of entities, attributes, and relationships. Unlike the conceptual design which tends to be abstract, the focus of the logical design is on the collection schema (embedded and reference) representation, including design patterns influenced by the formation of relationships. Furthermore, the focus of physical level design is to represent the schema in a more concrete form. The physical design is almost the same as the logical one, the difference lies only in the detail addition for data types and structures. The evaluation of data model designs was also carried out for each level. This study contributes to designing a data model with the advantage of read-intensive capability since a joint operation among collections is not required and the computation process recurrence for derivative attributes is not necessary.**

***Keywords*—Document-Based NoSQL, Data Modelling, Extended Reference, Computed.**

I. INTRODUCTION

The current development of data management in a database must be able to respond to the challenges of big data. The “5 V” challenges that are known to dominate practices and experiments in big data are volume, velocity, variety, veracity, and value [1]. Several issues related to challenges in data management are, among others, consideration of centralized or distributed data storage; data access based on information retrieval or data retrieval; determination of data search operation using pattern matching or exact string matching.

[1,2,3,4] *Department of Computer Engineering & Informatics, Bandung State Polytechnic, Gegerkalong Hilir Street, Ciwaruga Village, West Bandung District, Mail Box 1234, Postal Code 40559, West Java, INDONESIA (tel.: 022-2013789; fax: 022-2013889; email: [1]muhammad.riza@polban.ac.id; [2]t.semiawan@polban.ac.id; [3]hashri.hayati@polban.ac.id; [4]djoko.c.utomo@polban.ac.id)*
[*]*Corresponding Author*

[Received: 28 December 2021, Revised: 8 July 2022]

Data modeling is the base in responding to these challenges. It is the most essential segment in the design and development process of a data system. It offers a technique to illustrate information needs in the real world in a way that can be understood by users and can assist designers in implementing information needs into the database system [2]. Data modeling can provide an illustration in conceptual, logical, and physical models [1]-[5]. A conceptual model of a structured query language (SQL) database is used to represent an actual concept of a system that is acquired based on a client’s needs [6]. The next stage is to transfer the conceptual model to the logical stage by converting its entity, attribute, and relationship set into the relational scheme which explains its table and structure, as well as the relationship among tables. The logical model is then converted to a physical stage by creating a relational scheme according to the selected database management [7].

Many types of data modeling have been developed, starting from the relational data model to the graph data model [3], [4]. It is in line with the development of the structured relational database management as well as non-relational large-scale databases included in the big data process [8]. Currently, the management of non-relational data that uses not only SQL (NoSQL) has replaced database management with SQL that dominated the computation of relational database management for decades [8], [9].

In general, NoSQL’s approach is different from SQL, especially in terms of data storage and access [9]. It is implemented using four data modeling categories, namely key-value based, document-based, column-based, and graph-based [3], [8], [10]-[13]. By using this model, NoSQL is not limited to only ease analytic processing, but also handle transactional processes managed by the relational data model [3], [10], [14].

As the database management evolves from SQL to NoSQL, data modeling technology tools also develop. Several studies related to data modeling tools that have been used are entity relationship diagram (ERD), collection relationship diagram (CRD), document data model (DDM), unified modeling language (UML)-class diagram, NoSQL abstract model (NoAM) [3], [14], [15].

ERD is one of tools to model relational data representing data structure and information that can be implemented on a database at conceptual, logical, and physical stages [3]. There are also several relational data modeling tools, such as CRD, that can present data structure at logical and physical stages; DDM at logical stage; UML-class diagram at conceptual stages; and NoAM that is often implemented at logical and physical stages.

Reference [14] used CRD for document-based NoSQL data modeling as CRD can represent logical as well as physical stages that are close to the implementation. Components shown on the logical stage CRD are entity, embedded and reference relationship, and the cardinality of relationship. Meanwhile, the

TABLE I
BUSINESS PROCESSES

Code	Business Process	Description
PB-01	KoTA Registration	KoTA registers with the condition that it has a maximum of three members for D3 and two members for D4. Each KoTA that registers are required to submit the topic of the TA and the title of the TA, supervisor candidate, as well as member data. Students can only be registered in one KoTA. TA topics are selected from the list of topics provided and can be more than one.
PB-02	KoTA Code Assignment	The process of assigning a Code to each KoTA with the provisions of a certain structure showing the students' study programs and the Code serial numbers were based on the smallest order Student Identification Number (*Nomor Induk Mahasiswa*, NIM) of the KoTA members. The KoTA Code for each academic year is unique.
PB-03	Supervisor Arrangement	Supervisor 1 and Supervisor 2 for each group are determined by considering the TA topics and topics that are align with the lecturers' interests. Each lecturer can choose more than one topic from the list of given topics.
PB-04	Supervisor Load Calculation	The supervisor's load is the number of KoTA that are supervised by a lecturer.

physical stage CRD has all the components of the logical stage added with applied design patterns, structures, and data types. This study used schema versioning, subset, computed, bucket, and extended reference patterns. Beside explaining the use of logical and physical stage diagrams, this study also briefly shows the use of simple ERD (without attributes) to represent the conceptual stage diagram.

