P-3617

A NOVEL WATERMARKING SCHEME FOR DETECTING AND RECOVERING DISTORTIONS IN DATABASE TABLES

A PROJECT REPORT

Submitted by

BAMA T (0710108005)

HEMALATHA B (0710108018)

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

(An Autonomous Institution Affiliated to Anna University of Technology, Coimbatore)

COIMBATORE – 641 049

APRIL 2011

KUMARAGURU COLLEGE OF TECHNOLOGY – COIMBATORE-641 049

BONAFIDE CERTIFICATE

Certified that this project report entitled "**NOVEL WATERMARKING SCHEME FOR DETECTING AND RECOVERING DISTORTIONS IN DATABASE TABLES**" is the bonafide work of Bama T and Hemalatha B who carried out the research under my supervision. Certified also, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

[illegible]

SIGNATURE

Mrs. P. Devaki M.E (Ph.D)

HEAD OF THE DEPARTMENT

Department of Computer
Science and Engineering
Kumaraguru College of Technology
Coimbatore-641049

[illegible]

SIGNATURE

Mr . G.S. Nandakumar M.E (Ph.D)

ASSISTANT PROFESSOR

Department of Computer
Science and Engineering
Kumaraguru College of Technology
Coimbatore-641049

The candidate with University Register Nos. 0710108005, 0710108018 were examined by us in the project viva-voce examination held on 20·4·2011

[illegible]

INTERNAL EXAMINER

[illegible]

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project entitled " **A NOVEL WATERMARKING SCHEME FOR DETECTING AND RECOVERING DISTORTIONS IN DATABASE TABLES** " is a record of original work done by us and to the best of our knowledge, a similar work has not been submitted to Anna University or any Institutions, for fulfillment of the requirement of the course study.

The report is submitted in partial fulfillment of the requirement for the award of the Degree of Bachelor of Computer Science and Engineering of Anna University of Technology, Coimbatore.

T. Bama

Place: Coimbatore (BAMA T)

Date: 20 · 4 · 2011

[illegible]

(HEMALATHA B)

ACKNOWLEDGEMENT

First and foremost, we would like to thank the Lord Almighty for enabling us to complete this project.

We extend our sincere thanks to our management, our chairman, **Dr.N.Mahalingam**, our co-chairman **Dr. Krishnaraj Vanavarayar**, and our director, **Dr.J.Shanmugam** for providing us with the necessary facilities and opportunities.

We extend our sincere thanks to our Principal, Dr.**S.Ramachandran,** Kumaraguru College of Technology, Coimbatore, for being a constant source of inspiration and providing us with the necessary facility to work on this project.

We would like to make a special acknowledgement and thanks to **Mrs.P.Devaki M.E.,(Ph.D).,** Head of Department of Computer Science &Engineering, for her support and encouragement throughout the project.

We express our gratitude, gratefulness and sincere thanks to our Guide

Mr.G.S. Nandakumar M.E (Ph.D)., Department of Computer Science & Engineering, for his supervision, enduring patience, active involvement and guidance.

We would like to convey a special acknowledgement and honest thanks to our class advisor, **Mrs. Ramathilagam** and **all the other Faculty members** of the Department for their enthusiasm and wealth of experience from which we have greatly benefited.

We would also like to thank all our **non teaching staffs** for their constant support.

We also thank our **friends and family** who helped us to complete this project fruitfully.

ABSTRACT

The piracy of assets such as software, images, video, audio, text and databases has long been a concern for owners of these assets. Protection of these assets is usually based upon the insertion of watermarks into the data. The watermarking introduces small errors into the object being watermarked. These intentional errors are called *marks* and all the marks together constitute the *watermark*. The marks must not have a significant impact on the usefulness of the data and they should be placed in such a way that a malicious user cannot destroy them without making the data less useful. The embedded watermark should be robust against attacks which aim at removing the watermark or making it undetectable. Thus, watermarking does not prevent copying, but it deters illegal copying by providing a means for establishing the original ownership of a redistributed copy. The increasing use of databases in applications beyond "behind-the-firewalls data processing" is creating a similar need for watermarking databases.

In this paper a novel fragile watermarking scheme is proposed to detect, localize and recover malicious modifications in relational databases. In the proposed scheme, all tuples in the database are first securely divided into groups. Then watermarks are embedded and verified group-by-group independently. By using the embedded watermark, the modification made to the database are detected and localized and even the true data is recovered from the modified locations. Distortion detection and true data recovery both are performed successfully.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

With the widespread use of computers and internet, access and exchange o data became an extremely simple task. Since data can be easily duplicated and modified there is a great deal of concern about the integrity and intellectual property protection o these data.

A new technology, known as watermarking, provides a promising method o protecting data from illicit copying and manipulation by embedding a secret code directl into the data. The embedded secret code, called watermark, can be used in variou applications such as copyright protection, integrity checking, and fingerprinting. In short watermarking refers to embedding a secret imperceptible signal (watermark) in th original data. Watermarking schemes are mainly considered for database integrity Generally, the watermarking for integrity verification is called fragile watermarking a compared to robust watermarking for copyright protection.

Although most of watermarking research has focused on robustness, capacity and perceptibility issues, it has been recently acknowledged that security aspects are als (if not even more) important for many secure applications such as copy control, ownershi verification, and authentication. Recognizing that the development of secur watermarking schemes, and even that the exact definition of what security means in watermarking context, is still in its infancy.

This is increasingly important in many applications where relationa databases are publicly available on the Internet. For example, to provide convenient acces to information for users, governmental and public institutions are increasingly required t publish their data on the Internet. The released data are usually in tabular form. They ma be statistical data produced by Census Bureau demographic surveys and Federal agencie such as National Center for Education Statistics and Energy Information Administratio

they may also be databases released by the Department of Motor Vehicles and Healt
Maintenance Organizations. In these cases, all released data are public; what is critical fo
the owner of the data is to make sure that the released data are not tampered with.

Another application is the edge computing where databases are distributed t
edge servers that perform data processing on behalf of the central server. Since the edg
servers may not be trusted, to ensure the relational databases are not modified by the edg
servers, the central server may need to check the integrity of the relational database
regularly.

Recently, some researchers have recognized the importance of watermarkin
databases and proposed some watermarking schemes to protect relational databases
However, these schemes are robust schemes, which are designed for copyright protectio
Though it is very important to protect the ownership of databases, sometimes, it is not
big issue about others making copies of databases. The relational databases should b
authentic and any modifications made should be detected or recovered.

1.2 WATERMARKING

Watermarking is the process of embedding information into a data in a wa
that is difficult to remove. Watermarking provides a communication channel between tw
authorized persons. Watermark security refers to the inability of a third person to acces
this channel. Watermarking ensures adversaries cannot emit and decode hidden bits, o
destroy this channel. The objective is to attach ownership or other descriptive informatio
to the data in a way that is difficult to remove. It also is possible to use hidden embedde
information as a means of covert communication between individuals.

Depending on the application, watermarking techniques can be classified int
two main categories; ***Robust and Fragile Watermarking Techniques.***

ROBUST WATERMARKING

Robust watermarking is mainly used for copy right protection and fingerprinting applications, in which the goal of watermark is to sustain under all kinds of attacks that intend to remove the watermark while preserving the perceptual quality of the original media. In a robust watermarking scheme, the embedded watermark should be robust against attacks which aim at removing the watermark or making it undetectable.

FRAGILE WATERMARKING

Watermarking for integrity verification is called fragile watermarking. This is used for data authentication and is sensitive to any kind of processing that may occur. A fragile watermark is very sensitive and is designed to detect every possible change in a marked image, but in most multimedia applications, minor data modifications are acceptable as long as the content is authentic. The embedded watermark should be fragile to modifications so as to detect and localize or even recover the modifications. Most of the fragile watermarking scheme studied in the last few years, were about multimedia watermarking. Most of them focus on digital images some have been extended to digital video, and audio data.

A **semi-fragile** watermark is robust to acceptable content preserving manipulations such as lossy compression while fragile to malicious distortions such as content modifications.

1.3 WATERMARKING SECURITY

Watermark security refers to the inability by unauthorized users to have access to the raw watermarking channel. Security is not robustness. The former has a broader scope than the latter as it not only deals with watermark removal but also with unauthorized embedding and detection.

Moreover security deals only with intentional attacks, whereas robustnes measures the impact of classical content transformations on the detect ability of th watermark. In the context of robustness, it is immaterial whether such transformations ar intentional or not as a rule, they are assumed to consist of common image or signa processing operations such as compression, filtering, cropping, or noise addition. Thi allows making a clear distinction between the fields of security and robustness.

Security deals with malicious attacks where the adversary has goo knowledge of the watermark technique being used. The work can be divided into tw phases. First, it is necessary to gain some knowledge about the watermarking techniqu and its vulnerabilities. During this learning phase, the attacker accumulates enoug information to find a complete break in the technique. Then, he can remove the protectiv watermark from the content for whatever purpose (e.g., privacy), or can illegally embe or decodes a message with a high degree of reliability. In a security analysis, th watermarking technique is thus used in a hostile environment where the adversary i trying to hack the system.

1.4 APPLICATIONS

Watermarking may be used for a wide range of applications, such as:

- Copyright protection
- Source tracking_(different recipients get differently watermarked content)
- Broadcast monitoring (television news often contains watermarked video fror international agencies)
- Covert communication

1.5 RELATIONSHIP WITH CRYPTOGRAPHY

Although watermarking pertains to signal processing, it is also related t computer security; hence, the comparison with cryptography must be properly treate There are many differences between cryptography and robust watermarking technique beginning from the very objective of each technology. While in the worst case, as far as encryption is concerned, the goal is to make the semantic of a communication not understandable from a possible opponent assuming that no deterioration of the message carrier happens, in the latter case the aim is to protect the hidden communication itself from possible deterioration of the channel (i.e. of the cover data). Anyway, an investigation on how evaluation of cryptographic algorithms is preceded is compelling. That approach is borrowed for understanding the security issues of the watermarking tool. In particular, as it is done for cryptography since Di and Hellmann's article, the security analysis is driven by the data the opponent observes once the embedder starts producing watermarked contents.

Only watermarked content: The attacker can only have access to one or more watermarked documents.

Chosen watermarked content: The attacker can choose one or more (pretended) watermarked documents.

Original and watermarked pair: The attacker can have access to one or more pairs of original and corresponding watermarked documents.

Chosen original and watermarked pair: The attacker can choose one or more pairs of original and corresponding watermarked documents.

1.6 RELATIONAL DATABASES

A relational database matches data by using common characteristics found within the data set. The resulting groups of data are organized and are much easier for many people to understand. Relational databases are currently the predominant choice in storing data like financial records, medical records, personal information and manufacturing and logistical data. A *relation* is defined as a set of tuples that have the same attributes. A tuple usually represents an object and information about that object. Objects are typically physical objects or concepts. A relation is usually described as a table, which is organize into rows and columns. All the data referenced by an attribute are in the same domain an conform to the same constraints.

The relational model specifies that the tuples of a relation have no specific orde and that the tuples, in turn, impose no order on the attributes. Applications access data b specifying queries, which use operations such as *select* to identify tuples, *project* t identify attributes, and *join* to combine relations. Relations can be modified usin the *insert*, *delete*, and *update* operators. New tuples can supply explicit values or b derived from a query. Similarly, queries identify tuples for updating or deleting. It i necessary for each tuple of a relation to be uniquely identifiable by some combinatio (one or more) of its attribute values. This combination is referred to as the primary key.

