Analecta Technica Szegedinensia
Vol. 16, No. 01 ISSN 2064-7964 2022

CREATING THE CONCEPTUAL AND LOGICAL MODEL OF A JOURNAL DATABASE

György Hampel

Faculty of Engineering, University of Szeged, Mars tér 7, 6724, Szeged, Hungary
e-mail: hampel.gyorgy@szte.hu

ABSTRACT

This article describes the process of creating the conceptual and logical model of a journal database. To efficiently extract the information from the articles published in the journal so far, the idea of creating a database has emerged. To create a database, it is recommended to design a high-level conceptual model and convert that into a logical data model. The benefit of the thoughtful design is that it shows the structure of the database in an easily comprehensible form. The entity-relationship model is a fast and efficient way to create the conceptual model and it can be easily converted to a relational database model, which is a logical model. The initial version of the entity-relationship model of the journal database had one entity type, 25 attributes, and no relationship. The final version contained three entity types, 39 attributes, and three relationships. This final conceptual model was converted to a logical model, the relational model. The result was ten tables to store entity data with 22 different fields and another three tables to ensure the relationships between the tables. The developed model can be created in a relational database manager and is suitable for serving information needs related to the journal.

Keywords: database planning, entity-relationship model, relational model

1. INTRODUCTION

A Hungarian language printed yearbook at the Faculty of Engineering at the University of Szeged containing peer-reviewed scientific publications was first published under the care of the Institute of Economics and Rural Development in 2006 (ISBN: 978-963-482-799-3). The book transformed into a journal in 2009 (ISSN: 1788-7593) and since then, 26 issues of the "Contemporary social and economic processes" (Jelenkori társadalmi és gazdasági folyamatok) – including the first three yearbooks – were published until the end of 2021 under the care of the Institute of Engineering Management and Economics. Since volume 14 (2019) issue 3, the journal is available online as well (ISSN: 2676-9867). Since the first publication there were 489 articles published from 906 authors in a variety of disciplines.
Seeing the unbroken development, the idea arose to create a database which could make it easier to extract relevant information from issues of the journal. Based on this idea, the task was to (step one) design the database in a high-level model, (step two) to create a logical data model based on the previous step, and (step three) to physically create the database with a relational database manager.
This article describes the first two steps: the creation of the high-level concept with the use of the entity-relationship model, and the conversion to the logical model using the relational data model.
The benefits of this thoughtful design are:

- It shows the structure of the database, the entities, and their relationships in an easy-to-understand format.
- As it is easy to understand, it facilitates the dialogue between the user and the programmer.
- It gives new ideas to both users and programmers.
- It is a plan or description that does not need to be changed by selecting or modifying the database management system.

Fig. 1 shows the process of database modelling and implementation. This article deals with the first three steps. All four steps appear in [1] and [2].

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

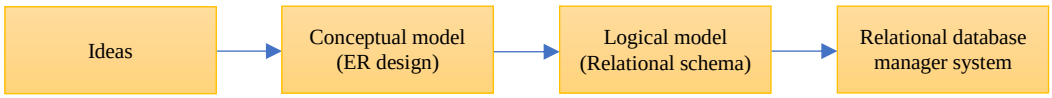

Figure 1. Database modelling and implementation process. Modified from [3].

2. MATERIALS AND METHODS

2.1. Database and data modelling

A database can be described several ways as there is no uniformly agreed definition for this concept. A few examples to define database: It consists of field-specific data, metadata describing the type of data and relationships, and a data management system [4]; It is a set of finite number of entity occurrences, their finite number of property values and relationship occurrences, organized according to a data model [5]; It is an integrated data structure that stores the occurrence data of several different objects in an organized manner according to a data model with auxiliary information called metadata, to ensure efficiency, integrity and data protection [6].
The data must be grouped to be retrievable according to some sorting principles and other aspects [5]. There are several benefits to creating and using a database, such as:

- it is a uniform, logically clear data structure,
- data can be queried in a flexible way,
- the data and the managing programs are independent of each other, so both can be modified without changing the other [7][8].

The database is always linked to a data model, in fact, it is an implemented data model [9]. On the one hand, the data model must be robust enough to grab the user's attention and meet his or her expectations, but it must also be simple and understandable for the user [7].
During the design of the database, one must first create the conceptual model, and then the logical model. Only after that should the physical implementation (the real, physical creation of the database) follow [10].
The creation of a conceptual (high-level) data model plays a significant role in the database design process, both in theory and in practice [11]. A conceptual data model is a collection of tools designed to describe reality in such a way that the created model is able to answer questions about reality [12]. This model does not address data structure or physical storage issues. Conceptual schemas include the set of entities, their characteristics, relationships, and constraints. In addition, the structure of the database is illustrated with diagrams. The database is illustrated in a way that is comprehensible even to those who are not experts in this field [4].
Of course – like everything – this can be ruined as well: improper use of modelling tools, limitations in the imagination of users and database designers, different visions and approach to the problem of users and database developers may contribute to poor design [13]. Although creating a conceptual model seems simple, it is essential that modeler has the ability to correctly identify the required objects and their relationships [12]. According to Watson [12], those not yet familiar with data modelling (e.g., university students learning database design) make these common mistakes:

- failure to recognize that something thought to be an attribute is, in fact, an entity,
- inability to generalize and assign entities to entity sets,
- failure to check relationships between entity sets from all directions, so the cardinality (number of connected entities) is incorrect,
- exceptions may be ignored.