Reference [14] has explained the need to have a NoSQL data model to minimize errors during the database implementation, especially in ensuring the atomicity of important operations and better performance by synchronizing operational priorities. Reference [15] has proposed NoAM as an abstract data model that is universal toward the NoSQL database. NoAM exploits the same elements at each NoSQL database and attempts to balance all differences and varieties that have been found. NoAM consists of three components, namely 1) database, which is a group of collections; 2) collection, which is a group of blocks with unique identities; and 3) block, which is a group of entries that are not empty. Each entry has a pair of entry keys and entry values. The uniqueness of this study is that it explains an abstract model that can represent an array and object of data structure that is usually found in physical data models.

Reference [3] has developed a document-based NoSQL data model at the logical stage in DDM through a transformation process from conceptual stage CDM represented by the modified UML-class diagram. The mapping of each element in DDM and CDM are 1) class with collection; 2) attribute with

TABLE II
BUSINESS RULES

Code	Description
BR-01	Each KoTA must have two supervisors.
BR-02	The maximum number of KoTA members for D3 is three people and for D4 are two people.
BR-03	Maximum length of NIM is nine digits.
BR-04	Maximum length for thesis title submission is fifteen words.
BR-05	The maximum number of submissions for proposed supervisors is five people and the minimum number is three people.
BR-06	Each KoTA will be given a KoTA Code with the following structure: "KoTA-" [five digits] + <code_program_class> [one digit] + <serial_number> [two digits]. Determination of <code_program_class > is: class A D3: 1; Class B D3: 2; Class A D4: 3; Class B D4: 4.

column; and 3) association with embedded or reference. This study only examined conceptual and logical stage data models; meanwhile, physical stage data models were not yet addressed.

Studies above shows that data modeling can be performed using modeling tools such as ERD, CRD, DDM, UML-class diagram, and NoAM. The modeling tool development cannot be separated from how the database is being used, from the development process, storage, to database access. It also considers the characteristics of existing data. A data model that has been developed by any tool must be able to represent or abstract the design of database characteristics and behaviors that will be used by certain applications to create information necessitated by application users. The data model must be visible from conceptual point of view (showing the application functionality) and logical and physical points of view (showing the technical process) from the user of the database.

With a more complex data structure, NoSQL data modeling becomes a considerable challenge. Especially on the document-based NoSQL, tools that can precisely represent or model data structure and information conceptually, logically, or physically are not yet available. This NoSQL data model must be able to manage shape and size flexibilities of structured, semi-structured, and polymorphic data sizes [10].

This study attempts to conduct relational modeling on the document-based NoSQL data structure. The modeling was done at the conceptual, logical, and physical stages that were applied on the Aplikasi Pengajuan Tugas Akhir (Thesis Submission Application) by the Department of Computer Engineering and Informatics (Jurusan Teknik Komputer dan Informatika, JTK), Bandung State Polytechnic. The data model diagram employed the results of several studies' synthesis [3], [14], [15] to be implemented on the MongoDB database management. The design made by this study will contribute to the D4 Information Technology Program of JTK, specifically as a case study for the development of web-based software.

II. METHODOLOGY

This study adopted a modified flexible study method [14] and [16] as the basis for data model designing process for all

TABLE III
BUSINESS NEEDS

Code	Description
REQ-F-01	The application can handle the addition of KoTA data for D3 and D4 study programs (NIM, Name, Department, Study Program, Class, Topic, Title and Supervisor).
REQ-F-01-01	The application can store all the KoTA entered by the user in the database.
REQ-F-01-02	The application can check the duplication of data that has been entered previously.
REQ-F-02	The application can automatically generate KoTA Code.
REQ-F-02-01	The application automatically generates KoTA Code after all KoTA in D3/D4 study programs have finished their submission.
REQ-F-02-02	The application can take one NIM of the smallest member of each KoTA member in each class, then sort the NIM data from all KoTA in each class.
REQ-F-02-03	The application can generate KoTA Code according to [BR-06].
REQ-F-03	The application can display all KoTA data that has been given a Code.
REQ-F-04	The application can match data between lecturer topics with student topics
REQ-F-05	The application can help in setting supervisor assignment data for each KoTA.
REQ-F-05-01	The application can display a list of lecturers having the same topic with KoTA.
REQ-F-05-02	The application can display a list of lecturer's data to be selected as a supervisor.
REQ-F-05-03	The application can remove the selected lecturer data on Supervisor 1 from the list of lecturer data to be selected as Supervisor 2.
REQ-F-05-04	The application can bring up a simple recap containing a list of all KoTA with a predetermined supervisor.
REQ-F-06	The application can calculate the load of each lecturer who has been assigned as supervisor.
REQ-F-06-01	The application can calculate the Supervisor Load after all supervisors is assigned to all KoTA
REQ-F-06-02	The application can calculate the Supervisor 1 load for each lecturer.
REQ-F-06-03	The application can calculate the Supervisor 2 load for each lecturer.
REQ-F-06-04	The application can calculate the KoTA load for each Lecturer.
REQ-F-07	The application can display a list of supervisor load data from each lecturer.