PROBLEM OVERVIEW

CHAPTER 2

PROBLEM OVERVIEW

2.1 PROBLEM DEFINITION

Proving ownership rights on outsourced relational databases is a crucial issu in today internet-based application environment and in many content distributio applications. A mechanism is needed for proof of ownership based on the secur embedding of imperceptible watermark in relational data.

2.2 OBJECTIVE

To propose a fragile watermarking scheme to verify the integrity of relational database and to recover the true database in case of any modifications. Thi scheme proves the ownership based on secure embedding of imperceptible watermark i the relational data.

2.3 TRADITIONAL METHOD

To check the integrity of relational databases, an intuitive method is to us the traditional digital signature to detect the alterations of databases. A hash value i computed over a relational database and then is signed with the owner's private key. Th generated signature is then appended to the database or stored separately. Though thi method is very simple, there are some problems with it. First, the signature can only b used to verify whether the database has been modified or not; it cannot be used to localiz and recover the modifications. Second, for a very large database, the failure of integrit verification will render the whole database useless. Finally, if there is a need to mak some necessary modifications to the database, we have to compute a new signature an discard the previous one. Besides, it is computationally intensive to generate and verif the signatures.

Due to these problems, it is desirable that we have a fragile watermarking scheme for relational databases so that any modifications can be detected, localized and successfully recovered. In this way, even if some part of a relational database has been altered, after localizing these modifications, and then we only need to repair the modified data by recovering the true data from modified locations and in such way we ensure that the whole of the data set is still authentic and reliable. In addition, because the fragile watermarking scheme is private key based, its computational cost is obviously less than that of digital signature schemes.

2.4 RELATED WORKS

In recent years, there has been relatively few works on watermarking relational databases. Agrawal and Kiernan [2] present a robust watermarking scheme for databases. According to an embedding key, some bits of some attributes of some tuples are modified to embed watermark bits.

Li et al. [5, 6] further extend this scheme to embed multiple bits informatior instead of one bit information as in Agrawal and Kiernan's scheme into a relationa database so that potential illegal distributors can be tracked. Also, this scheme is claimec to be more robust since false negative and false positive detection rates are bounded.

Sion et al. [8] present a different approach to robust watermarking scheme fo databases. In their scheme, all tuples are securely sorted and divided into non-intersectin subsets. A single watermark bit is embedded into some tuples of a subset by modifyin the distribution of tuple values. Watermark bits are embedded repeatedly and an erro control coding scheme is employed to recover the embedded bits. This scheme is claimec to be robust against attacks such as data resorting and data transformations.

Another scheme is proposed by Gross-Amblard [4], where database instance with bounded degree Gaifman graph are watermarked while local queries are preserved They also show that the difficulty of query-preserving watermarking is linked to th informational complexity of sets defined by queries. However, they do not analyze the robustness of their scheme, which is a very important property of watermarking schemes for copyright protection.

Devanbu et al. [3] and Pang and Tan [7] present schemes that are based on the Merkle Hash Tree, where each tuple is treated as a leaf node and a verifiable B+ tree is constructed by adding signed digest for very attribute and for each leaf node recursively until the root node is reached. Verification objects are created for query operations to authenticate query results. Though these schemes can detect any modifications, they cannot localize the modifications. In addition, they have extra overhead for storing and maintaining the tree.

All current robust watermarking schemes for relational databases are designed for copyright protection. Guo et al .[1] present a fragile watermarking scheme that cannot only detect but also localize the modifications made to the database, it was the most complete scheme which has been introduced yet. In this scheme all tuples in a relational database are first divided into groups then by using a hash function Watermarks are embedded and verified group by group independently according to some security parameters. The embedded watermarks can detect and localize the modifications made to the database. In the worst case, the modifications can be narrowed down to tuples in a group.

2.5 EXISTING SYSTEM

All current watermarking schemes for relational databases are designed for copyright protection or localizing the distortions made to the database tables. But none of them can recover the true data from the modified cells in database tables.

2.6 PROPOSED SYSTEM

In this paper a novel fragile watermarking scheme is proposed to detect, localize and recover malicious modifications in relational databases. In the proposed scheme, all tuples in the database are first securely divided into groups. Then watermarks are embedded and verified group-by-group independently. By using the embedded watermark, the modifications made to the database are detected and localized and even the true data is recovered from the database modified locations. The experimental results show that this scheme is so qualified; i.e. distortion detection and true data recovery both are performed successfully.

SYSTEM REQUIREMENTS AND ANALYSIS

CHAPTER 3

SYSTEM REQUIREMENTS AND ANALYSIS

3.1 SYSYEM STUDY

System study deals with the process of defining the functioning of existin system. The advantages and disadvantages of the existing system are elaboratel discussed to prove the way for the proposed system. Then the proposed system is define for the problem and the advantages of the proposed system are also defined.

3.2 REQUIREMENTS AND ANALYSIS

During the system analysis phase, a thorough analysis is carried out to fin the various features that can be incorporated to make the system better than the existin one. The following are the objectives of conducting such an analysis:

P- 3617

- Understanding the existing system
- Evaluation of the feasibility of the developing system
- Evaluation of the cost of implementation
- Preparation of the system requirements specification

The programming language most suitable for the development of the system and th minimum hardware requirements for the system are also identified.

3.3 PROJECT DEFINITION

This system is motivated by large problems arising in applications suc as copyright protection, integrity checking, and fingerprinting. The main objective of th system is to maintain the ownership and integrity of the relational database of the owner In this scheme, all tuples in a relational database are first divided into groups by using secret key. Then Watermarks are embedded in all groups and then watermark verificatio is done independently in each group. The embedded watermarks cannot only detect bu also localize the modifications made to the database, and even recover the true data fron modified cells in database tables. Since the watermarking process will inevitabl introduce small distortions to a relational database, the relational database to b watermarked has numerical attributes which can tolerate small changes.

3.4 PROJECT PLAN

3.4.1 Requirement analysis

The requirement phase deals with the sequence of activities i producing the software requirement specification (SRS) document. It is ensured that al the requirements of the software are elicited and analyzed. In other words the needs of th system are analyzed.

3.4.2 Problem analysis

In the problem analysis phase, the current system is analyzed and th changes to be made in the proposed system are decided upon.

3.4.3 Designing

Designing aims at how to satisfy the needs of the system. The differen modules of the system and the interaction between these modules to produce the desire functionality are identified. During the detailed design the internal logic of each modul and their algorithmic design are specified. The major and important decisions are made i this phase.

3.4.4 Coding

Coding is the process of translating the design into code. The cod developed must be understandable and consistent. Well-written code can reduce testin and maintenance effort.

3.4.5 Testing

Testing is done to check whether the requirements of the user are met. It involves checking the functionality of each module as per the design document.

3.5 FEASIBILITY

3.5.1 FEASIBILITY STUDY

The main purpose of the feasibility study is to consider all factors associatec with the project, and to yield a desirable result in the given time and other resources During the Feasibility Study stage, the project's goals, parameters and restraints are agreed upon with the client including project budget and rules for its adjustment, projec time frame, and conceptual problem solution. The study is conducted quite exhaustively by exploring many factors related to the project as

- Economic feasibility
- Technical feasibility
- Behavioral feasibility

Economic feasibility

For any system if the expected benefits equal or exceed the expected costs the system can be judged to be economically feasible. In economic feasibility, cost benefi analysis is done in which expected costs and benefits are evaluated. Economic analysis is used for evaluating the effectiveness of the proposed system.

In economic feasibility, the most important is ***cost-benefit*** analysis. As the name suggests, it is an analysis of the costs to be incurred in the system and benefits derivable out of the system.

Technical feasibility

A technical feasibility study is an excellent tool for trouble-shooting and long-terr planning. In technical feasibility the following issues are taken into consideration.

- Whether the required technology is available or not.
- Whether the required resources are available

  Manpower- programmers, testers & debuggers

  Software and hardware

Behavioral feasibility

An estimate should project how strong reaction the user is likely to have toward the development of a computerized system. If the users do not have the commo knowledge regarding the computer installations and introduction of improvised system then special effort is required to educate, sell and train the on new ways.

3.6 SPECIFIC REQUIREMENTS

3.6.1 FUNCTIONAL REQUIREMENTS

3.6.1.1 Embedding

The watermark bits are embedded at the last two bits of the data using attribut and tuple watermarking.

Input: Original database, secret key, watermark bits.

Output: Embedded database.

3.6.1.2 Decoding

The original database is decoded from the embedded data.

Input: Embedded database.

Output: Original database, watermark bits.

3.6.1.3 Software specification

▪ Operating system	:	Windows 2000 and above
▪ Programming Language	:	C#
▪ Back end	:	SQL Server 2005

3.6.1.4 Hardware specification

▪ Processor \ System	:	Pentium IV
▪ Main Memory	:	2 GB RAM
▪ Hard Disk	:	80GB

3.6.2 PERFORMANCE REQUIREMENTS

3.6.2.1 Invisibity

The watermark bits calculated are hidden at the least significant and the nex least significant bits of the data by performing attribute watermarking and tupl watermarking respectively.

3.6.2.2 Security

For the security reason, the embedding key should be selected from a larg enough key space so that it is computationally infeasible for an attacker to guess the ke Only the one who has the knowledge of K and g can determine the group that the tuple belong to.

3.6.2.3 Robustness

Since tuples in a relational database are independent, it is very important t enforce some relationship between them so that the embedded watermarks and th extracted watermarks can be synchronized. The grouping and sorting function do n change tuples's physical positions in the table.

3.6.2.4 Invertibility

The system is invertible if the authorized users can decode the original relation database from the embedded data.

3.7 SYSTEM DESIGN

ENCODING PROCESS

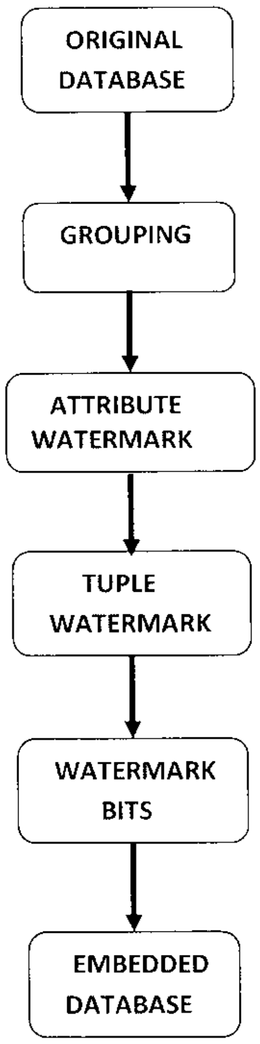


Fig. 1 Encoding process

Fig.1 represents the encoding process. The original database is grouped according tc number of groups and the secret key provided by the user. Attribute watermarking anc tuple watermarking are done individually to calculate the watermark bits to be embeddec for each group and the bits are embedded at the last two bits of each data.

