ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

Crypt Analysis of Des Cryptosystem over Man in the Middle Attack

[1] Dr.R.Viswanathan, [2] Chandrasegar.T, [3] S.Aravinthkumar
[1] Associate Professor (GalgotiasUniversity), [2] Assistant Professor ((SE), VIT Uinversity), [3] Assistant Professor (GalgotiasUniversity).

Abstract: Nowadays, there is lot of advancement taking place in the database system w.r.t consistency, redundancy, dependency, atomicity, data isolation etc. Various stages of normalization (1st, 2nd, 3rd data structure) and use of Relational database technology are thriving throughout the data processing industry. RDBMS systems are valued for their ability to decrease unnecessary data redundancy, preserve the integrity of data, and deliver maximum flexibility in retrieval. Well-structured RDBMS applications normally result in normalized tables that duplicated data, creates appropriate key to data associations within a table and eliminate repeating data groups. Most of the industries adheres the cod's rules to obtain standards to their environment. It is observed that third normal form is sufficient form for a small & large company sectors to maintain the database. In this paper, the proposed algorithm objective is to implement an automated tool using dependency matrix, directed graph matrix and inference axioms. It then continues with producing 2NF, 3NF. Tables are created as the procedure proceeds. One more side product of this research is to automatically differentiate one primary key for every final table which is generated.

Keywords: — Relational Database , Automatic Normalization, Primary Key and Functional Dependency.

I. INTRODUCTION

Database normalization is the process of converting data into well-formed or natural groupings which is stored in one place [1, 11]. The aim of normalization is to generate a set of relational tables with least amount of redundant data that can be consistently and correctly modified. The main aim of normalization technique is to design a database that eludes update anomalies and redundant information [2]. E.F Codd first proposed the process of normalization. Normalization process is a sequence of tests on a relation to determine whether it satisfies or violates the requirements of a given normal form. E.F Codd initially proposed three normal forms called first (1NF) second (2NF), third (3NF) normal form. However later on R.Boyce and E.F Codd made an amendment to 3NF, the trend of defining other normal forms continued upto eight normal form. But in our paper we are proposing normalization upto 3NF as it is an adequate form of normalization for small to large companies to maintain database. Summary of Normal Forms based on Corresponding Normalization and Primary Keys which is shown in Table 1. [3]

Normal Form	Test	Remedy(Normalization)
First (1NF)	Relation should have no non-atomic attributes or nested relations.	Create new relations for non-atomic attribute or nested relation.
Second (2NF)	For relations where primary key comprises multiple attributes, no nonkey attribute would be functional dependent on a part of the primary key.	Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it.
Third (3NF)	Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key.	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).

Table 1. Summary of Normal Forms Based on Primary Keys and Corresponding Normalization.

Normalization process proceeds in a top down fashion by testing and evaluating each relation against the criteria for normal forms and decomposing relations as necessary. If the database qualifies 1NF only then it can be normalized to 2NF and so on. This is clearly shown in the given flowchartFig1.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

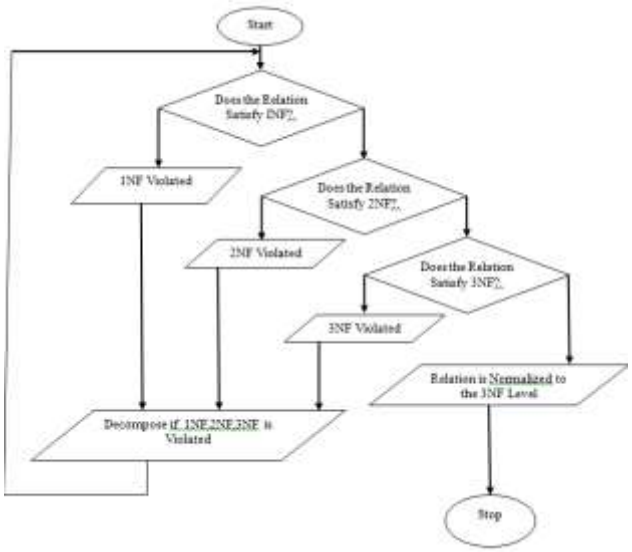

Figure 1. Flowchart.