TABLE IV
OPERATION LIST

Code	Operation	Type	Description
OP-01	Loading data on the KoTA application form.	Read	Containing all Topic, Prospective Supervisors, and Students data.
OP-02	Saving data on the KoTA application form.	Write	Saving the Topic, Title, Prospective Supervisor, NIM data of each Student.
OP-03	Loading data on the supervisor lecturer appointment form.	Read	Containing only one KoTA data that has been submitted, including the NIM, Name, Topic, Title, and Supervisor.
OP-04	Saving data on the supervisor's appointment form.	Write	Saving the Topic, Title, Prospective Supervisor, NIM data based on the selected KoTA.
OP-05	Loading sorted KoTA data to set KoTA code.	Read	Loading KoTA data that has been sorted by: 1. smallest NIM in KoTA 2. smallest NIM in all KoTA grouped by Study Programs and Classes.
OP-06	Saving the result of the assignment KoTA code.	Write	Saving data from the determination of the KoTA code accompanied by the following rules: 1. based on the results of sorting, 2. based on <code_program_class >.
OP-07	Loading group data that has been determined by the supervisor.	Read	Containing all data for the KoTA that has been submitted, including the KoTA Code, NIM, Name, Topic, Title, Supervisor 1, and Supervisor 2.
OP-08	Loading data on lecturers who have been assigned as supervisor.	Read	Containing all data on the load of lecturers who have been assigned as Supervisor, including Lecturer Code, Lecturer Name, Supervisor 1 Load, Supervisor 2 Load, Total Supervisor Load.

abstraction stages (conceptual, logical, and physical). In developing a logical and physical data model, workload was needed as comparison to decide relationship (embedded and reference), data structure, and design pattern. This study was conducted in several steps: literature review, data modeling planning, implementation or data model development, and data model evaluation.

A. Literature Review

This stage involved literature study on the document-based NoSQL data modeling at each abstraction stage (conceptual, logical, and physical stage), including identifying problems based on theoretical analysis results.

B. Planning

This stage incorporated preparations before the modeling development was done.

1) Identifying Business Processes, Rules, and Needs: The case used in this study was the actual development of Aplikasi Pengajuan Tugas Akhir at Bandung State Polytechnic. The conceptual model represents deeper abstraction in the scenario of its use [17]. To understand better abstraction or problem domain in the case, a scenario or description of the problem domain, that could be analyzed from its business processes, rules, and needs that can be predetermined, was needed. The data modeling design for the application that was developed

TABLE V
QUANTIFICATION AND QUALIFICATION OF OPERATIONS

Code	Workload	
	Quantification	***Qualification***
OP-01	20-35 read/day < 3 Seconds	No past data.
OP-02	20-35 write/day < 3 Seconds	Low workload.
OP-03	20-35 read/day < 3 Seconds	No past data.
OP-04	20-35 read/day < 3 Seconds	Low workload.
OP-05	3-10 read/action < 5 Seconds	• No past data. • Potentially high workload involving two data, namely: Student and KoTA.
OP-06	3-10 write/day < 5 Seconds	• Periodic writing. • Potentially moderate workload, involving only KoTA data.
OP-07	20-35 read/day < 5 Seconds	• No past data. • Potentially high workload involving four data, namely Student, KoTA, Lecturer, and Topic.
OP-08	20-35 read/day < 5 Seconds	• No past data. • Potentially high workload involving two data, namely KoTA and Lecturer.

included four business processes (Table I), six business rules (Table II), and twenty business needs (Table III).

The following are assumptions that limit the scope of the case study.

- Data of supervisors and topics that are relevant with the supervisors' expertise.
- Data of students are available but not yet assigned as members of Thesis Group (Kelompok tugas akhir, KoTA).
- As part of data validation, integrity constraint and atomic transaction between collections are implemented on the application.

Conceptual modeling is an activity that formally illustrates several aspects of the physical and social worlds among us for the purpose of understanding and communication. Conceptual model concentrates on illustrating entity, data type, relationship, user operation, and constraint [17]. Identifying entities on the conceptual model highly considers the description or explanation of business processes, rules, and needs in the case study as the model's actual illustration. Furthermore, a scenario was also needed to measure successful results of the illustration or abstraction of the above case study in the logical and physical models.

2) Identification of Workload: Workload plays an important role, especially in developing logical and physical stages of data model. The workload in this study was the measurement of the operational intensity, including information query that was appropriate for the application's business. Measurement was done based on the estimation of the implementation intensity, standard processing time, and quality of each operation. Through this measurement, the design could be viewed better and more appropriately since each operation could be measured based on its operation time in accordance with the given design pattern and data structure model. For this reason, a) the operation involved in the application (Table IV), and b) the quantity and quality of the operation (Table V) must be determined.

The operation must be predetermined to see the high potential operational time based on the given data structure. By looking at the high-level potential, the design pattern was expected to be more efficiently applied. Table IV shows a list of operations that must be present on the application based on business processes, rules, and needs. The list of operations would be executed by the database machine.

The operational quantification aims to measure the estimated value of routine operations being implemented and decision of standard time for the operations. Meanwhile, the operational quality is a description showing the characteristics or qualitative urgency level of the operation. The quantification and qualification of operations were determined based on the experiment of the application analysis that was previously conducted. Table V displays the results of quantification measurement value and qualification assessment with the assumption that each operation is implemented at the time the TA is submitted.