DECODING PROCESS

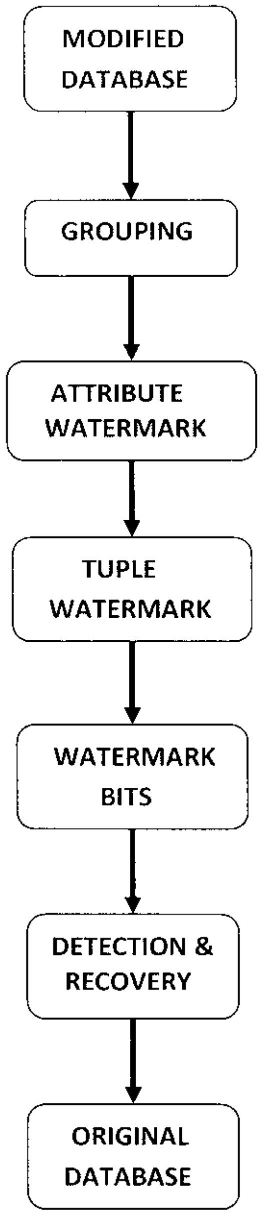


Fig. 2 Decoding Process

Fig.2 represents the decoding process. Grouping of the modified database is the same as is done in encoding process. The attribute and tuple watermarking are performed calculate the watermark bits and these bits are compared with the actual watermark bi embedded in the data. If there is any mismatch it denotes that the data has been modifie The modified data is localized and recovered back to obtain the original database.

MODULES

CHAPTER 4

MODULES

4.1 WATERMARK ENCODING

4.1.1 GROUPING

The relational database has a primary key P and y attributes, and is denoted by T(P,A1,A2, . . . ,Ay). All attributes are numeric. All tuples are first divided into groups according to the number of groups g, the hash value of a embedding key K and their primary key. Here XOR function is used as the hash function. For the security reason, the embedding key should be selected from a large enough key space so that it is computationally infeasible for an attacker to guess the key. Each group consists of an average of v number of tuples and they are sorted based on their primary key.

4.1.2 EMBEDDING

The embedding subroutine mainly consists of two parts:

Attribute watermark embedding.

Tuple watermark embedding.

Attribute watermark embedding.

Attribute watermark W1 consists of y watermarks of length v. The hash value is generated according to a message authentication code and the same attribute of all tuples in the group. The watermark bits are embedded at the least significant bit of each data. This scheme is able to embed the extracted attribute watermark from j-th column of group (W j1 watermark), in another column of group (p(j)-th column of group). This solution is required for true data recovery process.

Tuple watermark embedding

Tuple watermark W2 consists of v watermarks of length y. The hash value is generated according to the same message authentication code and all attribute values of the same tuple. The watermark bits are embedded at the next least significant bi of each data.

For generating the hash value, XOR operator is used as a hash functio instead of other miscellaneous hash function. This operator has heredity nature an therefore this scheme is able to recover the true data from the modified cells of databas table.

4.2 WATERMARK DECODING

4.2.1 DETECTION & LOCALIZING

As in watermark embedding, all tuples are divided into groups and each group i verified independently. For each group, we construct two verification vectors V1 = (V11 V2 1 ... Vy1) and V2 = (V12, V32... Vv2) where Vj1 denotes the verification result fo attribute watermark, and Vi2 denotes the verification result for tuple watermark. Eac element of the vectors is either true or false depending on whether the embedde watermark matches the related extracted watermark. If the two matches, the vector is true in both attribute and tuple watermarking otherwise, it is false.

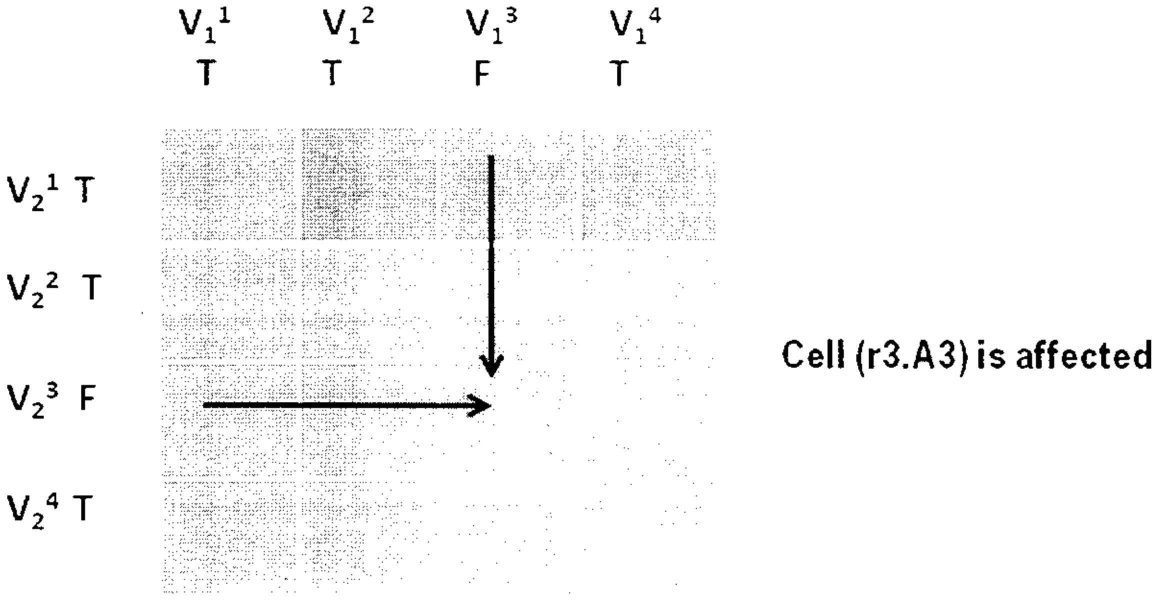


Fig. 3 An illustration of verification results in a cell

From the two vectors V1 and V2, any modifications made to tuples can be detecte and localized in a group. Fig. 3 shows the detection and localization of a modified data i the cell (r3.A3). 'False' denotes that the particular row or column has been modified.

4.2.2 RECOVERY

Other advantage of this proposed scheme is the ability to recover the tru data from modified cells. After detection and localization the modified cells, this schem can recover the true data. The XOR operator is used to recover the original value o modified cell which retrieves the original data back so that the original database i obtained.

EXPERIMENTAL RESULTS

CHAPTER 5

EXPERIMENTAL RESULTS

The security of proposed scheme is evaluated to recover the true. In this evaluatior the proposed scheme is applied as an application to a database table which has beer distorted. This scheme is run on the modified table in order to recover the true data. Thi work is done on several tables with the parameter (v=50, 100, 150, 200, 250, 300).

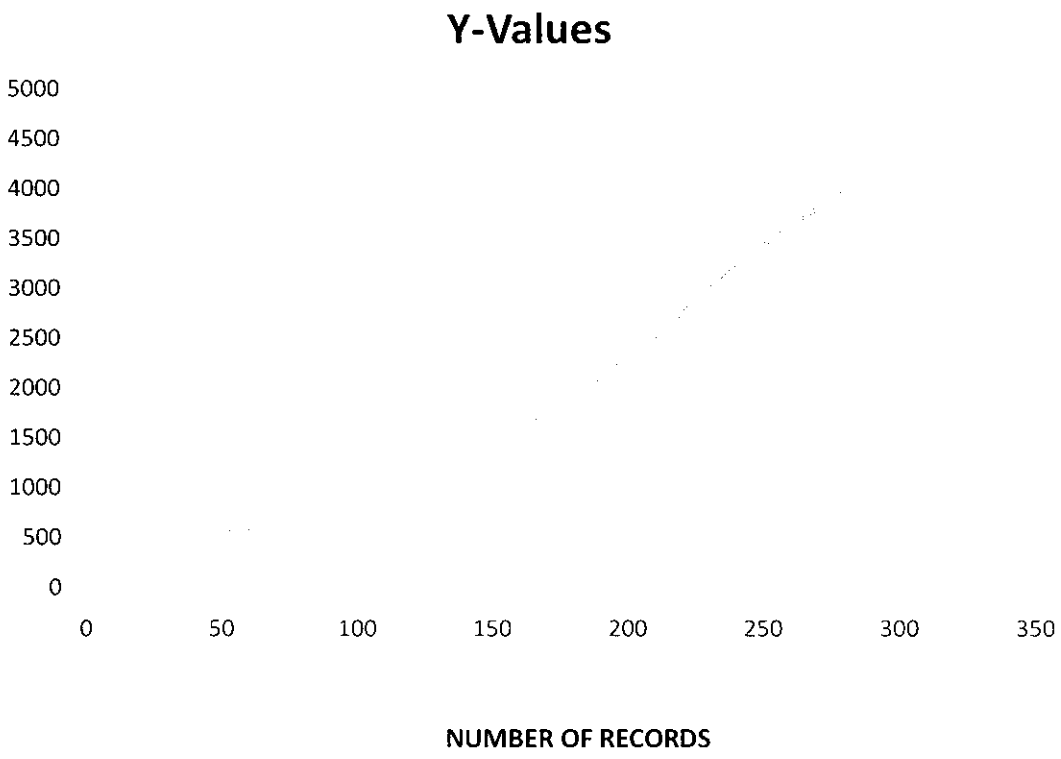


Fig 4: Response time in detecting the erroneous data

Fig 4 represents the response time in detecting the modified data in the database. Th time increases linearly as the number of tuples increases.

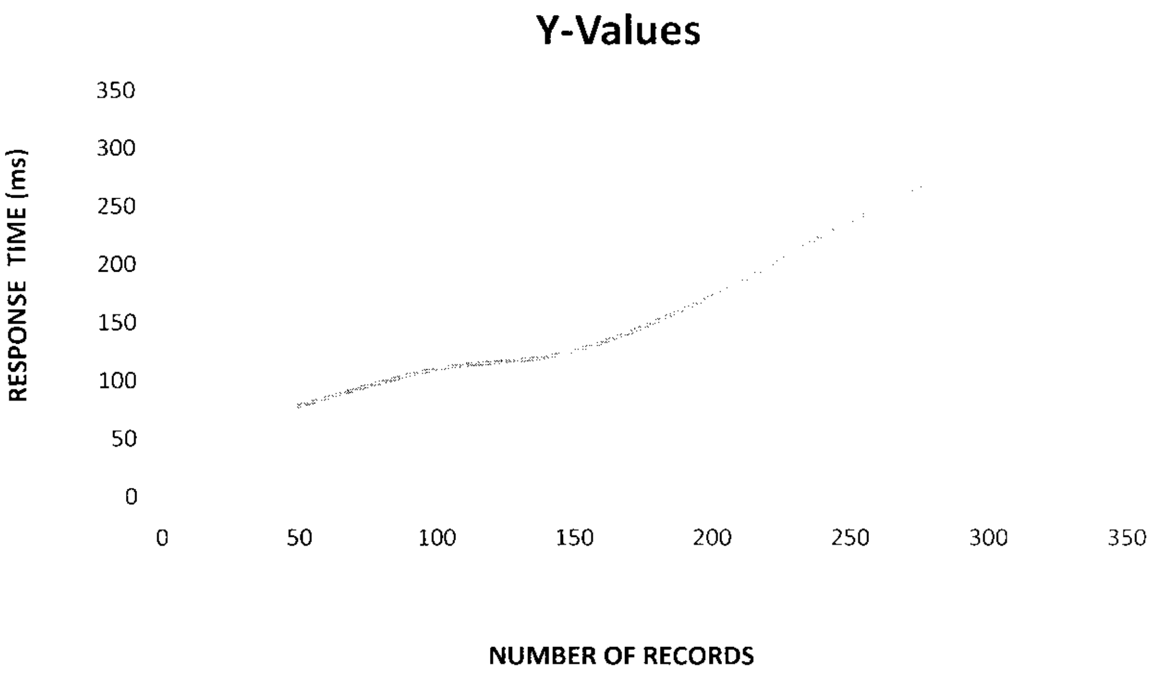


Fig 5: Response time in recovering the original data

Fig 5 represents the response time in recovering the original data in the database. Th time increases linearly as the number of tuples increases.