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

Based on [3], there are some useful principles to consider when designing a database:

1. Create a realistic model: Entity sets, attributes, and relationships must reflect reality. For example, the workplace should not be the attribute of the ARTICLE, but rather the AUTHOR entity set.
2. Keep it simple: Do not add more entity sets, attributes, and relationships than necessary. For example, in the case of AUTHOR, it is probably unnecessary to include the eye colour.
3. Choose the right type of objects: There are several ways to describe reality, for example, we can describe a piece of reality with an entity set, attribute, or relationship. In general, attributes are easier to put into the model than an entity set or relationship, but solving everything with attributes is not always practical, because it can make it difficult to manage the database (see Fig. 2 for an example).
4. Avoid (or at least try to prevent) unnecessary data redundancy: Every effort should be made to include all data only once. For example, do not store the author's name as an attribute in the ARTICLE and in the AUTHOR entity set as well. This avoids anomalies that make it difficult to use and maintain the database [14][15]. At the same time, there may be cases when, for reasons of expediency, we may still be forced to store data redundantly. This may depend on the situation, the needs, and the database management system.
5. Choose the right relationships: Entity sets can be related in many ways, especially if there are many of them. It is usually not advisable to create all relationships, as this can lead to anomalies which make it difficult (or even impossible) to manage the database. It is advisable to indicate only relationships necessary for the given problem to be solved and for this we need to know in advance what we expect from the database. For example, it is advisable to establish a many-to-many relationship between the AUTHOR and the ARTICLE entity sets, as an author may write more than one article and an article may have several co-authors; or there can be only one-to-many relationship between the JOURNAL and the ARTICLE entity sets, since an article can be published (normally) only once in a journal, and a journal issue can contain several articles.

2.2. The conceptual model: Entity-relationship model

Chen laid the foundations for the Entity-Relationship model (ER model) [16] and thus had a lasting effect on data modelling. It is the most popular tool for conceptual data modelling [17]; In addition to its simplicity, its popularity is due to its theoretical validity and the fact that it is excellent for preparing the relational data model used by today's database management systems. As a result, the ER model is taught in many higher education institutions, which also contributes to its popularity and prevalence [18]. On the one hand the entity-relationship model (ER model) is expressive, simple (even for non-professionals relatively easy to understand), uses few concepts (so it can be learned quickly), works with illustrative diagrams. On the other hand, due to the representation method used, it can be difficult to describe very complex databases [19].

The elements of the entity-relationship diagrams (ERD) are:

1. An entity can be any (abstract) object or concept which can be described with a name and has attributes. Similar entities form a set (for example: JOURNAL, ARTICLE, AUTHOR) [3]. It is represented in a rectangle.
2. The attribute belongs to an entity set or relationship, it describes its features, properties [3][4] (for example: in the case of an AUTHOR: name, academic degree, job, gender). An entity set must have at least one attribute, while it is not mandatory to assign attribute to a relationship. Like in the case of entities, a noun is commonly used to name an attribute [4]. An attribute can be simple (ellipse with single line), composite (can be divided into sub attributes, denoted in ellipses connected to the composite attribute), multivalued (ellipse drawn with a double line), or derived (can be calculated from other values, ellipse with a dashed line) [9].
3. A relationship – described with a verb – connects entity sets. The most common type is the binary relationship connecting two entity sets, but the model allows connecting any number of sets [3][4].

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

According to the studies of Dey [11], the understanding of entity sets, and their attributes usually does not cause problem for the users, however, this cannot be said for the interpretation of relationships. The types can be [3][4]: recursive (relationship between two entities within the same entity set), one-to-one or 1:1 (relationship between one entity of an entity set with one and only one entity in another entity set), one-to-many (1:N) or many-to-many (N:M). The degree of relationship is determined by the connected entity sets. In addition, the participation in the relationship can be total (mandatory) when all entities form a relationship or partial (optional) when there are entities with no connection to other entities [20].