3) Identifying Design Pattern: In a document-based NoSQL, the relationship modeling among collections usually contains 1) 1-to-1 with embedded; 2) 1-to-*N* with embedded or reference; and 3) *M*-to-*N* with embedded or reference. Each embedded and reference collection structure representing the relationship among collections has strengths and weaknesses. Embedded organized data based on redundant structure provides an ease in acquiring and manipulating related data only with a single operation. Meanwhile, references with nonredundant structure (same as relational data model in general) must jointly operate for each related collection. Naturally, using this joint operation will result in the decrease of reading operation performance.

In this study, there were two types of relationships, namely 1-to-*N* and *M*-to-*N*. However, in a document-based NoSQL, both can create the same collection structure due to the existence of an array of data structure support. Moreover, there were three operations with high-level workload, which was used as reference in considering the design pattern.

The application of a pattern design can provide freedom for the collection structure by aligning it with application needs so that it can be flexible in using embedded and reference. There are two design patterns that can be applied, namely extended and computed reference.

The extended reference allows the use of both embedded and reference collection structures [14]. This design pattern was considered based on the result of workload analysis in Table IV and Table V, which attempted to avoid the use of joint collaboration among collections by using embedded, specifically for operations with potentially high workload. In this case study, high workload operations were found in the KoTA and supervisor data. Furthermore, this pattern continued to maintain the subcollection to use reference toward the linked collection. Therefore, if only certain documents were needed, it was not necessary to access all documents in the subcollection. It could be done by directly accessing the collection document that had been linked by the subcollection.

The following is an illustration of the comparison between reference and extended reference data structure using JavaScript Object Notation (JSON). format.

<u>*Reference*</u>

```
A1 = {a0, a1, a2, a3, a4};

B1 = {b0, b1, b2, a0};
B2 = {b3, b4, b5, a0};
B3 = {b6, b7, b8, a0};
```

In the data structure above, A1 is a document in collection A with an ID value of a0. B1, B2, and B3 are documents in the collection B that conduct reference toward document A1 based on the ID owned by document A1. The process of obtaining document in collection A and B can be illustrated using relational algebra by incorporating joint operator as with natural join (⋈): A⋈B.

<u>*Extended Reference*</u>

```
A1 = {a0, a1, a2, a3, a4,
      [
        {b0, b1, b2},
        {b3, b4, b5},
        {b6, b7, b8}
      ]
    };

B1 = {b0, b1, b2, a0};
B2 = {b3, b4, b5, a0};
B3 = {b6, b7, b8, a0};
```

In data structure above, A1 is a document that conducts embedded, thus there are subdocuments B1, B2, and B3 in it. Documents B1, B2, B3 that operate reference to A1 can maintain the use of ID for A1, which is a0, or can also be deleted. The process of obtaining documents in collection A and B that apply extended reference design patterns do not need to involve joint operation: A.

Computed aims to reduce the intensity of repeat computation. In this case, computed was logically implemented to calculate the load of supervisor on lecturer, based on the role of lecturer in KoTA. Therefore, during the query to input data on Supervisor_Load, it only needed to read, while computation to recalculate was not necessarily needed.

C. Data Modeling

The flexibility of document-based data models can cause the scheme to become more complex. Therefore, in executing data modeling, the business process, rules, needs, and workload must be considered in the operation. The data modeling that are done are:

1) Conceptual Data Modeling Design: Conceptual data modeling must illustrate the big picture of the application or system, including how to organize, and its link with business rules and needs. This stage aims to identify conceptual objects and their attributes as well as identify the relationships among objects that fully characterize the behavior of the application being developed. The identification was done by referring to the predetermined business process, rules, and needs [18]. Modeling tools used for conceptual data models were ERD with Chen notation.

2) Logical Data Modeling: Logical data model is less abstract and must be able to sufficiently illustrate in detail the technical implementation of the created conceptual model. The modeling tools used for logical data models was logical CRD.

3) Physical Data Modeling: Physical data modeling must be concretely illustrated according to the targets of the database machine that is used; in this study, MongoDB was used. The modeling tool used in this physical data model was physical CRD.

D. Evaluation

Evaluation or data model design analysis was sequentially implemented, starting from the most abstract phase to concrete phase. The initial phase or prior must be completed to continue to the next phase. Each data model plan had different checklists in accordance with the visual reference and table used.

1) Conceptual Data Model Design Evaluation: It was implemented by mapping the appropriateness between the conceptual data model and business processes, rules, and needs determined based on the created conceptual object.

2) Logical Data Model Design Evaluation: It was implemented by mapping the appropriateness between the conceptual data model and design pattern that had been determined based on workload.

3) Physical Data Model Design Evaluation: It was implemented by mapping the appropriateness of the logical data model toward the collection, field, data type, and data structure that was defined on the physical data model. The evaluation scope on the physical data model did not include a direct examination on the database machine.

III. Results and Evaluation

A. Results

1) Conceptual Data Modeling: Based on the analysis of business processes description (Table I), business rules (Table II), and business needs (Table III), several conceptual objects required to develop Aplikasi Pengajuan Tugas Akhir were identified. These objects were four entities with twelve attributes and four relationships (Table VI). The relationship showed the logical link among entities, with attribute and cardinality. Cardinality symbol one-to-many was marked by 1-to-*N*, and many-to-many is marked by *M*-to-*N*.