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

A fragile watermarking scheme is applied for relational databases. The watermark
are embedded into a relational database on the group basis under the control of a secu
embedding key. The embedded watermarks form a watermark grid which can detect an
localize any modifications made to the database and also be able to recover true data fro
modified cells. Experimental results showed that proposed scheme is secure and true da
recovery failure probability is very teeny. Security analysis showed that it is very difficu
for an attacker to modify the database without affecting the embedded watermarks. Th
system has been tested with various samples. The performance of the system is very goo

6.2 FUTURE ENHANCEMENT

Future work will focus on designing a watermarking scheme that can embe
watermarks to non-numeric attributes. For this purpose there are two solutions.

- The first solution is to reform the structure of hash function so it can accept nc
numeric inputs.
- The ASCII values of the characters can be used for the computation of the has
function
- The second solution would be another mechanism instead of using a hash function

APPENDIX

CODING

MAIN FORM.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.IO;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace WindowsApplication9
{
  public partial class Main : Form
  {
    public Main()
    {
      InitializeComponent();

    }
    private void Main_Load(object sender, EventArgs e)
    {
     button1.Visible = false;
      button3.Visible = false;
      label2.Visible = false;
      label3.Visible = false;
      label5.Visible = false;
      label4.Visible = false;
      comboBox1.Visible = false;
      comboBox2.Visible = false;
      button2.Visible = false;
    }
    private void button2_Click(object sender, EventArgs e)
    {
      String connString = "Data Source=HEMA-PC\\SQLEXPRESS;Integrated
     Security=True";
      using (SqlConnection sqlConx = new SqlConnection(connString))
  {
    sqlConx.Open();
    DataTable tblDatabases = sqlConx.GetSchema("Databases");
    sqlConx.Close();
    comboBox1.Items.Clear();
    foreach (DataRow row in tblDatabases.Rows)
    {
                 comboBox1.Items.Add(row["database_name"]);

    }
  }
 }
private void button1_Click(object sender, EventArgs e)
{
   object cbox = comboBox1.SelectedItem;
  String connString = "Data Source=HEMA-PC\\SQLEXPRESS;Initial Catalog=" +
                                 cbox + ";Integrated Security=True";
   using (SqlConnection sqlConx = new SqlConnection(connString))
   {
     sqlConx.Open();
     DataTable tblDatabases = sqlConx.GetSchema("Tables");
     foreach (DataRow row in tblDatabases.Rows)
     {
            string tbl = row["table_name"].ToString();
       if (tbl.EndsWith("wat"))
       {
              string drop = "drop table " + tbl + "";
         SqlCommand cmd = new SqlCommand(drop, sqlConx);
         cmd.ExecuteNonQuery();
       }
     }
     sqlConx.Close();
   }
   object com1 = comboBox1.SelectedItem;
   object com2 = comboBox2.SelectedItem;
   string cbox1 = com1.ToString();
   string cbox2 = com2.ToString();
   Encoding en = new Encoding(cbox1,cbox2);
   en.Show();
 }
```

```
    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
       object cbox = comboBox1.SelectedItem;
       String connString = "Data Source=HEMA-PC\\SQLEXPRESS;Initial Catalog=" +
                                      cbox + ";Integrated Security=True";

      using (SqlConnection sqlConx = new SqlConnection(connString))
       {
          sqlConx.Open();
          DataTable tblDatabases = sqlConx.GetSchema("Tables");
          sqlConx.Close();
          comboBox2.Items.Clear();
          foreach (DataRow row in tblDatabases.Rows)
          {
                   string tbl=row["table_name"].ToString ();
            if (!tbl.EndsWith("wat")&&(!tbl.StartsWith("mod")))
            {
               comboBox2.Items.Add(row["table_name"]);
            }
          }

        }
      }
   }
}
```

ENCODING FORM.cs

```
namespace WindowsApplication9
{
   public partial class Encoding : Form
   {
      string dbase;
      string tab;
      string gtab;
      public Encoding(string db,string tb)
      {
```

```
  InitializeComponent();
  dbase = db;
   tab = tb;
}
string conn1;

private void Encoding_Load(object sender, EventArgs e)
{
   conn1 = "Data Source=HEMA-PC\\SQLEXPRESS;Initial Catalog=" + dbase + "
                   ;Integrated Security=True";
   gtab = string.Concat(tab, "wat");
   string create = "select * into " + gtab + " from " + tab;
   using (SqlConnection sqlconx1 = new SqlConnection(conn1))
   {
      sqlconx1.Open();
      SqlCommand cmddb = new SqlCommand(create, sqlconx1);
      cmddb.ExecuteNonQuery();
      string alter = "alter table " + gtab + " add groups int default (0)";
      SqlCommand cmddb1 = new SqlCommand(alter, sqlconx1);
      cmddb1.ExecuteNonQuery();
      sqlconx1.Close();
      string a = "Select * From ";
      a = string.Concat(a, tab);
      SqlDataAdapter adp4 = new SqlDataAdapter(a,conn1);
      DataSet ds4 = new DataSet();
      String b = "\"";
      String c = "\"";
      b = string.Concat(b, tab);
      b = string.Concat(b, c);
      adp4.Fill(ds4, b);
      dataGridView1.Visible = false;
   }
 }

 private void button1_Click(object sender, EventArgs e)
 {
    int g,w;
    sbyte s;
    s =System .Convert .ToSByte  ( textBox1.Text);
    g =System .Convert .ToInt16 ( textBox2.Text);
```

```
string a = "Select * From ";
a = string.Concat(a,tab);//stu
SqlDataAdapter adp = new SqlDataAdapter(a,conn1);
DataSet ds = new DataSet();
String b = "\"";
String c = "\"";
b = string.Concat(b, tab);
b = string.Concat(b, c);
adp.Fill(ds, b);
w = ds.Tables[0].Rows.Count;
MessageBox.Show("Total records: " +w.ToString());
sbyte pk;
int[] k = new int[w];
int i = 0;
foreach (DataRow drt in ds.Tables[0].Rows)
{
   string   st = drt[0].ToString () ;
   pk =System .Convert .ToSByte  (st );
   int  hval = s ^ pk ;
   k[i] = hval % g;
   i++;
}
string m = "Select * From ";
string a1 = string.Concat(m, gtab);
SqlDataAdapter adpw = new SqlDataAdapter(a1, conn1);
DataSet dsw = new DataSet();
String b1 = "\"";
String c1 = "\"";
b1 = string.Concat(b1, gtab);
b1 = string.Concat(b1, c1);
adpw.Fill(dsw, b1);
int w1 = dsw.Tables[0].Columns.Count;
int ss;
for (int u = 0; u < k.Length; u++)
{
  dsw.Tables[0].Rows[u]["groups"] = k[u];
  ss = k[u];
  int pno = System.Convert.ToInt16( dsw.Tables[0].Rows[u][0]);
  string t = dsw.Tables[0].Rows[u]["groups"].ToString();
```

```
      string update = "update " + gtab +" set groups= "+@ss+" where product_id=
                               "+pno+"  ";
      SqlConnection con=new SqlConnection(conn1);
      con.Open();
      SqlCommand cmd = new SqlCommand(update ,con);
      adpw.UpdateCommand = cmd;                        adpw.Update(dsw,b1 );
      adpw.Fill(dsw);
     }
    SqlDataAdapter adp3 = new SqlDataAdapter("Select * From "+gtab+" order by
                                    groups,product_id asc", conn1);
    DataSet ds3 = new DataSet();
    adp3.Fill(ds3, "gtab");
    label1.Visible = false;
    label2.Visible = false; ;
    textBox1.Visible = false;
    textBox2.Visible = false;
    button1.Visible = false;
    dataGridView1.Visible = true;
    dataGridView1.DataSource = ds3.Tables[0].DefaultView;
    SqlDataAdapter adpi = new SqlDataAdapter("Select * from owner", conn1);
    SqlCommandBuilder cbdi = new SqlCommandBuilder(adpi);
    DataSet dsi = new DataSet();
    adpi.Fill(dsi, "owner");
    DataRow dri = dsi.Tables["owner"].NewRow();
    dri["skey"] = textBox1.Text;
    dri["groups"] = textBox2.Text;
    dsi.Tables["owner"].Rows.Add(dri);
    adpi.Update(dsi, "owner");
  }

 private void button2_Click(object sender, EventArgs e)
  {
    Embedding emb = new Embedding(textBox1.Text, textBox2.Text, dbase, gtab);
    emb.Show();

  }
 }
}
```

EMBEDDING FORM.cs

```
namespace WindowsApplication9
{
  public partial class Embedding : Form
  {
      string dbase;
      string tab;
      string taba;
      string tabt;
      string tabw ;

        public Embedding(string i,string j,string db,string tb)
     {
            InitializeComponent();
            dbase = db;
            tab = tb;
            taba=  string.Concat("att",tab );
            tabt = string.Concat( "tup",tab);
            tabw = string.Concat( "emb",tab);
            textBox1.Text = j;
            textBox2.Text = i;
     }
    string con;

     private void Embedding_Load(object sender, EventArgs e)
     {
              con = "Data Source=HEMA-PC\\SQLEXPRESS;Initial Catalog="+dbase+"
              ;Integrated Security=True";
        string create = "select * into " + taba + " from " + tab;
        string create1 = "select * into " + tabt + " from " + tab;
        using (SqlConnection sqlconx1 = new SqlConnection(con))
        {
                  sqlconx1.Open();
                  SqlCommand cmddb = new SqlCommand(create, sqlconx1);
                  SqlCommand cmddb1 = new SqlCommand(create1, sqlconx1);
                  cmddb.ExecuteNonQuery();
                  cmddb1.ExecuteNonQuery();
                  sqlconx1.Close();
        }
    label1.Visible = false;
    label2.Visible = false;
    textBox2.Visible = false;
    textBox1.Visible = false;
}

  private void button1_Click(object sender, EventArgs e)
{
        fnc(taba);
   fnc(tabt);
      fnc(tab);
   MessageBox.Show("Watermark Embedding Completed");
}

  int fnc(string tbl)
{
   string a1 = "Select * From ";
   a1 = string.Concat(a1, tbl);
   string a = " order by groups,product_id asc ";
   a = string.Concat(a1,a);
   SqlDataAdapter adp = new SqlDataAdapter(a,con);
   DataSet ds = new DataSet();
   String b = "\"";
   String c = "\"";
   b = string.Concat(b, tbl);
   b = string.Concat(b, c);
   adp.Fill(ds, b);
   using (new SqlCommandBuilder(adp))
   {
      int gp = System.Convert.ToInt32(textBox1.Text);
      // MessageBox.Show(rows.ToString());
      int count = 0, rows = 0, i = 0;
      int w = ds.Tables[b].Rows.Count;
      int k = System.Convert.ToInt32(textBox2.Text);
      int index = 0;
      int p = 0;
      for (int g = 0; g < gp; g++)//group count(2)
      {
         index = count;
         count = 0;
```

```
            for (i = 0; i < w; i++)//w=row count(4)
      {
                if (ds.Tables[b].Rows[i]["groups"].Equals(g))
        {
          count++;
          rows = count;
        }
      }
      p = p + index;
      if(string . Compare(tbl,taba)==0)
      {
                  int num = attwat(tbl, k, rows, p, adp, ds);