The key attribute plays an important role. It is marked by underlining the attribute [3]. A key is an attribute that is suitable to uniquely identify an entity. If there is more than one attribute suitable for the purpose, we have to select one, and this will be the primary key, while the others will be alternative keys. It is possible that an attribute cannot be identified by one attribute alone, but by combining the minimum number of required attributes, we can obtain a composite key suitable for unambiguous identification. The model also allows so called weak entities, which have no key attributes, where the entity is in a double line rectangle. (While keys are not compulsory in the ER model, they are required in the relational model. Relational database management programs allow the creation of keyless tables which is fine when creating simple databases with only one table.)

It should also be noted that several variants of the representation of ER diagrams coexist [17], which differ mainly in the representation of relationships [21] or add additional possibilities to the model's ability to describe reality, such as introducing main and subclasses, generalization, and specialization concepts [3][4], but they do not play a significant role in the current database to be implemented. According to Watson, it is not the mode of representation that matters primarily, but the ability to model correctly [12].

2.3. The logical model: relational model

Several types of logical data models have emerged since the beginning of computer database management, such as the hierarchical, mesh, object-oriented and relational data models [4] [22][23][24]. The relational model first described by Codd is the most widespread due to its simplicity and mathematical validity [25]. A data model must have a database manager based on it, so relational database managers are based on the relational data model [4].

The relational model is made up of a system of tables. A logical connection between tables can be created using fields whose contents can be found in both related tables. The structure of the tables in the relational model is as follows: The table must have a unique name that distinguishes it from other tables. The table header contains the unique column names; these column names are the identifiers of the attributes. A column is called a field (and the header is the field name). A row (or tuple) other than the header is a record. The number of columns is the degree of the table, the number of rows is called cardinality. The order of the columns or rows (except for the header) is indifferent, they can be interchanged. The attribute by which each record in the table can be clearly distinguished from another is called the primary key. If there are several suitable fields, we select one as the primary key and the others are alternative keys. If a set of several fields form a primary key, that is called composite key. The number of fields in the key must be minimal. The foreign key can be used to refer to another table that is logically related to the reference table [3][4].

There are a lot of cases when data cannot be stored and managed efficiently in a single table, because that can result in unnecessary repetitions (redundancy), which can cause various insertion, modification and deletion difficulties, problems (anomalies) [4][14][15]. To avoid anomalies, the table should be broken down into several interrelated tables using normalization rules [13][14]. Relational database managers have been devised to efficiently manage the resulting tables. Note: some spreadsheet programs – for example Microsoft Excel with Microsoft Power Pivot add-in – can handle databases and create result tables form joined tables with queries [26], but they are less effective than relational database managers.

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

Relational and other additional operations can be used to extract the desired information from the relational tables [25][27].
If conceptual modelling is done using the ER model, it is necessary to transform it into a relational model. This is because although the ER model is an easy-to-learn and easy-to-use tool, there is no database manager based on it. The main rules that can be used during the transformation are the following [3][4][6][11]:

- Entity set: Each entity set is converted to a relation, i. e. a table.
- Attribute: Simple attributes of the ER model will be the fields in the tables and the fields are named after the attribute names. In case of composite attributes, only the sub attributes are added to the table as simple attributes. Since each cell of the relational schema can contain only one elementary value, multivalued attributes are put in a separate table linked to the original table using a foreign key. Derived attributes are calculated from data from other attributes, so they are not included as fields in the tables.
- Relationship: The transformation is determined by the type of relationship. The options are as follows: (1) One-to-one relationship (1:1): The primary key of table A is embedded into table B as a foreign key (or vice versa, it does not matter). (2) One-to-many relationship (1:N): The primary key of the table in the one-side of the relationship is embedded in the many-side table as a foreign key. (3) Many-to-many relationship (N:M): A separate table provides the connection between the two tables to be connected, which contains the primary key of the linked tables. This solution can be used in any type of relations (one-to-one and one-to-many) as well.

3. RESULTS AND DISCUSSION

3.1. The description of the created conceptual model with ER diagram

After collecting and overviewing the needs, the first version of the ER model was born (Fig. 2). It contains a single entity set and its associated attributes. Since this model contains aggregated data per journal issue, it is not suitable for storing detailed data of the articles (such as the names of authors, article titles etc.).
Name of the entity set: JOURNAL. The attributes, their type, and associated metadata are listed below in the following format: *attribute name; attribute type; metadata*. For ease of reference, multi-word expressions were used to name the attributes. Remember that the attribute data refers to an entire journal issue!

1. ID; Composite attribute; Used to identify each journal issue.
2. Volume; Simple attribute, a sub attribute of ID; The volume number of the journal.
3. Issue; Simple attribute, a sub attribute of ID; The issue number within a volume number.

The Volume and Issue together form a composite key (underlined) that uniquely identifies a particular journal.