Moreover, one descriptive object was identified as the attribute Role in the relationship between entities Lecturer and KoTA. This descriptive attribute aims to show that there are three types of supervisor identification status, namely 1) Lecturer that was proposed by KoTA, but not yet or not determined as a supervisor; 2) Lecturer that have been assigned

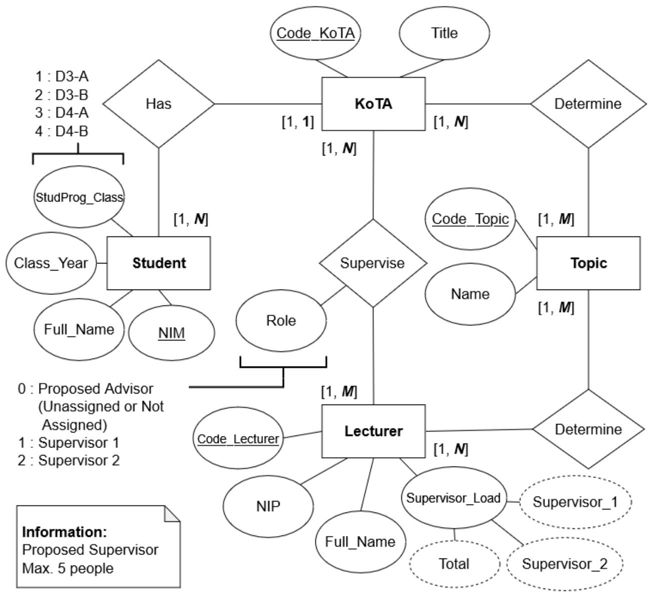

Fig. 1 Conceptual data model of Aplikasi Pengajuan Tugas Akhir.

TABLE VI
LIST OF ENTITIES, ATTRIBUTES, AND RELATIONSHIPS

No.	Entities	Attributes	Relationships (Reference to Entity)
1.	Student	• NIM • Full_Name • Class_Year • StudProg_Class	--
2.	KoTA	• Code_KoTA • Title	• Student (KoTA.Code_KoTA): 1-to-*N*
3.	Lecturer	• Code Lecturer • NIP • Full_Name • Supervisor_Load ○ Supervisor_1 ○ Supervisor_2 ○ Total	• KoTA (Lecturer.Code_Lecturer): *M*-to-*N* • Attribute Descriptive: Role
4.	Topic	• Code_Topic • Name	• KoTA (Topic.Code_Topic): *M*-to-*N* • Lecturer (Topic.Code_Topic): *M*-to-*N*

as Supervisor 1; and 3) Lecturer that have been assigned as Supervisor 2.

The entity, attribute, and relationship that were identified using ERD modeling tools were able to well illustrate the behavior of the application from the point of view of the data modeling. In Fig. 1, the Supervisor entity was considered to have the composite attribute, namely Supervisor Load, which was the derived attribute for Supervisor_1, Supervisor_2, and Total. Besides, there was the descriptive attribute Role on the relationship Supervision between Lecturer-KoTA to determine Lecturers as Supervisor 1, Supervisor 2, or not yet decided.

TABLE VII
APPLICATION OF DESIGN PATTERNS

No.	Design Patterns	Related Relationships /Attributes	Operation Code	Information
1.	Extended Reference	KoTA-Student (1-to-*N*)	5, 7	Student reference and embedded to KoTA.
		Lecturer-KoTA (*M*-to-*N*)	7	KoTA reference to Lecturer and Lecturer embedded to KoTA.
		Topic-KoTA (*M*-to-*N*)	7	KoTA reference to Topic and Topic embedded to KoTA.
		Topic-Lecturer (*M*-to-*N*)	7	Lecturer reference to Topic and Topic embedded to Lecturer.
2.	Computed	Lecturer. Supervisor_1 Calculate ((KoTA.Role = 1))	8	Calculate the load of Supervisor 1.
		Lecturer. Supervisor_1 Calculate ((KoTA.Role = 2))	8	Calculate the load of Supervisor 2.
		Lecturer.Total (Lecturer. Supervisor_1 + Lecturer. Supervisor_2)	8	Calculate total load of the Supervisor.

2) Logical Data Modeling: In this modeling type, all entities of the conceptual data model were implemented as a collection. Using CRD tools, the extended reference and computed design pattern were applied on this logical data modeling. Table VII depicts the application of the design patterns which was linked to the workload-based operation.

Fig. 2 depicts each relationship that had been created among collections as 1-to-*N* and *M*-to-*N* using the extended reference pattern. There were differences in the implementation of extended reference for both relationships. The KoTA-Student (1-to-*N*) or Student-KoTA (*N*-to-1) relationship, in the Student collection (child collection), contained the Kode_KoTA field as the reference for Kode_KoTA in the KoTA collection (main collection). Meanwhile, in the *M*-to-*N* relationship for Lecturer-KoTA, Topic-KoTA, and Topic-Lecturer, the specific field as reference had not been allocated. Moreover, there was a computed pattern on the entire attribute in the Supervisor Load subcollection in Lecturer collection. By applying the extended reference pattern, there was a read-intensive operation on the relationship operation (operation 5 and 7 in Table VII). It also occurred in the application of a computed pattern on the supervisor load calculation process (operation 8 in Table VII).

