A Low End Case Tool Synthesizing 3NF Relations Using Bernstein's Algorithm

Ruchika sharma[1]

Department of Computer Science and Engineering

Noida Institute Of Engineering and Technology

sruchika40@gmail.com

Areeba Kazim[2]

Department of Computer Science and Engineering

Noida Institute Of Engineering and Technology

aribakazim1995@gmail.com

Abstract:

Database design is considered good if there are minimum redundancies, no inconsistencies and no anomalies. The quality of a relation schema can be measured using formal and informal approaches. Informal approaches are the basic guidelines for a good database design where as the formal approach for measuring the quality of a relational schema uses the concept of normal forms. The two approaches for designing the database are decomposition approach and synthesis approach. The decomposition approach is time consuming and lengthy where as synthesis approach uses the Bernstein algorithm to convert a relation directly into 3NF. The tool being designed uses synthesis approach and also generates DDL statements for entire database. Students generally find normalization a theoretical concept so a tool has been developed to automate the process of normalization and use it as an teaching aid to enhance their expertise in normalization.

1. Introduction

Keywords

Normalization, Relation database, automatic normalization, formal approach

1.1. Definition of a good database design

A good database design has minimum redundancies, no inconsistencies, is lossless and should preserve dependencies.[19]

The term lossless means that if a relation A is decomposed into relation A_1, A_2,..A_n and if a projection or natural join operations are applied on each of those decomposed relations, the relation obtained should exactly be A. There should be no generation of spurious tuples that represent wrong information. Lossless doesn't mean loss of tuples; it simply means loss of information because of spurious tuples. Dependency preservation means dependencies of R are represented in one or the other decomposed relations. Although for a good database design dependency preservation can be compromised as compared to lossless property**.**

1.2 There are two methods for measuring the quality of a relational schema [1][2]

- First method is informal approach in which basic qualities of a relational database design are considered like anomalies and redundant tuples, spurious tuple generation,redundant information in tuples.
- The formal approach for measuring the quality of a database design uses the concept of normalization. Manual process of normalization process is a very tedious task and many normalization algorithms have been developed to automate this process. Micro, JMathNorm , CodaSys are many normalization tool. Basic tool that is used for measuring the quality of a database design are functional dependencies. Functional dependency is the basic idea or tool behind normalization. The paper discusses upto third normal form because upto third normal form most of the redundancies are reduced.

The three normal forms are:

- **First normal form (1NF)**: In 1NF only single values are allowed, relations with in relations are avoided.

- **Second normal form (2NF)**: In 2NF attribute which are not key should be fully functionally dependent on the primary key of a relation
- **Third normal form (3NF)**: In 3NF transitive dependencies are removed.

Thus if a relation is in 1NF, it has much redundancy which results into inconsistency and so insertion ,deletion update anomalies, then it has to be normalized for a better design. .

In this case a relation needs to be normalized to 2NF. A relation in 2NF has reduced redundancies, inconsistencies, anomalies but still can be reduced by normalizing it into 3NF. A relation in 3NF has least redundancy and least inconsistency.

In 3NF redundancies and inconsistencies are minimum and quality of the design gets better.

1.3 The two approaches for generating 3NF relation:[1][17]

- **Relational design by analysis:** It is a top-down approach or decomposition approach in which each relation is checked to see to which normal form a relation belongs and is decomposed then accordingly.
- **Relational design by synthesis:** This is synthesis approach in which functional dependencies are needed to generate 3 NF relations directly.

In this paper a tool has been devised that can convert a relation into 3NF using Bernstein's algorithm .

2. Literature Survey

Manual process of normalization requires large number of persons having expertise in normalization . but it had lot of drawbacks as it was more time consuming, due to large amount of relations and functional dependency it was more prone to errors and was costly too. So it was an idea why not to make the process of normalization automatic by using new tools and methods.**[13]** [22].

There is one theory named as normalizarion systems which is divided into modular and submodular tasks. As we see in database , normalization helps to design the database in the same way as the theory of normalization system deals with.[15]

The following section gives a brief description about all the tools proposed so far so as to automate the process of normalization there by describing why normalization needs to be automated.

In one of the patent **[18]** a normalization system is conceived in which database is normalized by observing a collection of records stored in a table. Micro is a tool proposed by Hong Du and Laurent Wery**[19]** where two linked list are used attributes are stored in one and functional dependencies are stored in the other

This way the entire relation along with all the dependencies that hold on it are represented by a two linked list . But it consumes both space and time and if this problem could be solved by taking only one linked list that RDBNORMA comes into the role and is one of the proposed tool**[13]**.