      }
      else if (string.Compare(tbl, tabt) == 0)
      {
            int tup = tupwat(tbl, k, rows, p, adp, ds);
      }
      else if (string.Compare(tbl, tab) == 0)
      {
       // MessageBox.Show("emb enter");
        dataGridView1.DataSource = ds.Tables[0].DefaultView;
        int num = attwat(tbl, k, rows, p, adp, ds);
        dataGridView1.DataSource = ds.Tables[0].DefaultView;
        int tup = tupwat(tbl, k, rows, p, adp, ds);
                dataGridView1.DataSource = ds.Tables[0].DefaultView;

              }
    }
      }
  return 0;
}

  int attwat(string tbl,int k, int rows,int p,SqlDataAdapter adpa,DataSet dsa)
{
  int kk = k;
  int q = p;
  int initial = p;
      int hval = 0;
  int res1;
```

```
String b = "\"";
String cd = "\"";
b = string.Concat(b, tbl);
b = string.Concat(b, cd);
int cols = dsa.Tables[b].Columns.Count;
int ro=dsa.Tables[b].Rows.Count;
string[] watbits = new string[cols ];
for (int j = 1; j < cols - 1; j++)//col<6
{
   k = kk;
   p = initial;
   int r;
   for (r = 0; r < rows; r++)
   {
      string st = dsa.Tables[b].Rows[p][j].ToString();
      int pk = System.Convert.ToInt32(st);
      string stsub = System.Convert.ToString(pk, 2);
      hval = k ^ pk ;
      k = hval;
      p++;
   }
       watbits[j] = extractbits(hval, rows);
 }
 for (int j = 1; j < cols - 1; j++)//col<6
 {
   k = kk;
   q = initial;
   int i;
         for ( i = 0;  i< rows; i++)
   {
      int result;
      string copy = null;
      int c = newatt(kk, j, cols);
      int tatt=System.Convert.ToInt32
                        (dsa.Tables[b].Rows[q][c].ToString());
                string   valat = System.Convert.ToString(tatt, 2);
      int len = valat.Length;
      if (valat.Length != 8)
         for (int y = len; y < 8; y++)
            valat  = valat.Insert(0, "0");
```

```
          int index = valat.Length - 1;
          char [] array=new char   [valat.Length ];
for(int u=0;u<valat.Length ;u++)
{
   if(u!=index )
   {
   array[u]= valat[u];
               }
   else
   {
                 copy = watbits[j];
      array[u]= copy[i];

   }
}
 string r=new string(array);
 result = System.Convert.ToInt32 (r,2);
       dsa.Tables[b].Rows[q][c] = result;
 string fieldname;
 string fieldname1;
 using (SqlConnection sqlconx1 = new SqlConnection(con))
 {
  sqlconx1.Open();
          SqlCommand cmd = new SqlCommand("Select * from "+@b+" ",
                                    sqlconx1);
  SqlDataReader reader = cmd.ExecuteReader();
  fieldname = reader.GetName(c);
  fieldname1 = reader.GetName(0);
  reader.Close();
 }
int pno = System.Convert.ToInt32(dsa.Tables[b].Rows[q][0]);
string up = "update "+@b+" set  "+@fieldname+"  = " + result  + "where
                     "+@fieldname1+" = " + pno  + "  ";
using (SqlConnection sqlconx1 = new SqlConnection(con))
{
  sqlconx1.Open();
  SqlCommand cmddb = new SqlCommand(up, sqlconx1);
  cmddb.ExecuteNonQuery();
  sqlconx1.Close();
}
```

```
     q++;
    }
  }
  dataGridView1.DataSource = dsa.Tables[b].DefaultView;
   return 0;
}



  int tupwat(string tbl,int k, int rows,int p,SqlDataAdapter adpa,DataSet dsa)
{
  int kk = k;
  int q = p;
  int initial = p;
  int hval = 0;
  int res1;
  String b = "\"";
  String cd = "\"";
  b = string.Concat(b, tbl);
  b = string.Concat(b, cd);
  int cols = dsa.Tables[b].Columns.Count;
  int ro = dsa.Tables[b].Rows .Count;
  string [] watbits = new string[ro];
  p = initial;
      for (int r = 0; r < rows; r++)
  {
    k = kk;
          for (int j = 1; j < cols - 1; j++)//col<6
    {
              string st = dsa.Tables[b].Rows[p ][j].ToString();
              int pk = System.Convert.ToInt32(st);
              hval = k ^ pk;
     k = hval;
    }
    watbits[p] = extractbits1(hval, cols - 2);
    p++;

       }

    int res2;
```

```
q = initial;
string copy = null;
for (int r = 0; r < rows; r++)
{
   k = kk;
    for (int j = 1; j < cols - 1; j++)
    {
                  int tatt = System.Convert.ToInt32(dsa.Tables[b]
                              .Rows[q][j].ToString());
                  string valat = System.Convert.ToString(tatt, 2);
                  int len = valat.Length;
                  if (valat.Length != 8)
                  for (int y = len; y < 8; y++)
         valat  = valat.Insert(0, "0");
           int index = valat.Length - 2;
                  char[] array = new char[valat.Length];
    for (int u = 0; u < valat.Length; u++)
    {
           if (u != index)
           {
               array[u] = valat[u];
           }
           else
           {
                  copy = watbits[q];
                              array[u] = copy[j-1];
           }
    }
    string r1 = new string(array);
    res2 = System.Convert.ToInt32(r1, 2);
             dsa.Tables[b].Rows[q][j] = res2;
    string fieldname;
    string fieldname1;
    using (SqlConnection sqlconx1 = new SqlConnection(con))
   {
     sqlconx1.Open();
     SqlCommand cmd = new SqlCommand("Select * from "+@b+" ",
                                    sqlconx1);
               SqlDataReader reader = cmd.ExecuteReader();
     fieldname = reader.GetName(j);
```

```
          fieldname1 = reader.GetName(0);
                   reader.Close();
        }
        int pno = System.Convert.ToInt32(dsa.Tables[b].Rows[q][0]);
                  string up = "update "+@b+" set  "+@fieldname+"  = " + res2  +
                              "where "+@fieldname1+" = " + pno  + "  ";
        using (SqlConnection sqlconx1 = new SqlConnection(con))
        {
          sqlconx1.Open();
          SqlCommand cmddb = new SqlCommand(up, sqlconx1);
          cmddb.ExecuteNonQuery();
          sqlconx1.Close();
        }
      }
        q++;

   }
   dataGridView1.DataSource = dsa.Tables[b].DefaultView;
   return 0;
  }
  int newatt(int kk, int j, int cols)
  {
          int atcol = ((kk + j) % (cols - 2)) + 1;
     return atcol;
  }
  string extractbits(int hval, int rows)
  {
     string w;
     string b = System.Convert.ToString(hval, 2);
     if (b.Length >= rows)
     {
               w = b.Substring(0, rows);
     }
     else
     {
       int m = rows - b.Length;
       w = string.Concat(b, extractbits(hval, m));
          }
     return w;
  }
```

```
      string extractbits1(int hval, int cols)
      {
         string w;
         string b = System.Convert.ToString(hval, 2);
         if (b.Length >= cols )
         {
            w = b.Substring(0, cols );

         }

         else
         {
            int m = cols - b.Length;
            w = string.Concat(b, extractbits(hval, m));
        }
         return w;
      }
   }
}
```

DETECTION FORM.cs

```
namespace WindowsApplication9
{
   public partial class Detection : Form
   {
      string dbase;
      string tab, taba;
      public Detection(string db, string tb)
      {
         InitializeComponent();
         dbase = db;
         tab = tb;
         taba = string.Concat("mod",tab );
      }
      string con;
      string fieldname;
      private void Detection_Load(object sender, EventArgs e)
      {
```

```
con = "Data Source=HEMA-PC\\SQLEXPRESS;Initial
            Catalog="+dbase+";Integrated Security=True";
using (SqlConnection sqlConx = new SqlConnection(con))
{
  sqlConx.Open();
  DataTable tblDatabases = sqlConx.GetSchema("Tables");
  foreach (DataRow row in tblDatabases.Rows)
  {
    string tbl = row["table_name"].ToString();
    if (tbl.StartsWith ("mod"+tab+""))
    {
      string drop="drop table mod"+tab+"";
      SqlCommand cmd = new SqlCommand(drop , sqlConx);
      cmd.ExecuteNonQuery();
    }
  }
  sqlConx.Close();
}
string create = "select * into " + taba + " from " + tab;//modstu
string alter = "alter table " + taba + " add groups int default (0)";
using (SqlConnection sqlconx1 = new SqlConnection(con))
{
  sqlconx1.Open();
  SqlCommand cmddb = new SqlCommand(create, sqlconx1);
  cmddb.ExecuteNonQuery();
  SqlCommand cmddb1 = new SqlCommand(alter, sqlconx1);
  cmddb1.ExecuteNonQuery();
  sqlconx1.Close();
  string a = "Select * From ";
  a = string.Concat(a, taba);//modstu
  SqlDataAdapter adp4 = new SqlDataAdapter(a, con);
  DataSet ds4 = new DataSet();
  String b = "\"";
  String c = "\"";
  b = string.Concat(b, taba);
  b = string.Concat(b, c);
  adp4.Fill(ds4, b);
  dataGridView1.DataSource = ds4.Tables[0].DefaultView;
}
}
```

```
private void textBox2_Enter(object sender, EventArgs e)
{
   SqlDataAdapter ade = new SqlDataAdapter("Select * from owner", con);
   SqlCommandBuilder cbe = new SqlCommandBuilder(ade);
   DataSet dte = new DataSet();
   ade.Fill(dte, "owner");
   DataColumn[] dc = new DataColumn[1];
   dc[0] = dte.Tables["owner"].Columns["skey"];
   dte.Tables["owner"].PrimaryKey = dc;
   DataRow dre = dte.Tables["owner"].Rows.Find(textBox1.Text);
   if (dre == null)
   {
      MessageBox.Show("No valid user");
      textBox1.Clear();
   }
   else
   {
      SqlDataAdapter ade1 = new SqlDataAdapter("Select * from owner where
                                    skey='" + textBox1.Text + "' ", con);
      DataSet dte1 = new DataSet();
      ade1.Fill(dte1, "owner");
      textBox2.Text = dte1.Tables["owner"].Rows[0]["groups"].ToString();
   }
}
private void button2_Click(object sender, EventArgs e)
{
   string a1 = "Select * From ";
   a1 = string.Concat(a1, taba  );//stumod
   string a = " order by groups ,product_id asc ";
   a = string.Concat(a1, a);
   SqlDataAdapter adp = new SqlDataAdapter(a, con);
   DataSet ds = new DataSet();
   String b = "\"";
   String c = "\"";
   b = string.Concat(b, taba );
   b = string.Concat(b, c);
   adp.Fill(ds, b);
   using (new SqlCommandBuilder(adp))
   {
      int gp = System.Convert.ToInt16(textBox2.Text);
```

```
        int count = 0, rows = 0, i = 0;
        int w = ds.Tables[0].Rows.Count;
        int k = System.Convert.ToInt16(textBox1.Text);
        int cols = ds.Tables[0].Columns.Count;
        int[] v1 = new int[cols];
        int[] v2 = new int[w+1];
        string[,] location = new string[w, cols];
        int index = 0;
        string[,] op = new string[w, cols];
        int p = 0;
        for (int g = 0; g < gp; g++)//group count(2)
        {
          index = count;
          count = 0;
                  for (i = 0; i < w; i++)//w=row count(4)
          {
            if (ds.Tables[0].Rows[i]["groups"].Equals(g))
            {
              count++;
              rows = count;//rows=1 g=1
            }
          }
          p = p + index;
          op = attwat(k, rows, p, adp, ds, location,v1,v2);
          Recovery.passValuesMethod(op);
          int size = op.Length;
        }
        dataGridView1.DataSource = ds.Tables[b].DefaultView;
      }
      MessageBox.Show("Detection Completed");
      Recovery rec = new Recovery(dbase, tab);
      rec.Show();
    }
    string[,] attwat(int k, int rows, int p, SqlDataAdapter adpa, DataSet dsa, string[,]
                                                      location, int[] v1, int[] v2)
    {
      int c;
      int kk = k;
      int q = p;
      int initial = p;
```