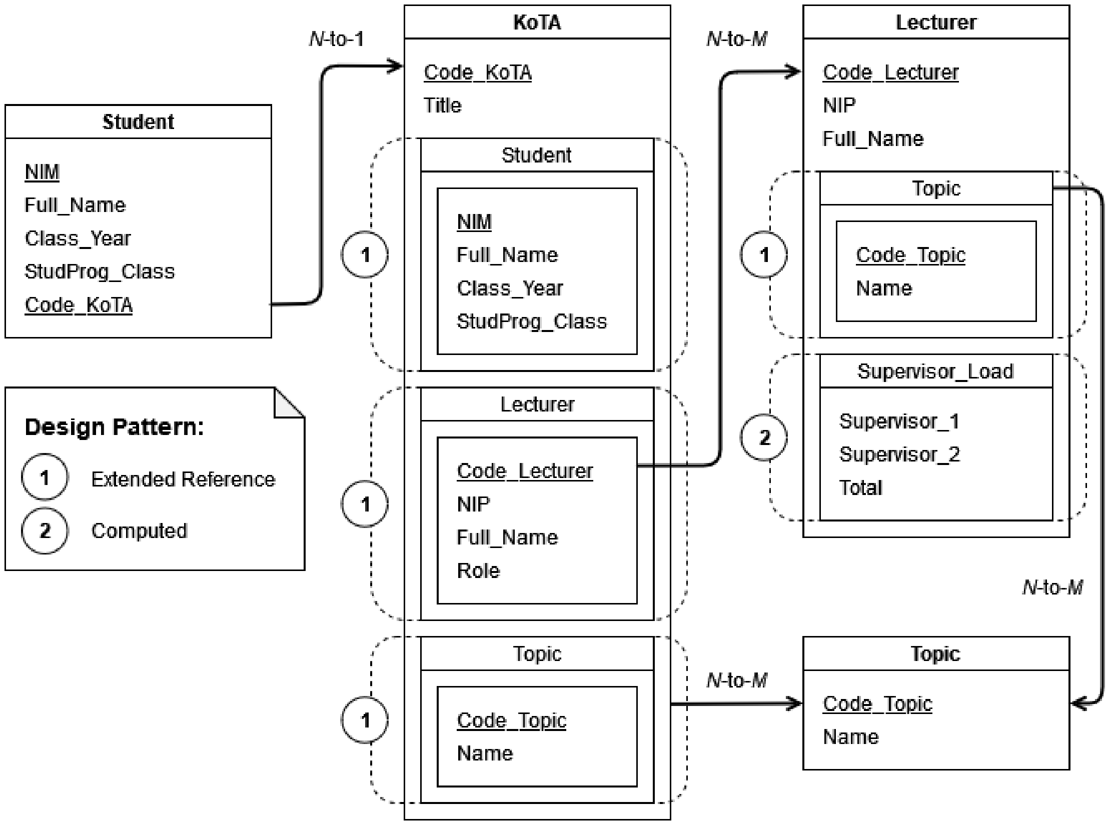

Fig. 2 Logical data model of Aplikasi Pengajuan Tugas Akhir.

It means that with read-intensive operations, the processing time can be more efficient because relationship operation was not a requirement in the joint process, and there had been no repeat operation calculation of supervisor load.

3) Physical Data Modeling: Physical data modeling that was developed using physical CRD tools (Fig. 3) only clarified logical data models based on the type and structure of data usage. It was to provide freedom of implementation on the targeted database machine. The type of data that was used was ObjectId, string, and integer. Meanwhile the data structure that was used was an object (to represent the subcollection) marked by "{ }" and an array (to accommodate composite attributes) marked by "[ ]".

B. Evaluation

1) Evaluation of the Conceptual Data Model Design: The evaluation results on the conceptual data model design showed that entities with attributes and relationships that had been created on the conceptual data model could represent actual objects matching business processes, rules, and needs. As seen in Fig. 1, the business process PB-01 (KoTA Registration) could be comprehended by the entities of KoTA, Student, Lecturer, and Topic; and by the relationships of KoTA-Student, KoTA-Lecturer, KoTA-Topic, and Lecturer-Topic. The KoTA entity was related to the Student entity to accommodate the KoTA data management. The KoTA registration process could be carried out according to the needs; the existence of a relationship with the Lecturer entity was used to determine Supervisor 1 and Supervisor 2, while the relationship with the Topic entity was used to determine the topic suitability between the KoTA and Supervisors.

The business process PB-02 (KoTA Code Assignment) could be identified from the relationship of KoTA-Student. The process of determining the KoTA Code could be done using the Code_KoTA attribute on the KoTA entity. The business process PB-03 (Supervisor Arrangement) could be determined from the relationship of KoTA-Lecturer. The process of determining Supervisor 1 and Supervisor 2 required a relationship between the KoTA entity and the Lecturer. The appointment of the supervisor considered the suitability of the Topic between KoTA and the Supervisor, thus it required the involvement of Topic entities, either between KoTA and Topics or Lecturer and Topic.

The business process PB-04 (Supervisor Load Calculation) could be understood from the relationship of Lecturer-KoTA. The calculation of the supervisor load could be accommodated using the composite attribute of Supervisor_Load which consists of the derived attributes of Supervisor_1, Supervisor_2, and Total. There was also a descriptive attribute Role in the relationship between KoTA and Lecturer as a data source to calculate the supervisor load.