Mathematica is a software system built in Wolfran Progamming language often referred as Wolfran Mathematica has been developed by Stephen Wolfran. It is used to build algorithms and software for evaluation of mathematical expressions , plot functions, creating user interface, matrix manipulation all could be done by mathematica. **[13]**

JMathNorm tool was proposed by Ali Ya zici ,et.al . It can normalize a relation upto BCNF . GUI of JMathNorm has been developed in java andit has been linked with mathemaica using Jlink library. **[7]**

Since it is considered as a theoretical concept by students so teaching normalization and its importance in large scale industry is a big and open challenge . LBDN is one such interactive tool**[9]** developed which takes as input as a table given by a user and functional dependencies that holds among the set of attributes and output is given as a set of BCNF normalized relation. Apart from normalizing a relation it helps in teaching the process of normalization by giving hands on practice to students so that they could learn by themselves as it demonstrates the whole process by giving detailed description at each step.

Functional dependency which are considered as a tool for normalization process can be represented by using three structures: Dependency Graph(DG),Dependency Matrix(DM), Directed Graph Matrix(DG) **[11]**

Functional dependency which are considered as a tool for normalization process can be represented by using three structures: Dependency Graph(DG),Dependency Matrix(DM), Directed Graph Matrix(DG) **[11]**

In **[21]** bubble diagram was used to give pictorial representation of FD's. ER diagram is used as the base for the construction of bubble diagram by finding all the prime attributes of the relation and then entire entity set is replaced by primary key bubble then the remaining attributes are connected by this key bubble. Database developers are not able to understand bubble diagram so it is a doubtful applicable concept form industrial point of view. **[14]**

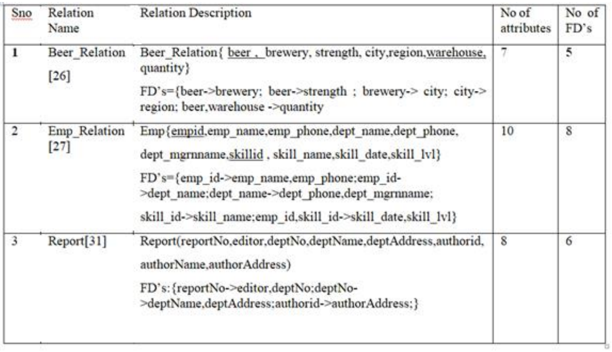
In **[22]**author states that if database developer should incorporate normalization concepts into the ER model so that when via mapping algorithms it is converted into relational schema then the resultant should be normalized. Normalization requires understanding of the organization of domain information and semantic dependencies to be taken care of. This knowledge is very difficult to be achieved at the time of conceptual design.

A modeling tool has been made by extending Graphical Editing Framework(GEF) and Draw2d called as AER IDE (Articulated entity relation) which is used for drawing, storing, validating and normalizing AER diagrams. AER IDE allows automation of normalization process at the conceptual design stage. JMathNorm , Micro do not allow normalization at the conceptual level. **[14]**

These tools only take relational schema as an input. It does normalization at the relational level. Normit **[15]** and Web based Normalization tool**[9]** all these are learning aided tools which makes teaching normalization easy by training and teaching the students practically rather than doing it theoretically.

In [13] the following table shows the relations along with their dependencies. Later on their these relations will be used to see how Micro and RDBNORMA could comparatively normalize these relations

Table 1: Depicting relations along with FD's



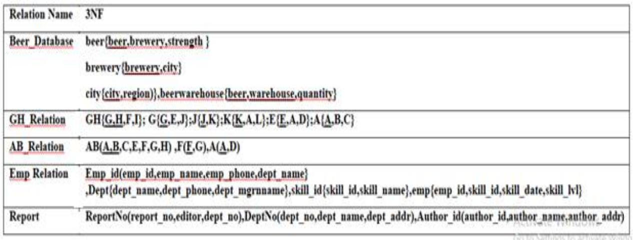
Sno	Relation Name	Relation Description	No of attributes	No of FD's
1	Beer_Relation [26]	Beer_Relation{ beer , brewery, strength, city,region,warehouse, quantity} FD's={beer->brewery; beer->strength ; brewery-> city; city-> region; beer,warehouse ->quantity	7	5

2	Emp_Relation [27]	Emp{empid,emp_name,emp_phone,dept_name,dept_phone,
		dept_mgrnname,skillid , skill_name,skill_date,skill_lvl}
		FD's={emp_id->emp_name,emp_phone;emp_id->dept_name;dept_name->dept_phone,dept_mgrnname;
		skill_id->skill_name;emp_id,skill_id->skill_date,skill_lvl}
			10	8

3	Report[31]	Report(reportNo,editor,deptNo,deptName,deptAddress,authorid, authorName,authorAddress) FD's:{reportNo->editor,deptNo;deptNo->deptName,deptAddress;authorid->authorAddress;}	8	6

In **[13]** experimental results of RDBNORMA were aggregated to see how these relations will be normalized to 3NF.