4. Year; Simple attribute; The year the journal was published.
5. Isbn Issn; Multivalued attribute; ISBN and ISSN numbers of the journal. The print and online versions have separate numbers, so it can have multiple values at the same time.
6. Publisher; Simple attribute; Name (and other details if applicable) of the responsible publisher. This attribute can even be a combined attribute (broken down to name, scientific grade, position, etc. sub-attributes) if there is a need to store these data separately for information retrieval.
7. Editor-in-Chief; Multivalued attribute; The position of editor-in-chief can be filled by several people at the same time.
8. Responsible Editor; Simple attribute; Name (and other details if required) of the editor.
9. Technical Editor; Multivalued attribute; An issue can have multiple technical editors at once.
10. Page; Simple attribute; The total number of pages in a journal issue (not just page number of the articles).

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

11. Topic; Simple attribute; Number of topics in a journal issue.
12. Article; Simple attribute; Number of scientific articles.
13. One Author; Simple attribute; Number of articles with one author.
14. Two Authors; Simple attribute; Number of articles with two co-authors.
15. Multiple Authors; Simple attribute; Number of articles with more than two co-authors.
16. Hungarian Article; Simple attribute; Number of Hungarian articles in an issue.
17. Foreign Article; Simple attribute; Number of foreign language articles.
18. Author; Simple attribute; Number of authors per journal issue.
19. Faculty Author; Simple attribute; The number of authors who are employees of the Faculty of Engineering.
20. Not Faculty Author; Simple attribute; The number of authors who are not employees of the Faculty of Engineering.
21. Degree; Simple attribute; Number of authors with an academic degree.
22. Hungarian Author; Simple attribute; Number of Hungarian authors (identified by name since citizenship data are not available).
23. Foreign Author; Simple attribute; Number of non-Hungarian authors.
24. Male; Simple attribute; Number of male authors.
25. Female; Simple attribute; Number of female authors.

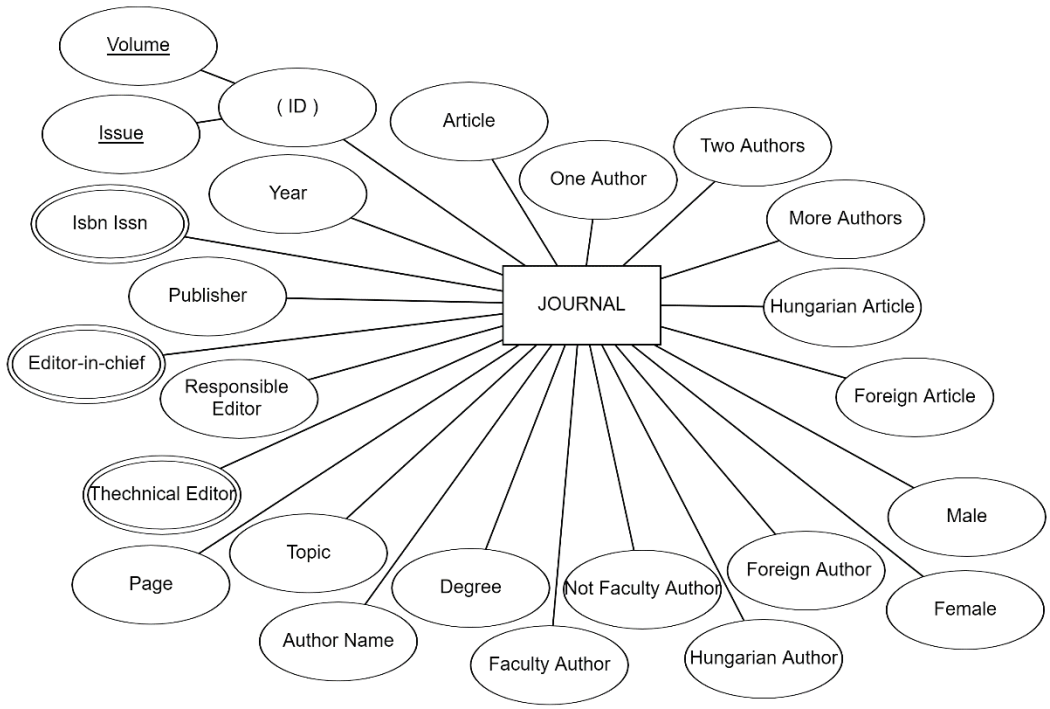

Figure 2. The ER diagram of the journal database (version 1) created with erdplus.com data modelling website.

This ER model which consists of a single entity set and twenty-five attributes is suitable to store a lot of information. But if it would be implemented this way, data entry would require a lot of preparation and preliminary manual aggregation, and the database would not contain important basic, detailed data.

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

After reviewing the first version of the ER model, a few modifications were made and eventually a final version was developed. This model contains three entity sets (JOURNAL, ARTICLE, AUTHOR) with several attributes, and three relationships (Writes, Publishes, Contains). The description of the model, the attributes, their type, and their associated metadata are listed in *attribute name; attribute type; metadata* format.