```
String b = "\"";
String cd = "\"";
b = string.Concat(b, taba);//stumod
b = string.Concat(b, cd);
int hval = 0;
int res1;
int cols = dsa.Tables[b].Columns.Count;
string watbits;
for (int j = 1; j < cols - 1; j++)//col<6
{
   k = kk;
   p = initial;
   for (int r = p; r < (p + rows); r++)
   {
      string st = dsa.Tables[b].Rows[r][j].ToString();
      int pk = System.Convert.ToInt32(st);
      hval = k ^ pk;
      k = hval;
   }
   watbits = extractbits(hval, rows);
   string watbits1 = null;
   char[] array = new char[rows];
   int l = 0;
   q = initial;
   for (int i = q; i < (q + rows); i++)
   {
      string a1 = "Select * From ";
      string tbl = string.Concat(tab, "wat");
      a1 = string.Concat(a1, tbl);
      string a = " order by groups,product_id asc ";
      a = string.Concat(a1, a);
      SqlDataAdapter adp1 = new SqlDataAdapter(a, con);
      DataSet ds1 = new DataSet();
      String b1 = "\"";
      String c1 = "\"";
      b1 = string.Concat(b1, tbl);
      b1 = string.Concat(b1, c1);
      adp1.Fill(ds1, b1);
      int result;
      c = newatt(kk, j, cols);
```

```
        int tatt = System.Convert.ToInt32
                                    (ds1.Tables[b1].Rows[i][c].ToString());
        string valat = System.Convert.ToString(tatt, 2);
        int len = valat.Length;
        if (valat.Length != 8)
            for (int y = len; y < 8; y++)
                valat = valat.Insert(0, "0");
        int index = valat.Length - 1;
        array[l] = valat[index];
        l++;
    }
    watbits1 = new string(array);
    int comp = 0;
    comp = string.Compare(watbits, watbits1);
    if (comp != 0)
    {
        v1[j] = 0;
    }
    else
    {
        v1[j] = 1;
    }
}
p = initial;
for (int r = p; r < (p + rows); r++)
{
    k = kk;
    for (int j = 1; j < cols - 1; j++)//col<6
    {
        string st = dsa.Tables[b].Rows[r][j].ToString();
        int pk = System.Convert.ToInt32(st);
        hval = k ^ pk;
        k = hval;
    }
    string watbitst = extractbits1(hval, cols - 2);
    int res2;
    char[] arrayt = new char[cols];
    string watbitst1 = null;
    int s = 0;
    for (int j = 1; j < cols - 1; j++)
```

```
{
   string a1 = "Select * From ";
   string tbl = string.Concat(tab, "wat");
   a1 = string.Concat(a1, tbl);
   string a = " order by groups,product_id asc";
   a = string.Concat(a1, a);
   SqlDataAdapter adp = new SqlDataAdapter(a, con);
   DataSet ds = new DataSet();
   String b1 = "\"";
   String c1 = "\"";
   b1 = string.Concat(b1, tbl);
   b1 = string.Concat(b1, c1);
   adp.Fill(ds, b1);
   int tatt = System.Convert.ToInt32
                              (ds.Tables[b1].Rows[r][j].ToString());
   string valat = System.Convert.ToString(tatt, 2);
   int len = valat.Length;
   if (valat.Length != 8)
      for (int y = len; y < 8; y++)
         valat = valat.Insert(0, "0");
   int index = valat.Length - 2;
         arrayt[s] = valat[index];
   s++;
         }
  //p++;
  watbitst1 = new string(arrayt);//stuwat
  int comp1 = 0;
  comp1 = string.Compare(watbitst, watbitst1);
  if (comp1 != 0)
  {
     v2[r] = 0;
  }
  else
  {
     v2[r] = 1;
  }
}
int n;
for ( n = initial; n < (initial + rows); n++)//row
{
        for (int m = 1; m < cols - 1; m++)//column
        {
          if (v1[m] == 0 || v2[n] == 0)
          {
            string a1 = "Select * From ";
            a1 = string.Concat(a1, taba);//stumod
            string a = " order by groups , product_id asc ";
            a = string.Concat(a1, a);
            SqlDataAdapter adpr = new SqlDataAdapter(a, con);
            DataSet dsr = new DataSet();
            String bq = "\"";
            String cq = "\"";
            bq = string.Concat(bq, taba);
            bq = string.Concat(bq, cq);
            adpr.Fill(dsr, bq);
            string result = dsr.Tables[bq].Rows[n][m].ToString();
            location[n, m] = result;
          }
        }
      }
      for (int x = 0; x < (initial+rows); x++)
      {
          for (int y = 1; y <cols  ; y++)
        {
            if (location[x, y] != null)
            {
              dataGridView1.Rows[x].Cells[y].Style.BackColor =
              System.Drawing.Color.IndianRed;
            }
          }
      }
      dataGridView1.DataSource = dsa.Tables[0].DefaultView;
      return location;
}
string extractbits1(int hval, int cols)
{
  string w;
  string b = System.Convert.ToString(hval, 2);
  if (b.Length >= cols)
  {
      w = b.Substring(0, cols);
    }
    else
    {
      int m = cols - b.Length;
      w = string.Concat(b, extractbits(hval, m));
    }
    return w;
}
int newatt(int kk, int j, int cols)
{
   int atcol = ((kk + j) % (cols - 2)) + 1;
   return atcol;
}
string extractbits(int hval, int rows)
{
   string w;
   string b = System.Convert.ToString(hval, 2);
   if (b.Length >= rows)
   {
      w = b.Substring(0, rows);
   }
   else
   {
      int m = rows - b.Length;
      w = string.Concat(b, extractbits(hval, m));
   }
   return w;
 }
 private void button1_Click_1(object sender, EventArgs e)//split
 {
    int g, w;
    sbyte s;
    s = System.Convert.ToSByte(textBox1.Text);//5
    g = System.Convert.ToInt16(textBox2.Text);//2
     string a1 = "Select * From ";
    a1 = string.Concat(a1,tab  );//stu
    SqlDataAdapter adp = new SqlDataAdapter(a1,con);
    DataSet ds = new DataSet();
    String b = "\"";
String c = "\"";
b = string.Concat(b, tab);
b = string.Concat(b, c);
adp.Fill(ds, b);
w = ds.Tables[b].Rows.Count;
MessageBox.Show("Total records: " + w.ToString());
sbyte pk;
int[] k = new int[w];
int i = 0;
foreach (DataRow drt in ds.Tables[0].Rows)
{
   string st = drt[0].ToString();
   pk = System.Convert.ToSByte(st);
   int hval = s ^ pk;
   k[i] = hval % g;
   i++;
}
string m = "Select * From ";
string a2 = string.Concat(m, taba);//modstu
SqlDataAdapter adpw = new SqlDataAdapter(a2, con);
DataSet dsw = new DataSet();
String b1 = "\"";
String c1 = "\"";
b1 = string.Concat(b1, taba);
b1 = string.Concat(b1, c1);
adpw.Fill(dsw, b1);
int w1 = dsw.Tables[0].Columns.Count;
int ss;
for (int u = 0; u < k.Length; u++)//u<10
{
   dsw.Tables[b1].Rows[u]["groups"] = k[u];
   ss = k[u];
   int pno = System.Convert.ToInt16(dsw.Tables[b1].Rows[u][0]);
   string t = dsw.Tables[b1].Rows[u]["groups"].ToString();
   string fieldname;
   using (SqlConnection sqlconx1 = new SqlConnection(con))
   {
      sqlconx1.Open();
      SqlCommand cmd = new SqlCommand("Select * from " + @b1 + " ",
                                                      sqlconx1);
```

```
            SqlDataReader reader = cmd.ExecuteReader();
            fieldname = reader.GetName(0);
            reader.Close();
          }
            string update = "update " + taba + " set groups= " + @ss + " where
                                          "+@fieldname +" = " + pno + "  ";
          SqlConnection conx = new SqlConnection(con);
          conx.Open();
          SqlCommand cmd1 = new SqlCommand(update, conx);
          adpw.UpdateCommand = cmd1;
          conx.Close();
               adpw.Update(dsw,b1 );
          adpw.Fill(dsw);
          dataGridView1.DataSource = dsw.Tables[b1].DefaultView;
        }
      }
    }
}
```

RECOVERY.cs

```
namespace WindowsApplication9
{
    public partial class Recovery : Form
    {
      string dbase,tab;
      public Recovery(string db,string tb )
      {
        InitializeComponent();
        dbase = db;
        tab=tb;
      }
      public static string[,] myNewArray;
         string con;
      string [] tname =new string[5];
      string ratt,rtup;
      int pno;

