ISSN No. 0976-5697

Volume 8, No. 5, May – June 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Relational Database Normalization under Tabular Approach: A Design Methodology

Kunal Kumar
Research Scholar
University Department of Statistics and Computer Applications
T. M. Bhagalpur University, Bhagalpur-812007, India

Dr. Sachindra Kumar Azad*
Associate Professor and Head
University Department of Statistics and Computer Applications
T. M. Bhagalpur University, Bhagalpur-812007, India

Abstract: Relational Database Design and normalization, is a collection of processes that ensure the representing, development, interpreted and maintenance of enterprise records and data management systems. It also helps to produce effective database management systems. That meets the requirements of the end users and has high performance. Normalization approach uses to reduce data redundancy, inconsistency and maintain the atomicity within a database relation. In this paper, A Tabular approach process is proposed to determining key(s) from set a given set of functional dependencies. The moment, it determine the key(s) of a relation from a set of functional dependency. It shows an automatic generation of all possible candidate key(s) and super key(s) of a relation. In the proposed method, a table has three columns and one row. This column and row of the table contains the given set of functional dependency. Once, putting all possible valid set of functional dependency into the tabular form, then using tabular method approach mechanisms applying to determine key(s) of a relation. Key of a relation is used to help to reach higher level of relational database design.

Keywords: Normalization, Tabular Approach, Relational Database, Functional Dependencies, Candidate key

I. INTRODUCTION

Relational Database Design has been a most popular research filed for many years because of its different applications prospective [1]. Some of its paramount potential application fields are University management system, Airlines, Data entry, etc. Normalization is, in database design the way of organizing data to minimize redundancy in relation. It usually involves decomposing a large database into one or more tables and also identifying relationships between the relations. The objective is to minimize upgrading problems that could arise after modification of a relation attribute. The developer of the relational model also proposed the concepts of normalization and Normal Forms (NF) [2]. In general, normalization requires additional relation and some designers find this first difficult. Violating one of the first three rules of normalization, make the application anticipates any problems that may could occur while designing database, such as redundant data and inconsistent data dependencies.

When using the simple definitions of the second and third normal forms (2NF and 3NF for short), it must be aware of partial and transitive dependencies on all candidate key(s) and not just the primary key(s) [3] of the relation. Database Normalization is the part of the logical database design. The main goal of normalization is to eliminate redundancy and potential modify anomalies. Redundancy means that the group of data is saved, more than once in a database relation. Update anomaly is a consequence of redundancy. If a piece of record is stored in more than one relation, the same record must be updated in more than one place. Normalization is an approach by which one can update the relation database schema to reduce the data redundancy [4], [5]. Each normalization stage adds more relations or tables into the database. Relational Databases change time to time as data is inserted and removed. The collection of data stored in the database table at a particular moment is called an instance of the database. The overall design of the relational database is called the database schema. Schemas are update in frequently way, if at all. The main concept of relational database schemas and instances can be understood by analogy to a program written in database languages [6]. A relational database schema corresponds to the variable declarations with data type in a program. Each variable has a particular literal value at a given instant. The values of the variables in a database program at a particular point in time correspond to an instance of a relational database schema [7]. Relational Database systems have several schemas, partitioned as per their levels of abstraction. The physical schema describes the relational database design at the physical level, while the logical database schema describes the database design at the logical level.

A relational database may also have many schemas at the view level, sometimes called sub schemas that show different views of the database. Of these, the logical database schema is by far the most important part, in terms of its effect on application oriented programs, since designer construct many applications by using the logical database schema [8], [9]. The physical database schema is hidden beneath the logical database schema, and can usually be changed easily without affecting any application programs [10]. Application programs are said to be exhibit physical data independence, if they are not depend on the physical database schema, and thus need not be rewritten if the physical database schema changes.