Table 2: Depicting relations normalized to 3NF



Relation Name	3NF

Beer_Database	beer{beer,brewery,strength }
	brewery{brewery,city}
	city{city,region)},beerwarehouse{beer,warehouse,quantity}

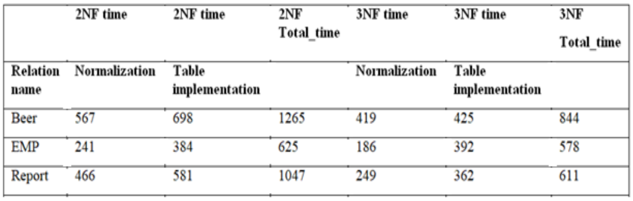
GH_Relation	GH{G,H,F,I}; G{G,E,J};J{J,K};K{K,A,L};E{E,A,D};A{A,B,C}
AB_Relation	AB(A,B,C,E,F,G,H) ,F(F,G),A(A,D)

Emp Relation	Emp_id(emp_id,emp_name,emp_phone,dept_name}
	,Dept{dept_name,dept_phone,dept_mgrnname},skill_id{skill_id,skill_name},emp{emp_id,skill_id,skill_date,skill_lvl}

Report	ReportNo(report_no,editor,dept_no),DeptNo(dept_no,dept_name,dept_addr),Author_id(author_id,author_name,author_addr)

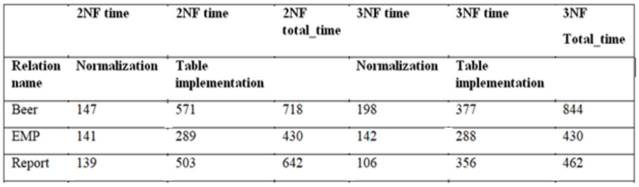
In IJDSB comparison between RDBNORMA and Micro[Du and Wery , 1999] has been done. They are implemented using java. And this comparison between tools was done on the basis of time taken by both to convert a relation from2NF to 3 NF. RDBNORMA is much faster in comparison to Micro . Micro is 3.9 times and 2.89 times as required by RDBNORMA when converting relations to 2NF and 3NF respectively.**[6][13]**

It was observed from this comparison that time required not only depends on the number of attributes but also on the number of functional dependencies that holds on the relation. [13]



	2NF time	2NF time	2NF Total_time	3NF time	3NF time	3NF Total_time
Relation name	Normalization	Table implementation		Normalization	Table implementation	
Beer	567	698	1265	419	425	844
EMP	241	384	625	186	392	578
Report	466	581	1047	249	362	611

Table 3: analysis of timing in MICRO



	2NF time	2NF time	2NF total_time	3NF time	3NF time	3NF Total_time
Relation name	Normalization	Table implementation		Normalization	Table implementation	
Beer	147	571	718	198	377	844
EMP	141	289	430	142	288	430
Report	139	503	642	106	356	462

Table 4: Analysis of timing in RDBNORMA

2.1 Decomposition approach:

In this approach functional dependencies and a relation is taken as input. A relation is checked whether it is in 1NF or not according to criteria of normal forms. In case it is not, appropriate remedies are adopted and it is converted to 1 NF. Now, this decomposed 1NF relation is checked against the criteria of 2 NF. In case, the decomposed relation does not satisfy the criteria of 2 NF, then a remedy is followed to convert a relation into 2NF. Again, decomposed relations (if formed) are checked against the criteria of 3NF. If it is not then a remedy is followed 3NF relation conversion. This approach is a top-down approach as it works step by step by first converting a relation into 1 NF then to 2NF and then to 3NF. In this way a good database design can be converted to better database design with no anomalies left further. **[2][10]**

Following is a table that shows the summarization of normal forms , the test that are used to check in which particular normal form a relation is and in case if a relation is not in that particular normal form what remedies should be adopted to normalize a relation so as to bring it (normalize) to that particular form.

Two objections to this approach as a technique for a database design are: [12]

- An initial design has to be found by the designer which is ultimately improved by 3NF decomposition
- An existing relation scheme is converted into decomposed relations instead of combining it into larger schemas.