All the normal forms except 1NF depends totally upon Functional Dependencies (FD) among the attributes of a relation. The functional dependency is a constraint between two sets of attributes in a database's relation. Given with a relation R and a set of attributes X, also in R,(written as X □ Y) , if and only if X value is associated with precisely one Y value, then ,Y is said to be dependent attribute and X is said to be determinant set. If X, Y and Z are sets of attributes in a given relation R, several properties of functional dependencies can be derived. Armstrong's axioms are the most important ones which are:

I. Axiom of Reflexivity:
If Y is a subset of X, then {X → Y}

II. Axiom of Augmentation:
If {X → Y}, then {XZ → YZ}

III. Axiom of Transitivity:
If {X →Y and Y → Z}, then {X → Z}

IV. Axiom of decomposition, or projection:
If {X → YZ}, then{X → Y} and {X → Z}

V. Axiom of pseudo transitivity:
If {X→Y, WY→Z, then {WX → Z}

With the repeated application of these axioms, all Functional Dependencies can be generated. These Functional Dependencies are the bases for database normalization. Normalization is the major task in the design of relational databases [4]. Normalization process saves time as well as money. Many approaches have been introduced since then. Various algorithms were introduced by the time. Despite of its importance, a very few algorithms were taken to design commercial automatic normalization tools. Mathematical normalization algorithm is implemented in [5]. In [6] elaborate the UML meta-mode by set of stereotypes and tagged values. Then, convert data model from one normal form to another one by using a graph rewrite rule. Later on Amir Hassan Bahmani came up with the automatic database normalization technique, which use dependency graph diagrams to represent functional dependencies of a database [7].

II. RELATED WORKS

Sungchul Lee et al. [9] implemented an architecture that can competently gather the information from various sensors, store them in a database, and offer a user interface for data retrieval. Arduino-based sensors are used due to their cost-effectiveness and flexibility. Data visualization can be done by Google Charts service and Restful Web Service is used for communication with the Arduino-based sensors.

Kunal Kumar et al. [10] introduced the Tabular approach algorithm method to produce candidate key from set a valid set of functional dependency. Once, it determines the candidate key of a database table from a given valid set of functional dependency, then applying normalization algorithms. Order-sorted rewrite theories includes Conditional term rewriting, which provides types, subtypes and rewriting modulo axioms, and encompasses the more restricted framework of conditional term rewriting systems (CTRSs) presented in S. Lucas et al. [12].

R. Vangipuram et al. [13] developed a web based Normalization tool which can handle 30 redundant attributes in the Functional dependency and more than 50 complex functional dependencies. Moussa Demba [14] describes an automatic approach for database normalization up to third normal form including all candidate keys, primary key. Ivan Ubaleht [15] proposed the set of inference rules, algorithm of computing of the closure of a set of attributes and algorithm to test the membership in the closure of the elementary relationships of attributes.

A common perception is that Armstrong relations are useful in the acquisition of data semantics. W.-D. Langeveldt et al. [16] reports on empirical evidence for this perception regarding the class of functional

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

dependencies. Investigated the usefulness of Armstrong relations with respect to various measures for this purpose. M Arenas et al. [1] provided a set of tools for testing when a condition on a database design, specified by a normal form, corresponds to a good design. They used techniques of information theory, and define a measure of information content of elements in a database with respect to a set of constraints.

Yazici et al. [17] proposed a Java user interface called JMath-Norm was designed to execute the Mathematica modules in a systematic way. Mathematica's Java link facility (JLink) is utilized to drive the Mathematica kernel for this purpose. JMath-Norm provides an effective interactive tool in an educational setting for teaching DB normalization theory. Du H et al. [18] explore a prototype system for normalization which includes 2NF, 3NF and BCNF normalization. They developed an algorithm for all normalization. Employ efficient data structures on functional dependencies and relation schemes to improve the performance of these algorithms.

Akehurst, D.H. et al. [19] provide a tool supporting the normalisation of database system designs can subsequently be developed providing an invaluable aid to the software system designer.

III. REPRESENTING THE DEPENDENCIES

We observe all relations between different attributes of a table using functional dependency. Graphically the dependencies can be represented by using a set of symbols. Simple keys (attributes) and composite keys (keys composed of more than one attribute) have been separated by (dotted) horizontal line.

A. Dependency Graph

Following are the rules to be followed in Dependency Graph.
And

a. At bottom of graph we draw each attribute and encircle it.
b. A horizontal dotted line is drawn on top of all attributes.
c. Each composite key are drawn on top of the horizontal line and they are encircled.
d. Arrows are drawn for all functional dependency.
e. Reflexivity rule dependencies are drawn (eg.ST-□S, ST-□T), using dotted arrows.