2) Evaluation of Logical Data Model: The logical data model design corresponded to the conceptual data model and design pattern (Table VII), which was the derivative results of workload analysis based on quantification and qualification estimates on each operation (Table V).

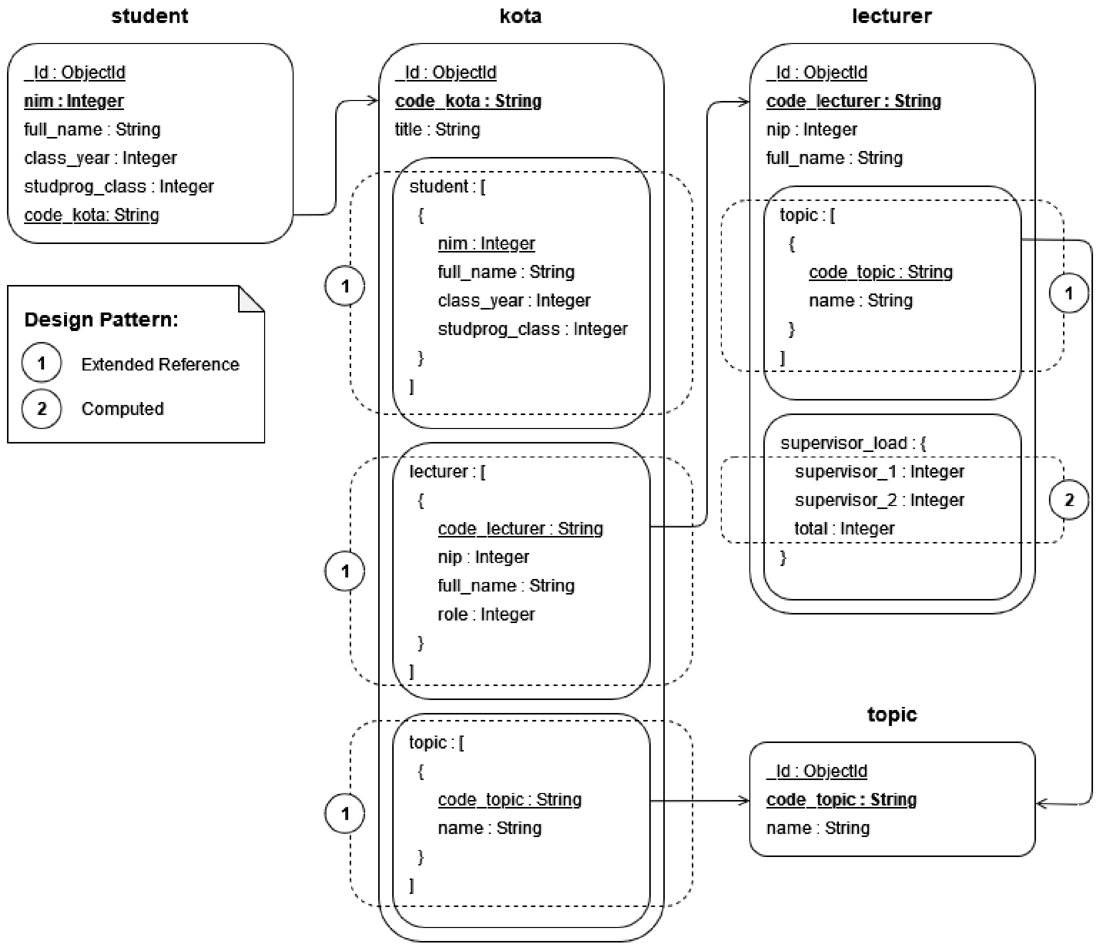

Fig. 3 Physical data model of Aplikasi Pengajuan Tugas Akhir.

The creation of the collections of Students, KoTA, Lecturers, and Topics accompanied by fields, was in line with the entities and attributes in the conceptual data model. The Student collection applied reference and embedded to KoTA. The KoTA collection applied references to Lecturer and Lecturer were embedded to KoTA. The collection of KoTA applied reference to Topic and Topic was embedded into the KoTA based on OP-07. The collection of Lecturer applied reference to Topic and Topic were embedded to Lecturer based on OP-07.

3) Evaluation of Physical Data Model Design: Physical design model design that had been analyzed showed consistency toward logical data model design that defined each collection, field, data type, and data structure. In the logical data model, applying the array of data structure becomes an integral part of creating the M-to-N relationship between KoTA-Student, Lecturer-KoTA, Topic-KoTA, and Topic-Lecturer. Furthermore, the data structure of objects can be used to accommodate derivative attributes.

The evaluation above clearly indicates that the ERD relational data model design tool with Chen notation can be used for unstructured data model on document-based NoSQL at the conceptual stage (repurposing). Likewise, CRD can be used to represent the logical stage and physical stage.

IV. CONCLUSION

The result of the study suggests that the relational data model can be applied to document-based NoSQL databases using ERD modeling tools with Chen notation on the conceptual data model, logical CRD on the logical data model, and physical CRD on the physical data model.