2.2 Synthesis approach

This approach uses Bernstein algorithm which is a bottom-up approach. It is not time consuming and lengthy like the earlier mentioned decomposition approach.**[1]**

It ensures

- Dependency preservation
- Non additive join property and
- Each resulting relation schema in the decomposition is in 3NF.

Bernstein algorithm synthesizes the output directly into 3NF. It first finds the minimal cover for a given set of dependencies that holds on the relation . Then the left hand side of functional dependencies (obtained from the minimal cover) is merged with their right hand sides. This creates a new relation schema with left hand side as the primary key of those individual relations formed. **[1]**
Bernstein's algorithm ensures lossless join and dependency preservation. But as a part of the algorithm, it is verified whether lossless join property is preserved or not. **[1]**

If the primary key of a universal relation R, obtained by the closure algorithm, is present in any of the decomposed relations formed, then it has lossless join property.In case, if a primary key is not present then a new relation is formed which contains the attributes that form the key of R so as to (having primary key as the attribute) attain lossless join property. Unlike lossless join property, dependency preservation is already ensured by Bernstein's algorithm so functional dependencies are preserved in relations, and it is not necessary to check the same. **[1] [4]**

In this way Bernstein algorithm generates relations directly into 3NF, with those relations holding lossless join and dependency preservation property both**[4]**

3. Proposed Methodology:

The proposed methodology for designing the database is synthesis approach i.e. Bernstein algorithm is used.

The algorithm computes a relation directly into 3NF along with the integrity constraints as follows: **[1] [2]**
Bernstein's algorithm uses other algorithm's too. That is , the first step is to compute the minimal cover for a given set of FD's using algorithm that is used to compute minimal cover. Minimal cover are minimal set of FD's which are in standard form and has no redundancies. Such a standard form will be obtained from three conditions computed in a step by step manner. First is there should a single attribute on RHS of FD's. Secondly, extraneous attributes should be removed from LHS of FD's and finally transitive dependencies if exists are removed . When all these three conditions are fullfilled then minimal cover is obtained from a given set of FD's which are free from redundancies.

Third step of the Bernstein's algorithm that checks whether the relation formed have lossless join property or not uses an algorithm that computes primary key of a relation. This computation is done using the concept of closure algorithm. Therefore, three algorithms are basically used to transform a relation into 3NF , first is the minimal cover algorithm, primary key computation algorithm which uses closure algorithm also. So these three algorithms together form Bernstein's algorithm.**[1][19].**

The input given is a universal relation R and a set of functional dependencies F on the attributes of R.

At the first stage, minimal cover for a given set of functional dependencies is computed. **[17][18][13]**

Functional dependencies that can be derived from other are removed. That is why the relations formed using minimal cover has minimum redundancies.

- Left hand side of functional dependencies obtained in minimal cover, are merged with their right hand side. This results in the creation of new relations with left hand side of those relations as primary key of the relation.
- The right hand side attributes are dependent on left hand side attributes so, merging left hand side with right hand side will bring functional dependencies with same left hand side attributes into one single relation with left hand side as primary key .The relations thus formed are in 3NF and hold dependency preservation property.
- At the last stage it is checked whether the relations formed have lossless join property or not.

Primary key is present in any one of the decomposed relation then the relations have lossless join property else a new relation is created with attributes forming a key of relation, as it might be possible that relations formed do not possess lossless join property, that is it might contain spurious tuples, so such relations can be made to possess lossless join property by creating such a new relation.

Apart from this, Bernstein's algorithm ensures dependency preservation which is not required to be checked (by default) as the relations are formed by using minimal cover which uses minimum set of functional dependencies without violating anyone. However, if required to be checked, it is also possible to check whether the relations formed preserve dependencies or not. For that, it is checked that all the functional dependencies given as input are present in some of the relations formed so that none of them is violated.**[11][5]**

The relations thus formed are in 3NF having lossless join and dependency preservation property.

Finally DDL statements are created for the entire database with integrity constraints specified and the choice of data type to be used is taken care of by the end user.

Example: create table <relation name>(<attribute

name><*data type(empty)*><primary key/foreign key constraint>…..);

The below section states the algorithm used

3.1 Bernstein's Algorithm to synthesize a relation into 3NF ensuring Dependency Preservation and Nonadditive (Lossless) Join Property [1][19]

Input : A universal relation A and a set of FD's B on the attributes of A.

```
• Find a minimal cover M for B
• For each left hand side A of FD's that appear in M create a relation schema in D with attributes $A \cup \{C_1\} \cup \{C_2\} \ldots \{C_k\}$ where $A\text{->} C_1, A\text{->} C_2, \ldots A\text{->} C_k$ is the only dependency in with A as left hand side (A is the key of the relation)
• If none of the relation schema in M contain a key of A, then create one more relation schema in D that contains attributes that form a key in A.
```