II. FUNCTIONAL DEPENDENCIES

Functional dependency and its family is a relationship that exists, when one or more attribute(s) uniquely identify another attribute(s). If R is a relation with attributes A

and B , a functional dependency relation between the attributes is represented as $A \to B$, which indicates Y is functionally dependent on attribute A . Here A is a determinant set and B is a dependent attribute [11]. Each value of attribute A is associated precisely with one attribute B value. Functional dependency in a database relation serves as a constraint between two sets of attribute(s). Defining functional dependency is an important part of relational database design and contributes to aspect normalization. A functional dependency is trivial functional dependency, if B is a subset of A [12]. In a relation with attribute(s) of students name and Aadhar_Number, students name is functionally dependent on Aadhar_Number because the Aadhar_Number is unique for individual students. An Aadhar_Number identifies the students specifically, but a student name cannot distinguish the Aadhar_Number because more than one student could have the same name. Functional dependency also defines BCNF normal form and 3NF normal form. This preserves dependency between attribute(s), eliminating the duplicate of data. Functional dependency is related to a candidate key(s), which uniquely identifies a row and determines the value of all other attribute(s) in the relation [13]. Functional dependency is constraints on well-formed relations and shows formalism on the structure of relation.

Definition I:

A functional dependency on a relation database schema R is a constraint $A \to B$, where A and B are subsets of attributes of R.

Definition II:

A Functional dependency is a relationship between an attribute " B " and a determinant (one or more other attributes) " A " such that for a given value of a determinant the value of the attribute is uniquely defined.

A Is a determinant of R

A Determines B

B Is functionally dependent on A

$A \to B$

$A \to B$ Is trivial if $B \subseteq A$

Definition III:

A Functional dependency $A \to B$ is satisfied in an instance r of R if for every pair of tuple(s), t and s: if t and s agree on all attributes in A then they must agree on all attributes in B

III. TABULAR MEHTOD

In order to understand the tabular method of determining key(s).In Tabular method approach has Table, and can be design using following rules.

a. A three columns and one row table can be designed. First row of the table is named as *left (L)*, second row of the table named as *middle (M)* and third row of the table is named as *right(R)*.

b. The attribute(s), which is left hand side of the arrow of Functional dependency, are inserted into the left column of the table.

c. The attribute(s), which is both side of the arrow of Functional dependency left as well as right hand side, are inserted in to the middle column of the table.

d. The attribute(s), which is right hand side of Functional dependency the arrow, are inserted in to the right column of the table.

e. The attribute(s) which inserted into right column of the table, will never be a part of the any key(s) attribute i.e. always seems as non-prime attribute.

After applying the above five rules, start finding closure of left hand column of the table, if we are unable to find key from left closure then, we take Cartesian of left and middle.

Consider a relation $R(A,B,C,D)$, FD's = { $A \to BC$ and $C \to D$ } ,Find Key of a Relation.

Table I is the Tabular representation of the dependencies

Table I. Tabular Approach representation of the dependencies

L	***M***	***R***
A	C	B,D

Now, we know the dependency is inserted into tabular format. The algorithm says that attribute(s) which reside(s) right column of the table will never be the part key(s).Now, we start finding closure of left hand side attribute(s).

So, the attribute(s) reside at left column is A

Closure set of $\{A\}^0 = \{A\}$

Closure set of $\{A\}^1 = \{A,B,C\}$

Closure set of $\{A\}^2 = \{A,B,C,D\}$

Closure set of $\{A\}^3 = \{A,B,C,D\}$

Closure set of $\{A\}^2 = \{A\}^3$

Hence, Closure of $\{A\}^+ = \{A,B,C,D\}$

Candidate key(s) of the relation R is A .No need to find out the closure of middle column attribute(s) i.e. C , because left hand side attribute A is able to find all the attribute(s) of the relation. If left hand side attribute not able to find all attribute(s) of the relation then, we consider the middle column attribute(s) to determine key, combined with left hand side attribute(s). So, a prime attribute(s) of the relation i.e. A is part of the key element. B, C and D is not a part of key, it is referred to as non-prime attribute(s).

IV. NORMALIZATION METHODOLOGY

It is assumed that the relational database table is in first normal form. Where the table is free from redundancy and there is no multi-valued attributes in a table. Relational Database tools are used for the simulation work.

A. THE SECOND NORMAL FORM

A table is in second normal form or 2NF if it is in 1NF and there is no non-key attribute is partially dependent on any candidate key(s) of the table. In other words, no $A \to B$, where A a strict subset of a candidate is key(s) and B is a non-key attribute. Simply, a table is in 2NF iff it is in 1NF and every non-key attribute of the relation is either fully dependent on the whole of any candidate key(s), or on another non-key attribute [14].