         public static void passValuesMethod(string[,] myNamesArray)
      {
```

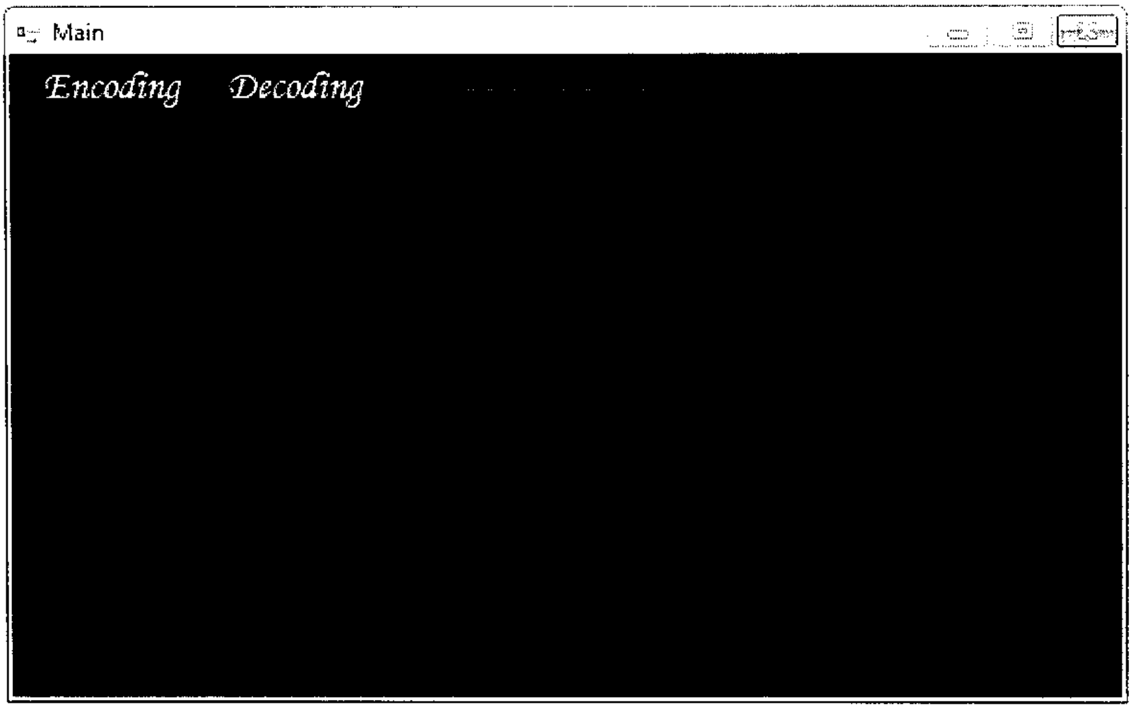
```
   myNewArray  = myNamesArray;
}
private void Recovery_Load(object sender, EventArgs e)
{
        con = "Data Source=HEMA-PC\\SQLEXPRESS;Initial
            Catalog="+dbase+";Integrated Security=True";
   using (SqlConnection sqlConx = new SqlConnection(con))
   {
      sqlConx.Open();
      DataTable tblDatabases = sqlConx.GetSchema("Tables");
      sqlConx.Close();
      foreach (DataRow row in tblDatabases.Rows)
      {
         string tbl = row["table_name"].ToString();
         if (tbl.StartsWith("att" + tab + ""))
         {
            ratt = row["table_name"].ToString();
         }
         else if (tbl.StartsWith("tup" + tab + ""))
         {
            rtup = row["table_name"].ToString();
         }
      }
   }
}
 private void button1_Click(object sender, EventArgs e)
 {
    string a1o = "Select * From ";
    a1o = string.Concat(a1o, tab );
    SqlDataAdapter adpo = new SqlDataAdapter(a1o, con);
    DataSet dso = new DataSet();
    String bo = "\"";
    String co = "\"";
    bo = string.Concat(bo, tab );
    bo = string.Concat(bo, co);
    adpo.Fill(dso, bo);
    int w = dso.Tables[bo].Rows.Count;
    int cols = dso.Tables[bo].Columns.Count;
    string[,] loc = (string[,])myNewArray.Clone();
    int len = myNewArray.Length;
```

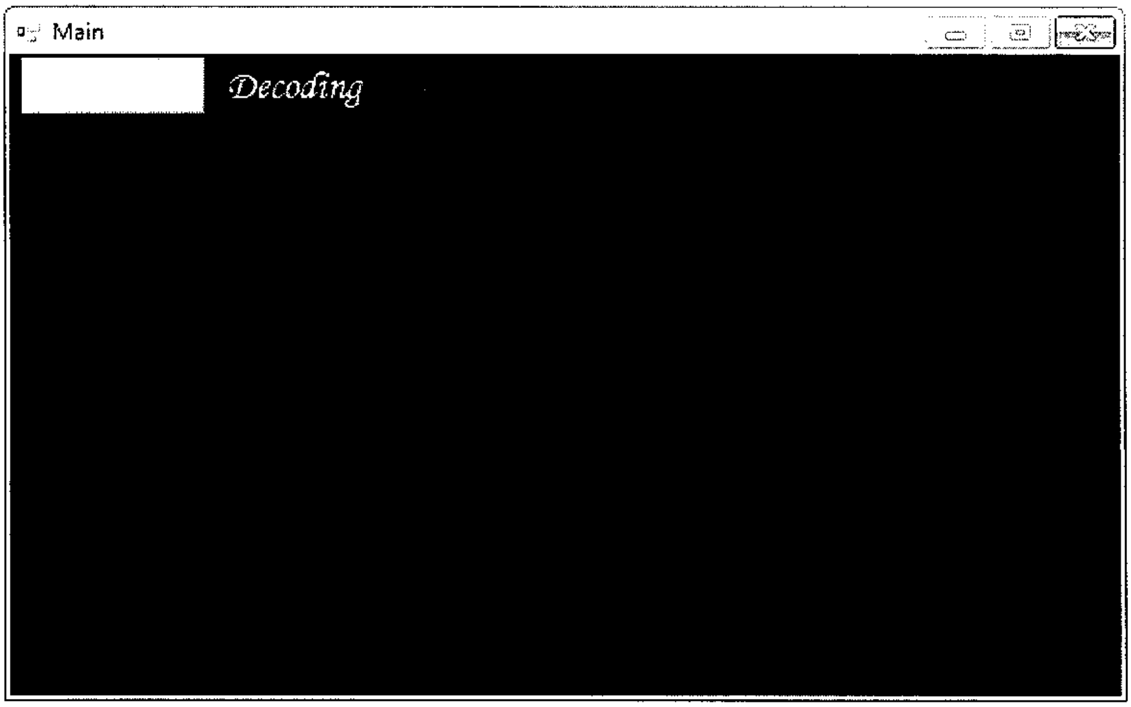
```
string strPosition = "";
foreach (string str in myNewArray)
{
   if (str != null)
   {
              for (int count1=0;count1 < myNewArray.GetLength(0);count1++)
     {
      for(int count2=1;count2 < myNewArray.GetLength(1);count2++)
       {
        if (myNewArray[count1, count2] == str)
        {
         strPosition = count1.ToString()+","+count2.ToString();
         string fieldname;
         string fieldname1;
         using (SqlConnection sqlconx1 = new SqlConnection(con))
         {
           sqlconx1.Open();
           SqlCommand cmd = new SqlCommand("Select * from " + @tab + " ",
                                        sqlconx1);
           SqlDataReader reader = cmd.ExecuteReader();
           fieldname = reader.GetName(count2);
           fieldname1 = reader.GetName(0);
           reader.Close();
         }
         string aval=fnc(count1, count2,fieldname1 );
         string tval=fnc1(count1, count2,fieldname1 );
         string final = string.Copy(aval);
         int n = final.Length - 1;
         final  = final.Remove(n);
         char  end=tval[tval.Length-1];
         final = final.Insert(n,end.ToString());
         int res = System.Convert.ToInt16(final , 2);
         string up = "update " + @tab + " set  " + @fieldname + "  = " + res +
                        "where " + @fieldname1 + " = " + pno + "  ";
                   string up1 = "update mod"+@tab+" set  " + @fieldname + "  =
                   " + res + "where " + @fieldname1 + " = " + pno + "  ";
         using (SqlConnection sqlconx1 = new SqlConnection(con))
         {
          sqlconx1.Open();
          SqlCommand cmddb = new SqlCommand(up, sqlconx1);
```

```
            cmddb.ExecuteNonQuery();
            SqlCommand cmddb1 = new SqlCommand(up1, sqlconx1);
            cmddb1.ExecuteNonQuery();
            sqlconx1.Close();
           }
          }
        }
      }
    }
  }
  string sv = "Select * From ";
  string  a1 = string.Concat("mod",tab);
  sv = string.Concat(sv,a1);
  string ab = " order by groups,product_id asc ";
  ab = string.Concat(sv, ab);
  SqlDataAdapter adp = new SqlDataAdapter(ab, con);
  DataSet ds = new DataSet();
  String b = "\"";
  String c= "\"";
  b= string.Concat(b, a1);
  b= string.Concat(b, c);
  adp.Fill(ds, b);
  dataGridView1.DataSource = ds.Tables[b];
  dataGridView1.Columns["groups"].Visible = false;
  for (int x = 0; x <w ; x++)
  {
    for (int y = 1; y <cols ; y++)
    {
      if (loc[x, y] != null)
      {
        dataGridView1.Rows[x].Cells[y].Style.BackColor =
               System.Drawing.Color.IndianRed;
      }
    }
  }
  MessageBox.Show("Recovery completed");
}

   string  fnc(int c1,int c2,string f)
{
     string a1 = "Select * From ";
     a1 = string.Concat(a1, ratt);//stumod
     string a = " order by groups ,product_id asc ";
     a = string.Concat(a1, a);
     SqlDataAdapter adp = new SqlDataAdapter(a, con);
     DataSet ds = new DataSet();
     String b = "\"";
     String c = "\"";
     b = string.Concat(b, ratt);
     b = string.Concat(b, c);
     adp.Fill(ds, b);
     string st = ds.Tables[b].Rows[c1 ][c2 ].ToString();
     pno = System.Convert.ToInt16(ds.Tables[b].Rows[c1][0]);
     int pk = System.Convert.ToInt32(st);
     string stsub = System.Convert.ToString(pk, 2);
     return stsub;
   }