3.2 Algorithm to compute Minimal cover

```
• Set B:=E
• Replace each FD $B\text{->}\{C_1, C_2 \ldots C_n\}$ in B be the n FD's
  $A\text{->} C_1$
  $A\text{->} C_2 \ldots\ldots A\text{->}C_n$
• For each FD A->C in F
  for each attribute E that is an element of X
  if{{M-{ A->C}}∪{X-{E}})->C is equivalent to F,
  then replace A->C with (A-{E})->C in F
• For each remaining FD A->C in M
  if {M-{A->C}} is equivalent to M,
      then remove A->C from M
```

3.3 Algorithm to find a key K for a universal relation R when given a set F of FD's [1]

Input: A universal relation A and a set of FD's M on the attributes of A

```
• Set K:=A
• For each attribute C in K
  {
      Compute $(K\text{-}C)^+$ with respect to M;
      If $(K\text{-}C)^+$ contains all the attributes in A, then set K:=K-{C}
  };
```

As per the algorithm , the key K is set to all the attributes of relation R, then the process starts by removing attributes one at a time and then checking whether remaining attributes are still forming a superkey of A. If after removing a particular attribute , remaining attributes are still forming a superkey that means that the remaining attributes are still forming a superkey when calculated using closure algorithm **[1][9]** .

This step is repeated for every attribute turn by turn for each cycle. At the end of the cycle those attributes that are left that is , whose removal causes problems as all attributes were not obtained when closure for that particular attribute is calculated , were included and at the end these attributes together forms a primary key**[9].**

After the primary key has been found , it is checked against the decomposed relations whether it is present in any one of the relations synthesized as done by the previous step of the Bernstein's algorithm **[10][16][9].**

3.4 Algorithm to compute closure X^+ :[1]

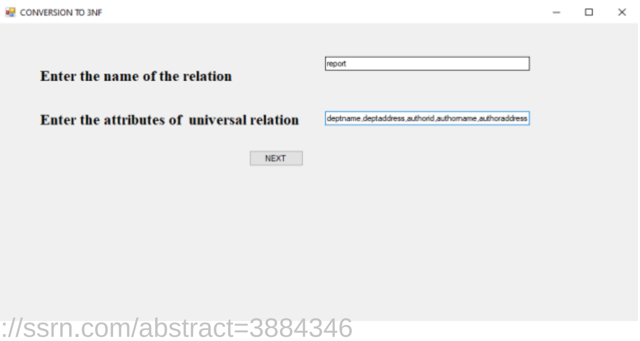
```
• $A^+ := A$
• Repeat
    Old $A^+ = A^+$ ;
    for each FD Y->Z in M do
    if $A^+ \supseteq Y$ then $A^+ := A^+ \cup Z$;
    until ($A^+ =$ old $A^+$ );
```

According to the above algorithm , closure of the left hand side of FD's is computed as by taking the LHS of the FD whose closure is computed as it is in the set and then all the FD are checked against the attribute in the set to check if the set obtained is a subset of any of the LHS of FD's.

If so, then RHS of that particular of which it is a subset is added to the set obtained initially,. Again the cycle goes to check if the attribute obtained in the set after checking the FD's again forms the subset or not. If all the attributes are obtained it is considered to be a full closure, if not then it is not considered full closure.

4. Description of the tool

The software that is used to develop the tool is Microsoft Visual studio 2008 as it helps to make applications user friendly and interactive. And real life scenario has been taken that shows how a relation can be synthesized into 3NF directly instead of using manual and time consuming process.**[15]**

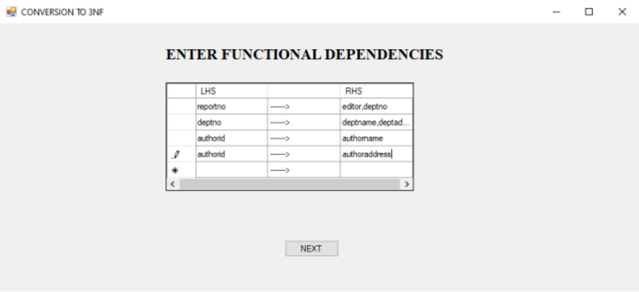


Enter the name of the relation: report

Enter the attributes of universal relation: deptname,deptaddress,authorid,authorname,authoraddress

Fig. 1 - universal relation screen

In this window user will be asked to enter the name of the relation and along with it attributes of the relation are also allowed to enter. In case field is left empty error message is displayed which serves as a validation check.