Theorem: Let us supposed the table R is in 1NF and having functional dependency F_d .We have the following results:

- **a.** If key is single attribute then R is automatically in 2NF.
- **b.** If there is partial dependency i.e. not fully dependent then R is not in 2NF.

Proof.

- **a.** If $F_d = NULL$ i.e., there is no partial dependency on table R that means also that the first if-condition of the algorithm will not satisfied..
- **b.** If $F_d \neq NULL$, i.e., the first if-condition of the algorithm holds at least once. Therefore, R is not in 2NF.

Let, that our relation is in $1NF$, and the resulting first normal form relation is: $R(A,B,C,D,E)$, FD's = { $AB \to C$, $B \to D$ and $D \to E$ },Table II is the Tabular representation of the dependencies

Table II. Tabular representation of the dependencies

L	M	R
A,B	D	C,E

After putting the FD's into tabular form, it found that AB is the key of the relation R and the relation is not in 2NF .In 2NF every non-prime attribute must be dependent on candidate key(s), but not part of the key. Here, the dependency $B \to D$ shows partial dependency which shows it is not in 2NF. To form this schema R to in 2NF, It have to decompose the table into $R_1(A,B,C)$ and $R_2(B,D,E)$. Now, R_1 and R_2 are in 2NF and the decomposition is lossless, where functional dependency preserve holds.

B. THE THIRD NORMAL FORM

A relation is said to be in third normal form if it is in 2NF and none of its non-prime attributes are transitively dependent upon any candidate key or non-prime attribute. An alternative definition is a relation is in 3NF if in every non-trivial dependency $A \to B$ either A is a super key or B is a key attribute (i.e., B is a prime attribute) [15]. The transitive dependencies set on 2NF relations are defined by:

$$F_d = \{A \to B \in F_c\}$$

- **a.** A is not a candidate key, and
- **b.** B Is not a prime attribute.

Theorem: A relation schema R is in third normal form iff for every key S of R and Every non key-Attribute A depends directly on S not part of S.

Let, that above relation is in $2NF$, and the resulting first normal form relation is $R(A,B,C,D,E)$, FD's = { $AB \to C$, $B \to D$ and $D \to E$ }, Consider the Table 3, it shows that the AB is the candidate key of the relation and $R_1(A,B,C)$ and $R_2(B,D,E)$ are in second normal form but not in third normal form. The reason which violet the third normal form is in table $R_2(B,D,E)$ the dependency $D \to E$ violets , because D is a non prime attribute which is not a part of the key determine the attribute E, is illegal in 3NF[16]. Now, to convert this table to 3NF it is compulsory to decompose the table $R_2(B,D,E)$ into one more different table R_3 . Now, $R_2(B,D)$ and $R_3(D,E)$, it is in 3NF but not in BCNF. Now, $R_1(A,B,C)$, $R_2(B,D)$ and $R_3(D,E)$ are in 3NF with lossless decomposition and functional dependency preserve holds.

C. THE BOYCE-CODD NORMAL FORM FORM

One of the more strict normal forms that can obtain is Boyce–Codd normal form (BCNF) . It eliminates all data redundancy and data inconsistency that can be derived based on family functional dependencies, there may be other types of redundancy remaining in the relation. Relation database schema R is in BCNF with respect to a set of all possible functional dependencies F if, for all functional dependencies in F^+ of the form $A \to B$, where $A \subseteq R$ and $B \subseteq R$, at least of these holds [17].

- **a.** $A \to B$ is a trivial functional dependency i.e. $B \subseteq A$
- **b.** A is a super key of relational schema R

Let, that our relation is in $1NF$, and the resulting first normal form relation is $R(A,B,C,D,E,,F,G,H,I,J)$, FD's = { $AB \to C$, $B \to D$, $D \to EF$, $A \to GH$ and $H \to IJ$ }

Table III is the Tabular representation of the dependencies

Table III. Tabular representation of the dependencies

L	M	R
A.B	D,H	C,E,F,G,I,J

After feeding the FD's into tabular form, it found that AB is the key of the relation R and the relation is not in 2NF because, $B \to D$ and $A \to GH$ shows the partial dependency (PD). So, to eliminate this PD's, it is compulsory to create two different set of tables $R_1(A,G,H,I,J)$ and $R_2(B,D,E,F)$, $R_3(A,B,C)$ .Now, these three tables are in 2NF but neither in 3NF nor in BCNF. So, to make these tables to 3NF or BCNF to have to decompose the table because, the dependency $H \to IJ$, and show that non-prime attribute determines the non prime attribute. Decompose $R_1(A,G,H,I,J)$ into $R_{11}(A,G,H)$ and $R_{12}(H,I,J)$ and also decompose $R_2(B,D,E,F)$ into $R_{21}(B,D)$ and $R_{22}(D,E,F)$ .Now,

$R_3(A,B,C)$, $R_{11}(A,G,H)$, $R_{12}(H,I,J)$, $R_{21}(B,D)$ .Are 3NF and as well as BCNF.