Attributes of JOURNAL:

1. Volume; Simple attribute; The volume number of the journal (a serial number).
2. Issue; Simple attribute; The issue in a volume.

The Volume and Issue together is a composite key (underlined).

3. Year; Simple attribute; The publication year of the volume issue.
4. Isbn Issn; Multivalued attribute; ISBN and ISSN numbers of the journal.
5. Responsible Publisher; Simple attribute; Name (and other details if applicable) of the responsible publisher.
6. Editor-in-Chief; Multivalued attribute; Name (and other details) of editor-in-chiefs.
7. Responsible Editor; Simple attribute; Name (and other details) of the editor.
8. Technical Editor; Multivalued attribute; Name (and other details) of the technical editors of an issue.
9. Page; Simple attribute; The number of pages of the journal.
10. Total pages; Derived attribute; The sum of pages in a journal (the sum of Page attribute).

Attributes of ARTICLE:

1. Article Title; Simple attribute and primary key; Number of scientific articles. There can be no articles with the same title in the database.
2. Topic; Simple attribute; The name of the scientific topic where the article belongs.
3. First Page; Simple attribute; The first page of the article in the journal issue.
4. Last Page; Simple attribute; The last page of the article.
5. Language; Simple attribute; The language of the article.
6. Keyword Hungarian; Multivalued attribute; Hungarian keywords of the article.
7. Keyword English; Multivalued attribute; English keywords of the article.
8. No of Sources; Simple attribute; Number of sources in the article.
9. Article Pages No; Derived attribute; The total number of pages of the article.
10. Article No; Derived attribute; Number of articles.
11. Topic No; Derived attribute; Number of topics.
12. Foreign Article No; Derived attribute; Number of articles written in foreign language (so far only English).
13. Hungarian Article No; Derived attribute; Number of Hungarian language articles.
14. N Author No; Derived attribute; Number of articles written by N number of authors.
15. Faculty Article No; Derived attribute; Number of articles written by faculty members.
16. Not Faculty Article No; Derived attribute; Number of articles written by not faculty members.

Attributes of AUTHOR:

The attribute(s) to identify each author should be carefully selected. The name alone is not enough, as different authors may have the same name, or if someone changes name between writing articles, he/she will have two different names and will therefore count as two different people. Combining the attributes of name, degree (and workplace) doesn't get us any further: if an author who publishes multiple times gets a (new) academic degree or changes workplace, he/she will again count as two (or more) different people. The problem can be handled with an arbitrarily created unique ID that can be used as a key attribute.

1. Author ID; Simple attribute and primary key; Serves as a unique identifier to distinguish authors.
2. Author Name; Simple attribute; The first name(s) and family name of the author
3. Degree; Multivalued attribute; The (scientific) degree(s) of the author when writing the article.
4. Workplace; Multivalued attribute; Details of the author's workplace(s).

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

5. Gender; Simple attribute; The gender of the author by name (since we do not ask the author's gender this data is based on the first name).
6. Author No; Derived attribute; Total number of authors.
7. Degree No; Derived attribute; Number of authors with academic degree.
8. Hungarian Author No; Derived attribute; Number of Hungarian authors (based on the author's name).
9. Foreign Author No; Derived attribute; Number of foreign (not Hungarian) authors (based on the author's name).
10. Faculty Author No; Derived attribute; Number of authors of the Faculty of Engineering.
11. Not Faculty Author No; Derived attribute; Number of authors not working at the Faculty of Engineering.
12. Female No; Derived attribute; The number of female authors (based on the author's name).
13. Male No; Derived attribute; The number of male authors (based on the author's name).

The features of the relationships:

- 1 issue can contain articles from several authors, 1 author can publish in several issues, 1 article can be published in only 1 issue.
- Each author must publish in one of the issues: an author must have at least 1 relationship with an article and can have relationships with multiple articles; an author must have at least 1 relationship with a journal issue and can have multiple ones with journal issues.
- All articles must be published in one issue: one article must have 1 and only 1 relationship with the journal; an article must have at least 1 relationship with one author and can have relationships with multiple authors.
- All journal issues must have an author: one journal issue must have at least 1 relationship with 1 author and can have relationships with more than one author.
- Each journal issue must have an article: a journal issue must have at least 1 relationship with an article.
- As a result of the above, all relationships are total. A many-to-many relationship must be created between the JOURNAL and the AUTHOR and between the AUTHOR and the ARTICLE, and a one-to-many relationship must be established between the ARTICLE and the JOURNAL.

Fig. 3 shows the ER diagram of the completed conception model.