Considering the following case:

Relation ST {M, N, O, P, Q, R, S, T, U, V W, X}, with dependencies: FDs = {M→NO, Q→MP, S→MQVW, ST→RU, W→MX, and V→W } which is shown in Figure 2.

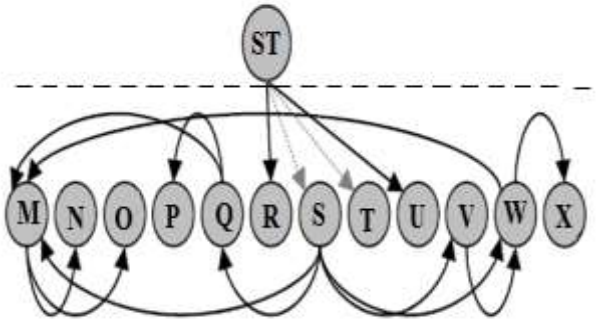

Figure 2. Graphical representation of dependencies.

B. Dependency Matrix (DM):

As we have obtained all dependencies between determinant keys thus we can create all dependencies between all the attributes of a relation. These dependencies are denoted by using a Dependency Matrix (DM) which is as follows:

1. DM [n] [m] is a matrix where
   n=number of determinant keys
   m=number of simple keys
2. Suppose that b a, c a and
   b, c {Simple key set}
   a {Determinant key set}
3. Establish DM elements as follows:
   If a → b => DM[a] [b] =2
   If a → c => DM[a] [c] =1
   Otherwise => DM[a] [c] =0

The DM matrix for above example is indicated in Figure 3.

	M	N	O	P	Q	R	S	T	U	V	W	X
M	2	1	1	0	0	0	0	0	0	0	0	0
Q	1	0	0	1	2	0	0	0	0	0	0	0
S	1	0	0	0	1	0	2	0	0	1	1	0
ST	0	0	0	0	0	1	2	2	1	0	0	0
W	1	0	0	0	0	0	0	0	0	0	2	1
V	0	0	0	0	0	0	0	0	0	2	1	0

Figure 3. Initial dependency matrix for figure 1.

2. Directed Graph Matrix

The Directed Graph (DG) Matrix for determinant keys represents all the possible direct dependencies between determinant keys. The Directed graph is an n×n matrix where n describes the number of determinant keys. We can generate DG matrix as shown in Figure 4:

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

Initially set all the elements of DG matrix to zeroes. Then scan the DM matrix row by row.

	M	N	S	ST	W	V
M	0	0	0	0	0	0
N	0	0	0	0	0	0
S	0	0	0	0	0	0
ST	0	0	0	0	0	0
W	0	0	0	0	0	0
V	0	0	0	0	0	0

Figure 4. Setting all the elements of the DG matrix as zeroes.

Pseudocode

Using the following pseudo code generate DG Matrix.

```
Directed_Graph_Matrix ( )
{
    For (i=0; i<n; i++)
    For (k=each attribute that composed of determinant key i)
    For (j=0; j<n; j++)
    {
    If (DM[j] [k]! =0 && DG[j] [i]! =-1)
    DG[j] [i] =1;
    Else
    DG[j] [i] =-1;
     }
}
```

Using this, the DG Matrix for the above example is generated as Figure 5:

	M	N	S	ST	W	V
M	1	-1	-1	-1	-1	-1
N	1	1	-1	-1	-1	-1
S	1	1	1	-1	1	1
ST	-1	-1	1	1	-1	-1
W	1	-1	-1	-1	1	-1
V	-1	-1	-1	-1	1	1

Figure 5. DG matrix.

Now our aim is to determine all possible paths between all pairs. This matrix shows all transitive dependencies between determinant keys. If a path can be found from node i to every element of subset of node j in graphical representation of the dependencies either directly or indirectly then set path [i] [j] = 1 else set path [i] [j] = -1.The path matrix for above example is shown in Figure 6:

	M	N	S	ST	W	V
M	1	-1	-1	-1	-1	-1
N	1	1	-1	-1	-1	-1
S	1	1	-1	-1	1	1
ST	1	1	1	1	1	1
W	1	-1	-1	-1	1	-1
V	1	-1	-1	-1	1	1

Figure 6. Determinant key transitive dependencies.