V. A COMPREHENSIVE ANALYSIS

To express the applicability of the all the above algorithm. Consider the table given which is un normalized table. This table is in ONF i.e. un-normalized table. The motto of this above algorithm is to normalize the given table up to BCNF level. The table contains four attributes and two functional dependencies. Let us consider a relation $R(roll, name, subject\_code, subject)$ and functional dependencies $FD = \{roll \to name, subject\_code \to Subject\}$ .Applying tabular approach to find out the key of the table R.

Table IV: Tabular representation of the dependencies

L	M	R
$roll, subject\_code$		$name, subject$

After feeding the FD's into tabular form, it found that $roll, subject\_code$ is the candidate key of the relation R and the relation is not in 2NF.

Table V: Student table

ROLL	NAME	SUBJECT_CODE	SUBJECT
11	A	MCA1001	C
12	B	MCA1002	JAVA
13	C	MCA1003	DBMS
14	D	MCA1004	HTML

The dependencies $roll \to name, subject\_code \to Subject$ have partial dependency. So, the relation is not in 2NF. To convert this table to 2NF it is compulsory to decompose into two different tables $R_1(roll, name)$ and $R_2(roll, subject\_code, subject)$.

Table VI: R_1 is in 2NF

ROll	NAME
11	A
12	B
13	C
14	D

Table VII. R_2 is not in 2NF

Roll	SUBJECT_CODE	SUBJECT
11	MCA1001	C
12	MCA1002	JAVA
13	MCA1003	DBMS
14	MCA1004	HTML

R_2 is not in second normal form because; FD's $subject\_code \to Subject$ shows the partial dependency in table R_2. Decompose R_2 into two different tables $R_2(roll, subject\_code)$ and $R_3(subject\_code, subject)$. After decomposition of table into three different tables, the table contents satisfy 2NF.

Table VIII. R_2 is in 2NF

Roll	SUBJECT_CODE
11	MCA1001
12	MCA1002
13	MCA1003
14	MCA1004

Table IX. R_3 is in 2NF

SUBJECT_CODE	SUBJECT
MCA1001	C
MCA1002	JAVA
MCA1003	DBMS
MCA1004	HTML

The table R is divided into three different tables R_1, R_2 and R_3 are in 2NF. If a table is having only two attributes then, the table is in BCNF automatically.

Table X. R_1 is in BCNF

ROll	NAME
11	A
12	B
13	C
14	D

Table XI. R_2 is in 2NF

Roll	SUBJECT_CODE
11	MCA1001
12	MCA1002
13	MCA1003
14	MCA1004

Table XII. R_3 is in BCNF

SUBJECT_CODE	SUBJECT
MCA1001	C
MCA1002	JAVA
MCA1003	DBMS
MCA1004	HTML

If a table is in BCNF, it also satisfies the requirement of 3NF.So, it can assume that the above three tables are in 3NF and the of decomposition is lossless decomposition where functional dependencies preserve holds.

VI. DISCUSSION AND CONCLUSION

This content of this paper shows a new technique of key generation. It is based on the tabular approach method. Earlier it usages the closure based key generation technique. One best thing about the relational table, it eliminate the non-prime attribute which is not a part of key in the table designing phase, where an attribute(s) which are inserted right hand side of the tabular format will not be the part of key. Tabular approach to database design is the process of generating a detailed data model of a database. This data model selects all the needed logical and physical design choices and physical storage arguments needed to design in a data definition language, which can then be used to design a database. A fully attributed database data model contains detailed attributes for each record. The effectiveness of the proposed approach is saving time and reducing mind work.

VII. ACKNOWLEDGEMENT

The authors would like to thank the referees for their valuable comments and improvements to this paper and also thankful to University Department of Statistics and Computer Applications, T. M. Bhagalpur University, Bhagalpur for providing the basic requirement for the preparation of the paper.