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

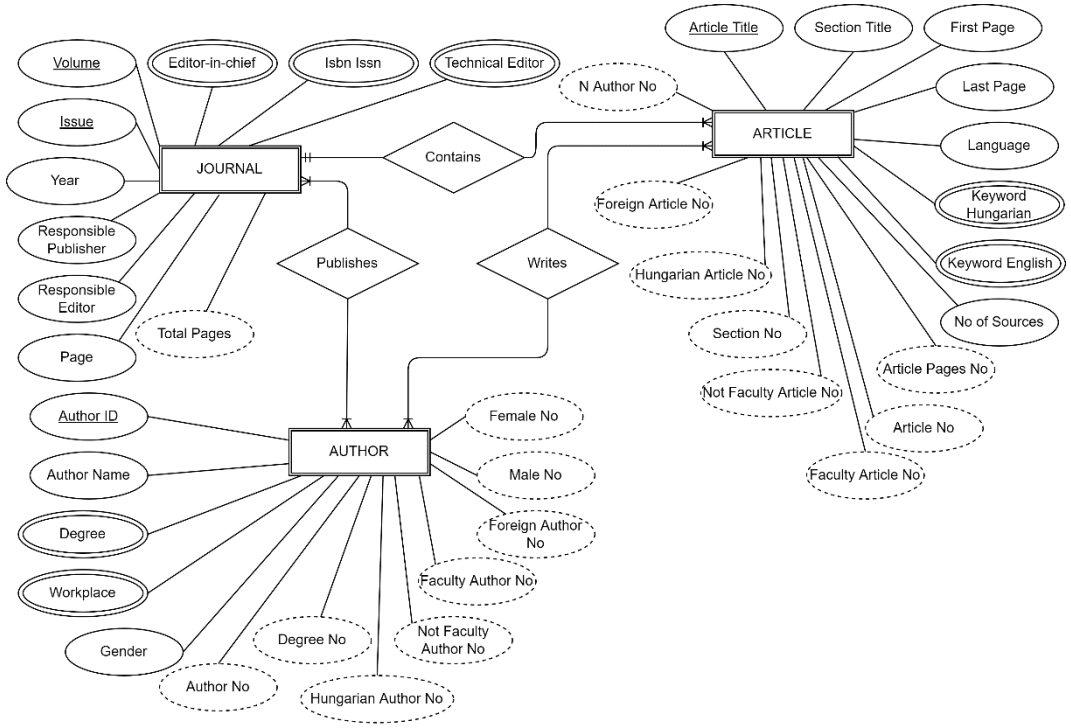

Figure 3. The ER-diagram of the journal database (final version) created with app.diagrams.net website.

3.2. The description of the logical model with relational schema

In this description of the relational schema – which is the converted ER model –, the *names of the tables* are capitalized and followed by the *field names* (attributes) in parentheses, with the *primary keys* underlined:

1. Journal entity set and its attributes: It can be divided into the following four tables:

- JOURNAL (Volume, Issue, Year, ResponsiblePublisher, ResponsibleEditor, NumberOfPages);
- EDITORINCHIEF (Volume, Issue, NameOfEditorInChief);
- ISBNISSN (Volume, Issue, IsbnIssnNumber);
- TECHNICALEDITOR (Volume, Issue, NameofTechnicalEditor);

The ER model includes a derived attribute (Total Pages) in the JOURNAL entity set which is not included in any of the tables because it can be calculated from the page data.

2. Author entity set and attributes: This set can be converted to the following tables:

- AUTHOR (AuthorID, AuthorName, Gender);
- DEGREE (AuthorID, DegreeName);
- WORKPLACE (AuthorID, WorkPlaceName);

The ER model denotes eight derived attributes in Author entity set, but since these are calculated from other attributes, they are not included in the scheme of the tables (the number of Hungarian and foreign authors, number of different degrees, males, females, etc.)

3. Article entity set and attributes: The following tables can be created in the relational model from this entity set:

- ARTICLE (ArticleTitle, TopicName, FirstPage, LastPage, Language, NumberOfSources);
- KEYWORDHUN (ArticleTitle, KeywordHungarian);

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

- KEYWORDENG (ArticleTitle, KeywordEnglish);

The ER model also contains eight derived attributes in the ARTICLE entity type that will not be included in the tables in the relational model (Number of articles with N authors, Number of foreign and Hungarian language articles, etc.).

4. Relationships: Based on the ER model, all relationships are total. There is many- to-many relationship between the JOURNAL and the AUTHOR and between the AUTHOR and the ARTICLE tables. The name of the tables providing the relationships are PUBLISHES and WRITES. A one-to-many relationship must be created between the ARTICLE and the JOURNAL; the name of the connection table is CONTAINS. The schemas of the tables are:

- PUBLISHES (Volume, Issue, AuthorID);
- WRITES (AuthorID, ArticleTitle);
- CONTAINS (ArticleTitle, Volume, Issue);

Fig. 4 created in MySQL Workbench shows the relational model of the journal database.