Dependency closure procedure identifies the dependencies. Pseudocode for dependency closure is given below:

```
Dependency_closure ( )
{
    For (i=0; i<n; i++)
    For (j=0; j<n; j++)
    If (i! =j && path[i] [j]! =-1)
    {
    For (k=0; k<m; k++)
    If (DM[j] [k]! =0 && DM[j] [k]! =2)
    DM[i] [k] =j;
    }
}
```

The final Dependency closure matrix for our example is shown in Figure 7.

	M	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**
M	**2**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Q	**1**	**M**	**M**	**1**	**2**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
S	**W**	**Q**	**Q**	**Q**	**1**	**0**	**2**	**0**	**0**	**1**	**V**	**W**
ST	**W**	**S**	**S**	**S**	**S**	**1**	**2**	**2**	**1**	**S**	**V**	**W**
W	**1**	**M**	**M**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**1**
V	**W**	**W**	**W**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**1**	**0**

Figure 7. Dependency closure matrix.

IV. NORMALIZATION PROCESS

As the descriptions of different normal forms are already given, we may proceed with the algorithm. The process of normalization makes use of both determinant key dependency and transitive dependencies.

1. Second Normal Form (2NF)

When all the other attributes depend on a set of attributes then that set of attributes is called as candidate key. From the final DM we found that ST is the candidate key. Thus we got the database in 1NF. The resulting 1NF relation is:

ST_Relation:{ST ,M,N,O,P,Q,R,U,V,W,X}

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 4, April 2018

Next step is to create the 2NF form, we need to remove all partial dependencies. In order to do this, the DM is scanned row by row (ignore the primary key row), starting from the very first row. If all values of the simple keys which is used to create the determinant key of the row being scanned are equal to 2 and the values of the corresponding columns of the candidate key are equal to 2, then the partial dependency is found.

In above table, dependency of S to ST is partial. Therefore, we have to create a new table. From the DM matrix, we notice that Q and V are directly dependent to S. The new table will be composed of S, Q, V, and all simple keys which are transitively dependent on S. The transitive dependencies are obtained from the determinant key transitive dependencies matrix. S indicates the primary key of this table. There is no other partial dependency. The DM matrix is divided into two new DMs corresponding to new tables which are in 2NF as indicated in Figure 8.

	M	N	O	P	Q	S	V	W	X
M	2	1	1	0	0	0	0	0	0
Q	1	M	M	1	2	0	0	0	0
S	W	Q	Q	Q	1	2	1	V	W
W	1	M	M	0	0	0	0	2	1
V	W	W	W	0	0	0	2	1	0

(a) S_Relation :{ S,Q,V,W,M,N,O,P,X }.

	R	S	T	U
ST	1	2	2	1

(b) ST_Relation :{ ST, R, U}

Figure 8: Database normalized up to NF.

2. Third Normal Form (3NF)

In order to achieve the relation into 3NF, each Directed Matrix is look over row by row starting from the first row. If a determinant key is met whose dependency is neither wholly nor has partial dependent on the primary key, a distinct table to be formed. The new table consists of the determinant key and all other attributes which transitively depend on this key as shown in Figure 9.

(A)

	R	S	T	U
ST	1	2	2	1

(B)

	V	W
V	2	1

(C)

	M	W	X
W	1	2	1

(D)

	Q	S	V
S	1	2	1

(E)

	M	P	Q
Q	1	1	2

(F)

	M	N	O
M	2	1	1

V. RESULTS

Thus we obtained the given relational database in to third normal form (3NF).While the manual approach and the existing algorithms are much time consuming, particularly the process of converting relations into 3NF, but the given algorithm for automatic database normalization is much more successful. The given algorithms are observed with MPI and its implementation results on EDM. It showed that such parallel approach decreases the time efficiently [8]. Using p processors has reduced the time of Automatic Database Normalization to $\frac{n^2-m}{p} + c$ in which n is the number of determinant keys, m indicates the number of simple keys, and c is the communication overhead between the processors.