ENTER FUNCTIONAL DEPENDENCIES

	LHS		RHS
	reportno	------->	editor,deptno
	deptno	------->	deptname,deptad...
	authorid	------->	authorname
✎	authorid	------->	authoraddress
*		------->	

Fig. 2 – functional dependency module

On the click event of next button functional dependency module will be displayed which will allow the user to enter LHS and RHS of functional dependency. Data and input entered in the previous screen is carried over to the next screen. ADD button allows the user to enter functional dependency in a data grid view. The dependencies entered are clear

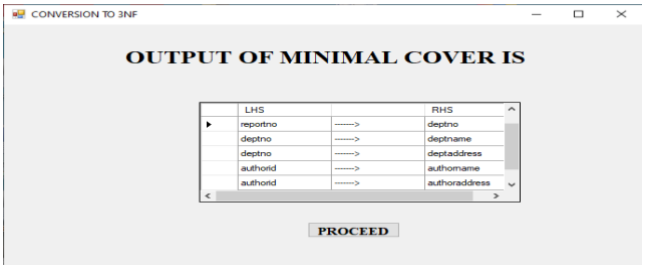
Report(reportNo,editor,deptNo,deptName,deptAddress,authorid,authorName,authorAddress)

FD's:{reportNo->editor,deptNo;

DeptNo->deptName,deptAddress;authorid->authorAddress;}

Again on the click event of Next button two buttons will appear named as step by step and final result. If a user directly wants to view a relation after being normalized into 3NF then final result button is used. In case if a user wants to view how it is converted into 3NF using synthesis or bottom up approach then step by step button is used.

When Step by Step button is clicked

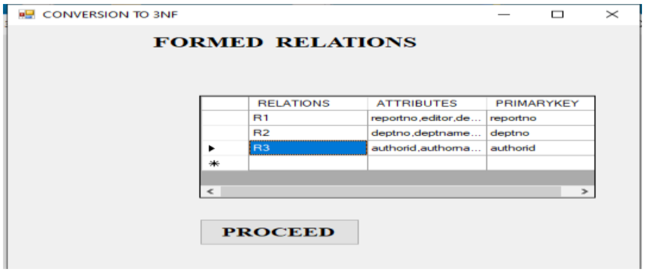


OUTPUT OF MINIMAL COVER IS

	LHS		RHS
▶	reportno	------->	deptno
	deptno	------->	deptname
	deptno	------->	deptaddress
	authorid	------->	authorname
	authorid	------->	authoraddress

Fig. 3 – minimal cover computation screen

This screen gives the output of the first screen which minimal cover of the given set of functional dependency.

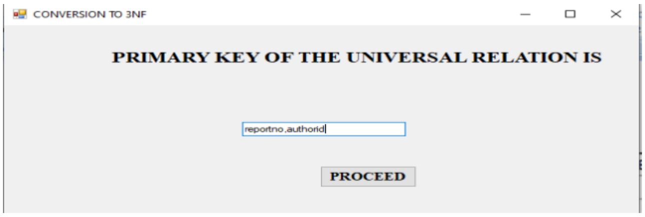


FORMED RELATIONS

	RELATIONS	ATTRIBUTES	PRIMARYKEY
	R1	reportno,editor,de...	reportno
	R2	deptno,deptname...	deptno
▶	R3	authorid,authorna...	authorid
*			

Fig. 4 – formed relations screen

On the click event of proceed button next window will appear showing next computation done as the next step of the algorithm is to form the relations using FD's obtained as a result of minimal cover obtained in previous step. The relations formed in this step will ensure dependency preservation.



PRIMARY KEY OF THE UNIVERSAL RELATION IS

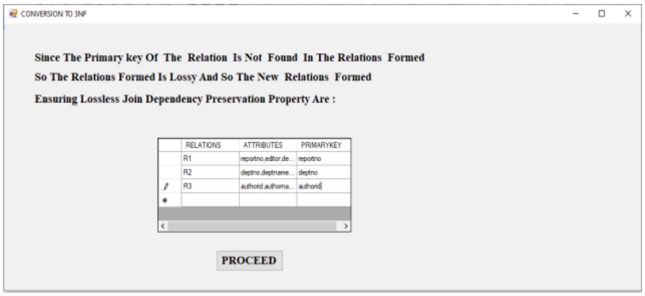
reportno,authorid

Fig.5- primary key computation screen

The next step of the algorithm is to compute the primary key of the Universal Relation . One can easily find from this relation whether a relation formed have lossy or lossless join property .