This study has contributed to the data model design with better read-intensive performance since it did not require a joint operation between collection with relationship (extended reference) and did not require repeat computation toward its derivative attribute (computed). Furthermore, an interesting discovery was found in the logical and physical data model designs, in which the creation of 1-to-*N* and *M*-to-*N* was able to create the same collection structure, unlike the relational data model on structured data that had been able to create different table structures due to the support of the object’s array of data structure on the collection.

CONFLICT OF INTEREST

The authors state that the representation as well as interpretation of this study is not connected to situations and conditions that may be related to personal or organizational conflict of interest.

AUTHOR CONTRIBUTION

Conceptualization, Muhammad Riza Alifi and Transmissia Semiawan; Case Study, Djoko Cahyo Utomo L.; Methodology, Muhammad Riza Alifi and Djoko Cahyo Utomo L.; Validation, Transmissia Semiawan and Hashri Hayati; Analysis and editing, Muhammad Riza Alifi, Transmissia Semiawan and Hashri Hayati; Supervision, Transmissia Semiawan.

ACKNOWLEDGMENT

This study was implemented with the competitive grant support to improve the capacity of study program in Bandung State Polytechnic, grant ID is B/75.5/PL1.R7/PG.00.03/2021.

REFERENCES

[1] V.C. Storey and I. Y. Song, “Big Data Technologies and Management: What Conceptual Modeling Can Do,” *Data and Knowl. Eng.*, Vol. 108, pp. 50–67, Mar. 2017.

[2] H.E. Samra, A.S. Li, B. Soh, and M.A. AlZain, “Review of Contemporary Database Design and Implication for Big Data,” *Int. J. Smart Educ., Urban Soc.*, Vol. 12, No. 4, pp. 1–11, Oct. 2021.

[3] K. Shin, C. Hwang, and H. Jung, “NoSQL Database Design Using UML Conceptual Data Model Based on Peter Chen’s Framework,” *Int. J. Appl. Eng. Res.*, Vol. 12, No. 5, pp. 632–636, Dec. 2017.

[4] A. Silberschatz, H.F. Korth, and S. Sudarshan, *Database System Concepts*, 7th ed., New York, AS: McGraw-Hill Book Company, 2020.

[5] IBM Cloud Education (2020) “Data Modeling,” [Online], https://www.ibm.com/cloud/learn/data-modeling, access date: 28-Sep-2021.

[6] S. Bjeladinovic, “A Fresh Approach for Hybrid SQL/NoSQL Database Design Based on Data Structuredness,” *Enterp. Inf. Syst.*, Vol. 12, No. 8–9, pp. 1202–1220, Oct. 2018.

[7] G. Hampel, “Preparing the Conceptual Model of a Database,” *QUAESTUS Multidiscip. Res. J.*, Vol. 19, pp. 290–304, Jun. 2021.

[8] J.R. Lourenço, dkk., “Choosing the Right NoSQL Database for the Job: A Quality Attribute Evaluation,” *J. Big Data*, Vol. 2, pp. 1-26, Aug. 2015.

[9] A. Schram and K.M. Anderson, “MySQL to NoSQL: Data Modeling Challenges in Supporting Scalability,” *SPLASH '12: Proc. 3rd Annu. Conf. Syst., Program., Appl.: Softw. Humanit.*, 2012, pp. 191-202.

[10] L. Schaefer (2021), “What Is NoSQL?” [Online], https://www.mongodb.com/nosql-explained, access date: 6-Sep-2021.

[11] J. Pokorný, “Functional Querying in Graph Databases,” *Vietnam J. Comput. Sci.*, Vol. 5, pp. 95–105, May 2018.

[12] A. Meier and M. Kaufmann, “NoSQL Databases,” in *SQL & NoSQL Databases*, Wiesbaden, Germany: Springer Fachmedien Wiesbaden, 2019, pp. 201–218.

[13] A. Haseeb and G. Pattun, “A Review on NoSQL: Applications and Challenges,” *Int. J. Adv. Res. Comput. Sci.*, Vol. 8, No. 1, pp. 203-207, 2017.

[14] Y. Genkina (2020) “Data Modeling with MongoDB,” [Online], https://webassets.mongodb.com/_com_assets/cms/mongodb_data_modeling_with_mongodb-44y55ekiu3.pdf, access date: 6-Sep-2021.

[15] P. Atzeni, F. Bugiotti, L. Cabibbo, and R. Torlone, “Data Modeling in the NoSQL World,” *Comput. Stand., Interfaces*, Vol. 67, pp. 1-14, Jan. 2020.

[16] J. LaBreck and D. Coupal (2020) “Advanced Schema Design Patterns,” [Online], https://webassets.mongodb.com/_com_assets/cms/mongodb_advanced_schema_design_patterns-pndclmx967.pdf, access date: 6-Sep-2021.

[17] B. Thalheim, “Conceptual Model Notions – A Matter of Controversy: Conceptual Modelling and its Lacunas,” *Enterp. Model., Inf. Syst. Archit. (EMISAJ)*, Vol. 13, pp. 9–27, Feb. 2018.

[18] T. Semiawan, “User Interface Design Analysis Pertaining to Computational Thinking Framework,” *Proc. 8th Int. Conf. Inform., Environ., Energy, and Appl.*, 2019, pp. 238–242.