VI. CONCLUSION

In this paper an automated relational database normalization method is presented. We are doing normalization of the given database by generation of dependency matrix, determinant key transitive dependency matrix and directed graph matrix. Normalization upto 2NF, 3NF have been discussed in details. A complete illustration of an example is given, and the defined algorithms have been applied in order to generate the desired final tables. As a side product of the given algorithms, the automatic distinctive of one primary key for each final table is generated. We believe that this algorithm is very efficient as compared to the others which we surveyed. In future we will compare it with more algorithms. We are also applying this algorithm and pseudocodes in developing a user friendly Graphical User

Interface (GUI). This GUI is used for normalizing databases effectively taking varied user inputs.

REFERENCES

[1] M. Arenas, L Libkin, "An Information-Theoretic Approach to Normal Forms for Relational and XML Data", Journal of the ACM (JACM), Vol. 52(2), pp. 246-283, 2005.

[2] Kolahi, S., "Dependency-Preserving Normalization of Relational and XML Data", Journal of Computer SystemScience, Vol. 73(4): pp. 636-647, 2007.

[3] R. Elmasri, S.B. Navathe. "Fundamentals of Database Systems", Third Edition: pp.490, 2000.

[4] Du H., and L. Wery, "A Normalization Tool for Relational Database Designers", Journal of Network and Computer Applications, Volume 22, No. 4, pp. 215-232, October 1999.

[5] Yazici, A., and Z. Karakaya, "Normalizing Relational Database Schemas Using Mathematica", LNCS, Springer-Verlag, Vol.3992, pp. 375-382, 2006.

[6] Akehurst, D.H., B. Bordbar, P.J. Rodgers, and N.T.G.Dalgliesh, "Automatic Normalization via Metamodelling", ASE 2002 Workshop on Declarative Meta Programming to Support Software Development, 2002.

[7] A. H. Bahmani, M. Naghibzadeh, and B. Bahmani. "Automatic database normalization and primary key generation", IEEE CCECE/CCGEI, pages 11–16, May 2008.

[8] Amir -H. Bahmani, S.Kazem Shekofteh, Mahmoud Naghibzadeh, Hossein Deldari, "Parallel Algorithms for Automatic Database Normalization", IEEE/ICCAE, Publication, Year: 2010, Page(s): 157-161.

[9] Sungchul Lee, Juyeon Jo, Yoohwan Kim, Haroon Stephen, "A Framework for Environmental Monitoring with Arduino-Based Sensors Using Restful Web Service", Services Computing (SCC) 2014 IEEE International Conference on, pp. 275-282, 2014.

[10] Kunal Kumar, S. K. Azad, "Database normalization design pattern", Electrical Computer and Electronics (UPCON) 2017 4th IEEE Uttar Pradesh Section International Conference on, pp. 318-322, 2017.

[11] Date, C.J., "An Introduction to Database Systems", Addison-Wesley, Seventh Edition 2000.

[12] S. Lucas, J. Meseguera, "Normal forms and normal theories in Conditional rewriting", Elsevier Journal of Logical and Algebraic Methods in Programming, vol. 85, pp. 67-97, 2016.

[13] R. Vangipuram, R. Velputa, V. Sravya, "A Web Based Relational database design Tool to Perform Normalization", International Journal of Wisdom Based Computing, vol. 1, no. 3, 2011.

[14] Moussa Demba, "Algorithm for relational database Normalization up to 3NF", International Journal of Database Management Systems, vol. 5, no. 3, June 2013.

[15] Ivan Ubaleht, "The design of relational database schemes based on the elementary relationships of attributes: Computing of the closure of a set of attributes for one type of relationship", Engineering Computer and Information Sciences (SIBIRCON) 2017 International Multi-Conference on, pp. 360-364, 2017.

[16] W.-D. Langeveldt, S. Link, "Empirical evidence for the usefulness of Armstrong relations in the acquisition of meaningful functional dependencies", Inf. Syst., vol. 35, no. 3, pp. 352-374, 2010.

[17] A. Yazici, Z. Karakaya, "Normalizing Relational Database Schemas Using Mathematica", LNCS, Springer-Verlag, vol. 3992, pp. 375-382, 2006.

[18] Du H., and L. Wery, "A Normalization Tool for Relational Database Designers", Journal of Network and Computer Applications, Volume 22, No. 4, pp. 215-232, October 1999.

[19] Akehurst, D.H., B. Bordbar, P.J. Rodgers, and N.T.G. Dalgliesh, "Automatic Normalization via Metamodelling", ASE 2002 Workshop on Declarative Meta Programming to Support Software Development, 2002.