In case if a primary key of a Universal relation is not found in the relations formed then it is said to be lossy. In that case , the algorithm adds the attributes forming the primary key of the relation to the relations formed so as to make it lossless and if a primary key of the universal relation is already found in the relations formed then it is said to hold lossless join property. That's why it is said that the Bernstein's algorithm ensures lossless join property and also dependency preservation.

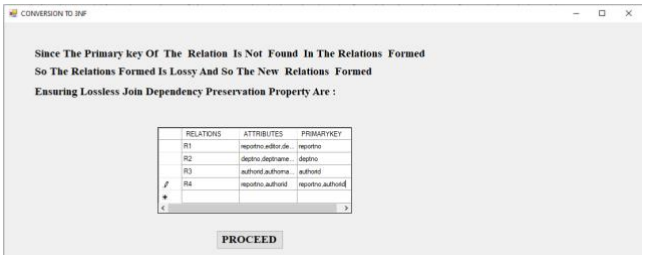
Below screenshot checks whether relations formed are lossy or lossless. In the example taken at first key was not found so it came out to be lossy later key was added to allow the meaning to be added to the relation there by making it lossless .



Since The Primary key Of The Relation Is Not Found In The Relations Formed
So The Relations Formed Is Lossy And So The New Relations Formed
Ensuring Lossless Join Dependency Preservation Property Are :

	RELATIONS	ATTRIBUTES	PRIMARYKEY
	R1	reportno,editor,de...	reportno
	R2	deptno,deptname...	deptno
	R3	authorid,authorna...	authorid
*			

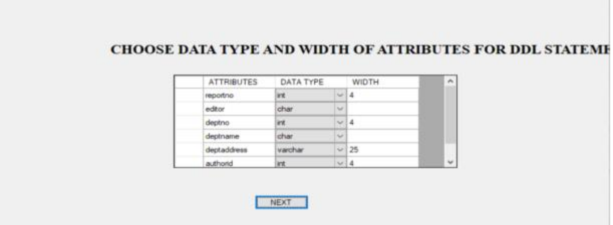
Fig. 5- relations formed are lossy



Since The Primary key Of The Relation Is Not Found In The Relations Formed
So The Relations Formed Is Lossy And So The New Relations Formed
Ensuring Lossless Join Dependency Preservation Property Are :

	RELATIONS	ATTRIBUTES	PRIMARYKEY
	R1	reportno,editor,de...	reportno
	R2	deptno,deptname...	deptno
	R3	authorid,authorna...	authorid
	R4	reportno,authorid	reportno,authorid
*			

Fig. 6- relations formed are lossless

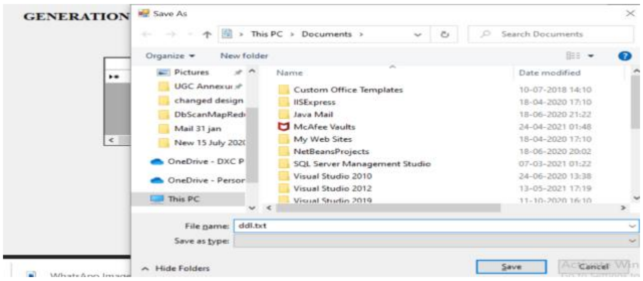


CHOOSE DATA TYPE AND WIDTH OF ATTRIBUTES FOR DDL STATEME

ATTRIBUTES	DATA TYPE	WIDTH
reportno	int	4
editor	char	
deptno	int	4
deptname	char	
deptaddress	varchar	25
authorid	int	4

Fig. 7- screenshot depicting datatype and width of attribute

After the window immediately the user will see next window in which user is allowed to choose data types and enter width for the same. The user can choose data types from the drop down list and in case of char data type field the width column can be left blank as by default it will take the width of char data type as 1.



Name	Date modified
Custom Office Templates	10-07-2018 14:10
IISExpress	18-04-2020 17:10
Java Mail	18-06-2020 21:22
McAfee Vaults	24-04-2021 01:48
My Web Sites	18-04-2020 17:10
NetBeansProjects	18-06-2020 20:02
SQL Server Management Studio	07-03-2021 01:22
Visual Studio 2010	24-06-2020 13:38
Visual Studio 2012	13-05-2021 17:19
Visual Studio 2019	11-10-2020 16:10

File name: ddl.txt
Save as type:

Fig. 8- save module

After choosing the data type and width , the user can click over the next button to open DDL statement generation module where tool would automatically generate DDL statements for the entire database and can use save button to save the file in MS Word .

On the click event of final result button

If a user directly wants to view the result then go to the final result button and get the screen for filling in the data type and width of the attributes and directly DDL statements will be generated

5. Impact of proposed research in academic/ industry:

Manual calculation for deriving a 3NF relation using Bernstein algorithm is long , tedious task and requires deep understanding of the algorithm by the user. In present scenario where a large amount of data is there which is required to normalized such a tool could be boon there by making a tedious task simple.