VIII. REFERENCES

1. Salvador Lucas, Jose Meseguera ,Normal forms and normal theories in Conditional rewriting , Elsevier Journal of Logical and Algebraic Methods in Programming, 85, 67–97, 2016.

2. Zichen Xu, Yi-Cheng Tu, and Xiaorui Wang, Online Energy Estimation of Relational Operations in Database Systems, IEEE transactions on computers, vol. 64, no. 11, November 2015

3. Carlos Busso, Soroosh Mariooryad, Angeliki Metallinou and Shrikanth Narayanan, Iterative Feature Normalization Scheme for Automatic Emotion Detection from Speech, IEEE transactions on affective computing, vol. 4, no. 4, October-December 2013

4. MOUSSA DEMBA,” Algorithm for relational database Normalization up to 3NF” International Journal of Database Management Systems (IJDMS) Vol.5, No.3, June 2013

5. Vimala, S., Khanna Nehemiah, H., Saranya, G. and Kannan, A.“Analysis and Modeling of Multivalued Attributes in Entity Relationship Modeling: An Approach for Improved Database Design, CRL Publishing - Computer Systems Science and Engineering,Vol. 28, No. 4, 2013

6. Vimala, S., Khanna Nehemiah, H., Saranya, G. and Kannan, A.“Applying Game Theory to Restructure PL/SQL Code”, International Journal of Soft Computing, Vol.7, No.6, pp.264-270, 2012.

7. Ashish Kamra and Elisha Bertino,”Design and Implementation of an Intrusion Response System for Relational Databases”, IEEE transactions on knowledge and data engineering, vol. 23, no. 6, june 2011

8. Tauqeer hussain, shafay shamail, mian m. Awais,” Eliminating process of normalization in relational database design”, IEEE transactions on software engineering, vol. 31, no. 2, February 2005

9. S.Chaudhuri, Z. Chen, K. Shim and Yuqing Wu,” Storing XML (with XSD) in SQL Databases: Interplay of Logical and Physical Designs”,IEEE transactions on knowledge and data engineering, vol. 17, no. 12, Dec-2005

10. Keng Siau and Xin Tan,” Cognitive Mapping Techniques for User– Database Interaction”, IEEE transactions on professional communication, vol. 49, no. 2, june 2006

11. Levein, R. E. and Maron, M. E. “A Computer System for Inference Execution and Data Retrieval”, Communication of ACM, Vol.10, No.11,pp.715-721, 1967.

12. Mens, T. and Tourwe, T. “A Survey of Software Refactoring”, IEEE Transactions on Software Engineering, Vol.30, No.2, pp.126-139, 2004.

13. Normann, R. “Minimal Lossless Decompositions and some Normal Forms between 4NF and PJ/NF”, Information Systems, Vol.23, No.7, pp.509-516, 1998.

14. Coleman, D.M., Ash, D. B., Lowther and Oman, P.W. “Using Metrics to Evaluate Software System Maintainability”, Journal of Computer, Vol.27, No.8, pp.44-49, 1994.

15. Chen, P. P. “ The Entity-Relationship Model - Toward a Unified View of Data”, ACM Transactions on Database Systems, Vol.1, No.1, pp. 9-36, 1976.

16. An, Y., Hu, X. and Song. Y. “Maintaining Mappings between Conceptual Models and Relational Schemas”, Journal of Database Management, Vol.21, No.3, pp.36-68, 2010.

17. Jones, T.H. and Song, I.Y. “Binary Equivalents of Ternary Relationships in E-R Modeling: A Logical Decomposition Approach,” Journal of Database Management, Vol.11, No.2, pp.12-19, 2000.

Mr. Kunal Kumar received the M.C.A and B.C.A degree from the Birla Institute of Technology, Mesra, Ranchi, India in 2012 and 2008. He is currently working as a Research Scholar toward the PhD degree at the University Department of Statistics and Computer Applications. Tilka Manjhi Bhagalpur University, Bhagalpur, India. His current research interests areas include database design and database theory.

Dr. Sachindra Kumar Azad is an Associate Professor and Head in the University Department of Statistics and Computer Applications at Tilka Manjhi Bhagalpur University, Bhagalpur, India.