4. CONCLUSION

Systematic reflection on what seemed to be a simple (and quick to do) task at first (what to include as an entity set, what attributes to add, what kind of relationship is needed between the entity sets) helped to create a correct database concept.

The created relational model is suitable for the implementation, but it is advisable to check – even with test data – whether (1) it contains anomalies, (2) whether it is worthwhile to further reduce the redundancy in the data, (3) whether is it at least in the 3rd normal form. Depending on these, further refinement or modification may be necessary.

After checking the created relational model, the next phase of database design, the physical implementation can follow. This would be an SQL-based implementation of the relational schema in the selected relational database management system.

Should the need arise in the future, the implemented database can be further developed to be suitable for text mining. In this case, it can contribute even more effectively to the exploitation of the knowledge inherent in the Journal of Contemporary Social and Economic Processes.

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

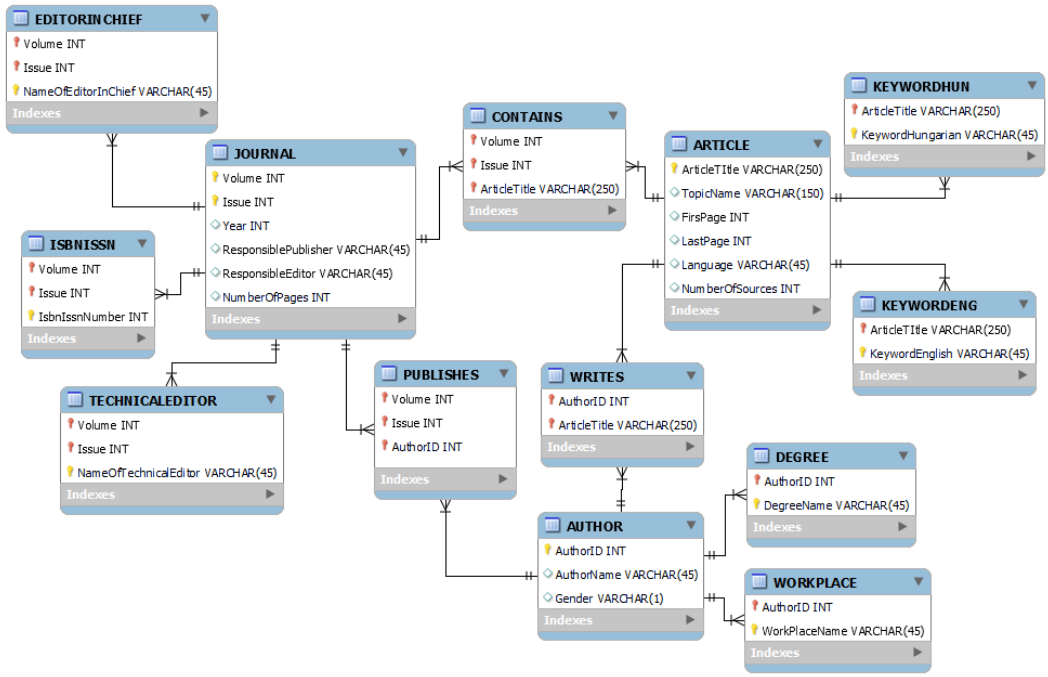

Figure 4. The relational model of the journal database created with MySQL Workbench.

REFERENCES

[1] Z. Fabulya, Access alkalmazás kialakítása ügyfélközpontú szolgáltatások nyilvántartására, Jelenkori Társadalmi és Gazdasági Folyamatok, 13 (1-2) (2018), pp. 67-76. http://acta.bibl.u-szeged.hu/55838/1/jelenkori_013_001_002_067-076.pdf

[2] Z. Fabulya, Access alkalmazás kialakítása dolgozói jelenlét nyilvántartására, Jelenkori Társadalmi és Gazdasági Folyamatok, 13 (1-2) (2018), pp. 151-160.
http://acta.bibl.u-szeged.hu/62062/1/jelenkori_013_003_004_151-160.pdf

[3] H. Garcia-Milina, J. D. Ullman, J. Widom, Database Systems: The Complete Book, 2nd Edition, Prentice Hall, Upper Saddle River, New Jersey, 2008

[4] L. Tímár, K. Vígh, J. Tátrai, J. Szigeti, Á. Vathy, É. Telekesi, I. Vass, T. Kocsis, R. Zs. Priskinné, M. Erdélyiné, Építsünk könnyen és lassan adatmodellt! Veszprémi Egyetem & Műszertechnika-Veszprém Kft., Veszprém, 19997

[5] B. Halassy, Az adatbázis-tervezés alapjai és titkai, IDG Magyarországi Lapkiadó Kft., Budapest, 1994

[6] L. Kovács, Adatbázisok tervezésének és kezelésének módszertana, Computer Books, Budapest, 2004

[7] D. M. Kroenke, C. D. Gray, Toward a Next Generation Data Modeling Facility: Neither the Entity-Relationship Model nor UML Meet the Need, Journal of Information Systems Education, 17 (1) (2006), pp. 29-38.