No such deep understanding of the algorithm is needed by the user as human hardwork will be converted into automated task. Also, in industries where large databases are used such a tool can be of great importance as it will directly generate a relation into 3NF .

In academics, students will be able to see that what they calculate manually is correct or not by matching it with the results of the tool, that is, practically they will be able to understand the normalization concept more. Apart from this the tool also provides the features of saving word file and at later stage printing option could be given .

Further it can be made online so that as a free tool we could use and enhance the learning process and videos could be added as a teaching tool for step by step description of the process and lectures assignments could be attached to it which could act as a self training tool and this can be further enhanced by mapping it into ER model as well6. References

6. References

1.Codd E. F. (1970), “A relational model of data for large shared data banks”, Communications of the ACM. vol. 13, No.6, pp. 377–387.

2. Codd E. F. (1971), “Further normalization of the data base relational model", IBM Research Report, San Jose, California, vol. RJ909

3. Philip A. Bernstein,”Synthesizing Third Normal Form Relations from Functional Dependencies” A.C.M transactions on Database Systems, Vol.1, No. 4, December 1976, pp. 277-298

4. Ronald Fagin,”The decomposition versus the synthetic approach to relational database design”, Proc. 3rd Intl.

Conf.Very Large Databases Conference, Tokyo, Japan, 1977, pp 441-446.

5. Beeri C and Bernstein P. A.(1979), “Computational problems related to the design of normal form relational schemas”, ACM Transactions on Database Systems, Vol.4, No.1, pp.30-59.

6. Date C. J. (1986), “An introduction to database system”, fourth edition, Addison Wesley.

7. Dawson kathyn s.& parker lorraine m. Purgailis,.(1998) “From Entity Relationship Diagram to Fourth Normal Form: A Pictorial Aid to Analysis”, The Computer Journal, Volume 31, Number 3

8. Hetch and Stephen C (1998), US Patent 5778375 - Database normalizing system

9. Du H and Wery L (1999), “ Micro: A normalization tool for relational database designers”, journal of network and computer application, Vol.22, pp.215-232.

10. Antony S. R. and Batra D (2002), “CODASYS: A consulting tool for novice database designers”, ACM SIGMIS, vol.33, issue 3, pp.54-68.

11. Mitovic Antonija,(2002) “NORMIT: Web-Enabled Tutor for Database Normalization”, Proceedings of IEEE International Conference on Computer Education(ICCE-02).

12. TAUQEER HUSSAIN*, SHAFAY SHAMAIL, & MIAN M. AWAIS(2003) “ Eliminating process of normalization in relational database design”, Proceedings IEEE INMIC CODD E. F.,(1970) “A relational model of data for large shared data banks”, Communications of the ACM, vol. 13, No.6, pp. 377-387.

13. Elmasri Navathe ,Somayajulu Gupta .Fundamentals of Database Systems, 4th Edition, Pearson Education, Inc.,2004

14. Beynon-Davies, P. (2004) Database systems, Palgrave Macmillan, Third edition, ISBN 1–4039— 1601–2.

15. https://www.researchgate.net/publication/47554361_A_Novel_Database_Design_for_Student_Information_System?enrichId=rgreq-8ce88610657f6c512a65a2c497417a20-XXX&enrichSource=Y292ZXJQYWdlOzQ3NTU0MzYxO0FTOjQyNTM3OTE4MjcxNDg4MUAxNDc4NDI5NzA4MzU3&el=1_x_3&_esc=publicationCoverPdf

16. Thomas, C., Carolyn, B. (2005) Database Systems, A Practical Approach to Design, Implementation, and Management, Pearson Fourth edition .

17. Kung Hsiang-Jui and Tung Hui-Lien (2006), “A web based tool to enhance teaching/Learning database normalization”, in Proceeding of international conference of southern association for information system.

18. NABIL ARMAN,(2006). “Normalizer: A case Tool to normalize relational database schemas”, Information Technology Journals (2) 329-331 .ISSN 1812-5638.

19. Silberschatz, Korth and S. Sudarshan (2006), “Database system Concepts”, McGraw Hill international edition, Fifth edition.

20. Yazici A, Ziya K (2007), “JMathNorm: A database normalization tool using mathematica”, In proc. international. conference on computational science, pp.186-193.

21. Bahmani A., Naghibzadeh, M. and Bahmani, B. (2008) "Automatic database normalization and primary key generation", Niagara Falls Canada IEEE.