[8] B. L. Kacsukné, T. Kiss, Bevezetés az üzleti informatikába, Akadémiai Kiadó, Budapest, 2019 http://www.doi.org/10.1556/9789634544852

[9] Gy. Hampel, Cs. Heves, Informatika alapjai mérnököknek, alapszakos hallgatók számára, Szegedi Tudományegyetem, Szeged, 2019

DOI: https://doi.org/10.14232/analecta.2022.1.110-121

[10] V. T. N. Chau, S. Chittayasothorn, "A Bitemporal SQL Database Design Method from the Enhanced Entity-Relationship Model," 2021 7th International Conference on Engineering, Applied Sciences and Technology (ICEAST), 2021, pp. 85-90, http://www.doi.org/10.1109/ICEAST52143.2021.9426270
[11] D. Dey, V. Storey, B. Terence, Improving Database Design Through the Analysis of Relationships, ACM Transactions on Database Systems, 24 (4) (1999), pp. 453-486.
https://doi.org/10.1145/331983.331984
[12] R. T. Watson, The Essential Skills of Data Modeling, Journal of Information Systems Education, 17 (1) (2006), pp. 39-41. https://aisel.aisnet.org/jise/vol17/iss1/7/
[13] A. Badia, D. Lemire, A Call to Arms: Revisiting Database Design, Sigmod Record, 40 (3) (2011), pp. 61-69. https://doi.org/10.1145/2070736.2070750
[14] B. Halassy, Adatmodellezés. Elmélet és gyakorlat, Budapest, 2000, https://mek.oszk.hu/11100/11144
[15] R. Foorthuis, On the nature and types of anomalies: a review of deviations in data, International Journal of Data Science and Analytics, 12 (2021), pp. 297-331. https://doi.org/10.1007/s41060-021-00265-1
[16] P. P.-Sh. Chen, The Entity-Relationship Model – Toward a Unified View of Data, ACM Transactions on Database Systems, 1 (1) (1976), pp. 9-36. https://doi.org/10.1145/320434.320440
[17] S. Hartmann, Reasoning about participation constraints and Chen's constraints, Database Technologies 2003, Proceedings of the 14th Australasian Database Conference, ADC 2003, Adelaide, South Australia, February 2003, pp. 105-113.
https://www.researchgate.net/publication/221152543_Reasoning_about_participation_constraints_and_Chen%27s_constraints
[18] T. A. Carte, J. Jasperson, M. E. Cornelius, Integrating ERD and UML Concepts When Teaching Data Modeling, Journal of Information Systems Education, 17 (1) (2006), pp. 55-63.
https://aisel.aisnet.org/jise/vol17/iss1/9/
[19] Ch. L. Dunn, G. J. Gerard, S. V. Grabski, Critical Evaluation of Conceptual Data Models, International Journal of Accounting Information Systems, 6 (2) (2005), pp. 83-106.
https://doi.org/10.1016/j.accinf.2004.03.002
[20] J. Dullea, Il-Y. Song, I. Lamprou, An Analysis of Structural Validity in Entity-Relationship modeling, Data & Knowledge Engineering, 47 (2) (2003), pp. 167-205.
https://doi.org/10.1016/S0169-023X(03)00049-1
[21] C. E. H. Chua, V. C. Storey, Issues and Guidelines in Modeling Decomposition of Minimum Participation in Entity-Relationship Diagrams, Communications of the Association for Information Systems, 29 (9) (2011), pp. 159-184. https://doi.org/10.17705/1CAIS.02909
[22] Nyerges T., Logical Data Models, The Geographic Information Science & Technology Body of Knowledge (1st Quarter 2017 Edition), John P. Wilson (ed.). 2017.
https://doi.org/10.22224/gistbok/2017.1.2
[23] A. Silberschatz, H. F. Korth, S. Sudarshan, Data Models, ACM Computing Surveys, 28 (1) (1996) pp. 105-108. https://doi.org/10.1145/234313.234360
[24] B. Szabó, Adatbázis fejlesztés és üzemeltetés I., Eszterházy Károly Főiskola, Eger, 2013
[25] E. F. Codd, Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks, Research Report RJ599, IBM, San Jose, California, 1969
[26] M. Russo, A. Ferrari, Analyzing Data with Power BI and Power Pivot for Excel, Microsoft Press, Redmond, Washington, 2017
[27] A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts, 7th editon, McGraw-Hill Education, New York, 2020

DOI: https://doi.org/10.14232/analecta.2022.1.110-121