     string  fnc1(int d1, int d2,string f)
   {
     string a1 = "Select * From ";
     a1 = string.Concat(a1, rtup);//stumod
     string a = " order by groups,product_id asc ";
     a = string.Concat(a1, a);
     SqlDataAdapter adp = new SqlDataAdapter(a, con);
     DataSet ds = new DataSet();
     String b = "\"";
     String c = "\"";
     b = string.Concat(b, rtup);
     b = string.Concat(b, c);
     adp.Fill(ds, b);
     string st = ds.Tables[b].Rows[d1][d2].ToString();
     int pk = System.Convert.ToInt32(st);
     string stsub = System.Convert.ToString(pk, 2);
     return stsub;
   }
  }
}
```

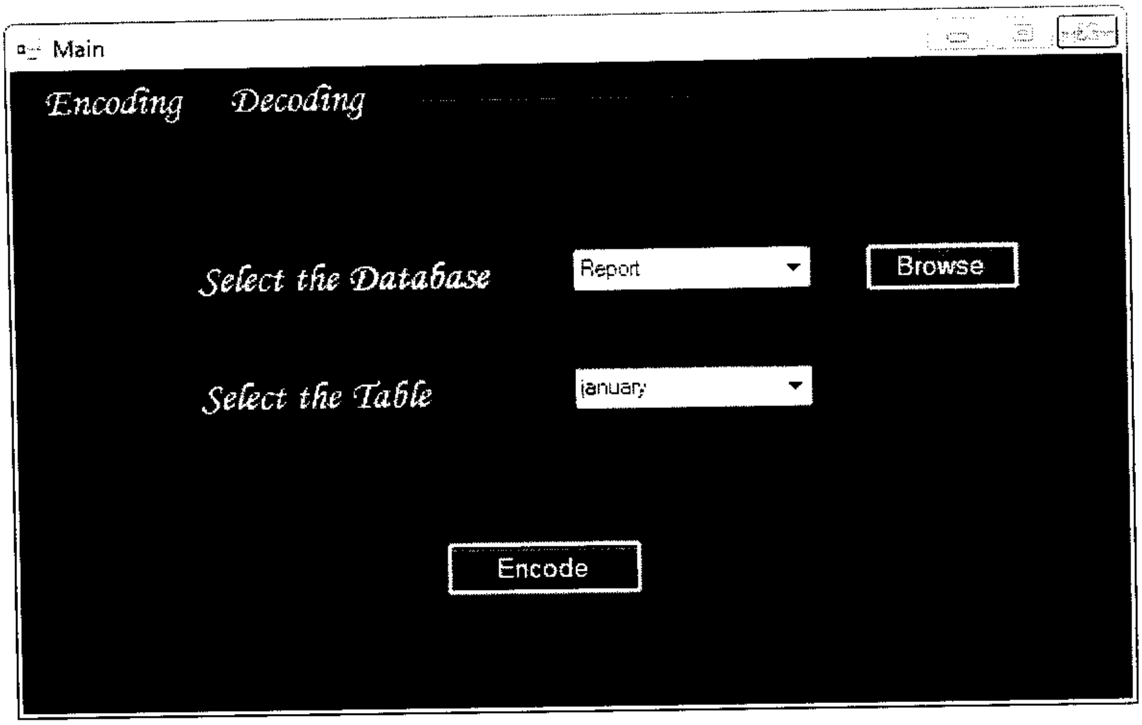
OUTPUT

MAIN FORM



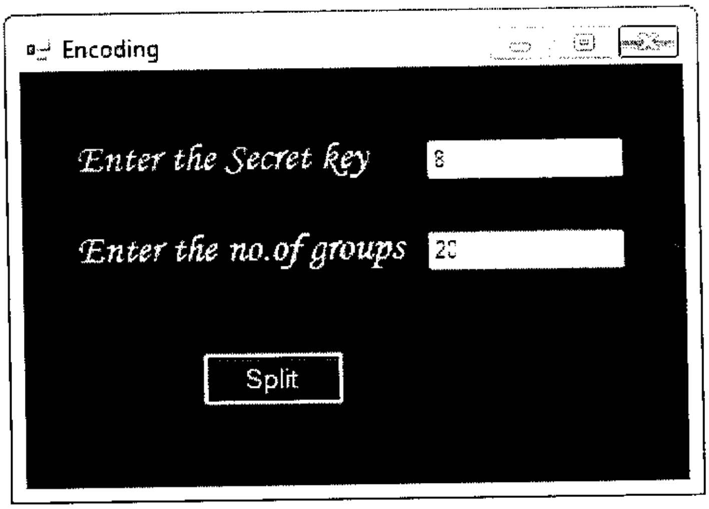


ENCODING



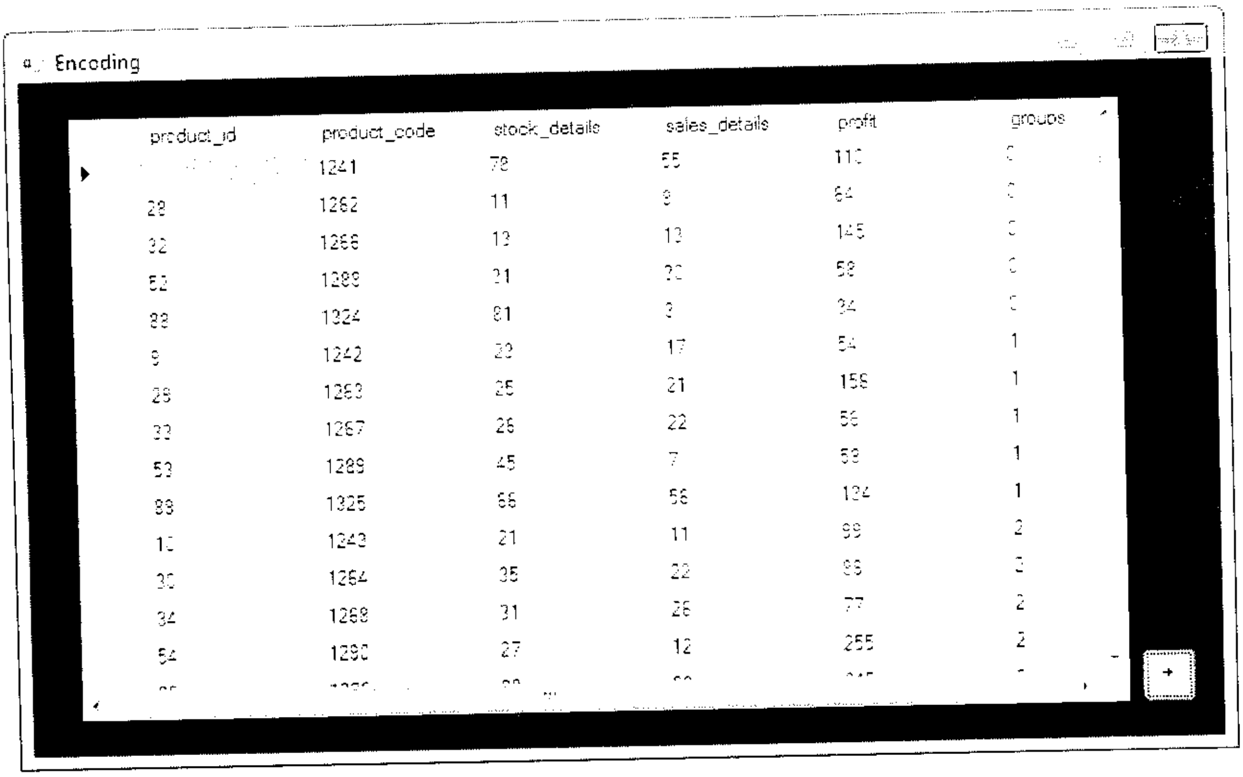
Select the Database: Report

Select the Table: january



Enter the Secret key: 8

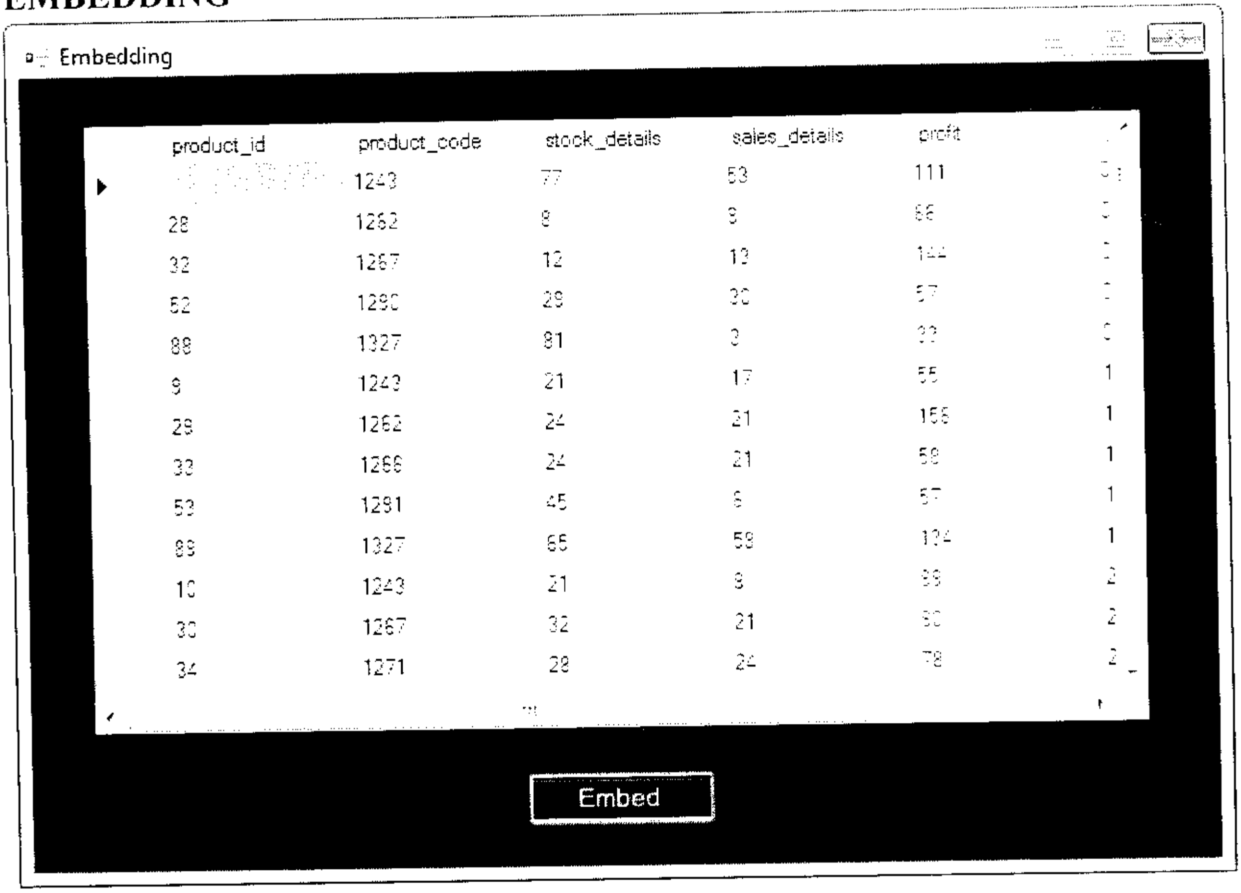
Enter the no.of groups: 20



Encoding

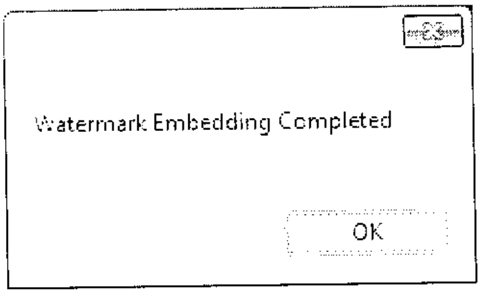
	product_id	product_code	stock_details	sales_details	profit	groups
▶	[illegible]	1241	78	55	110	0
	28	1262	11	8	64	0
	32	1266	13	13	145	0
	52	1288	21	20	58	0
	88	1324	81	3	34	0
	9	1242	22	17	54	1
	29	1263	25	21	158	1
	33	1267	26	22	56	1
	53	1289	45	7	58	1
	89	1325	66	56	134	1
	10	1243	21	11	99	2
	30	1264	35	22	96	2
	34	1268	31	26	77	2
	54	1290	27	12	255	2

EMBEDDING

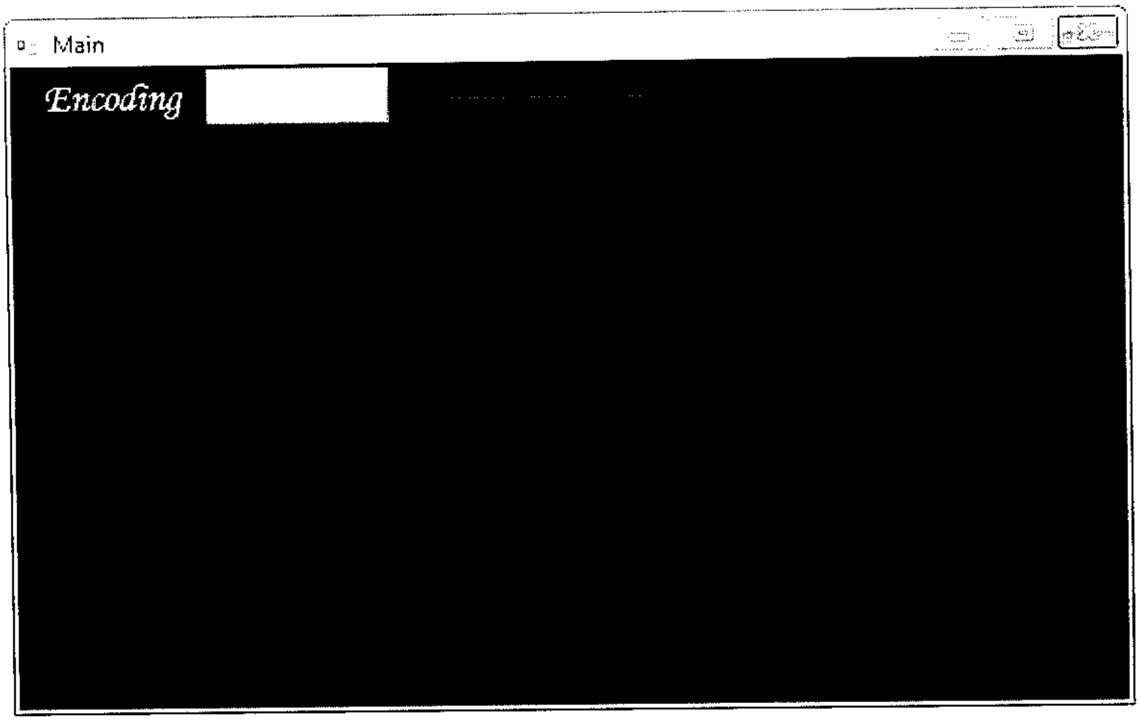


Embedding

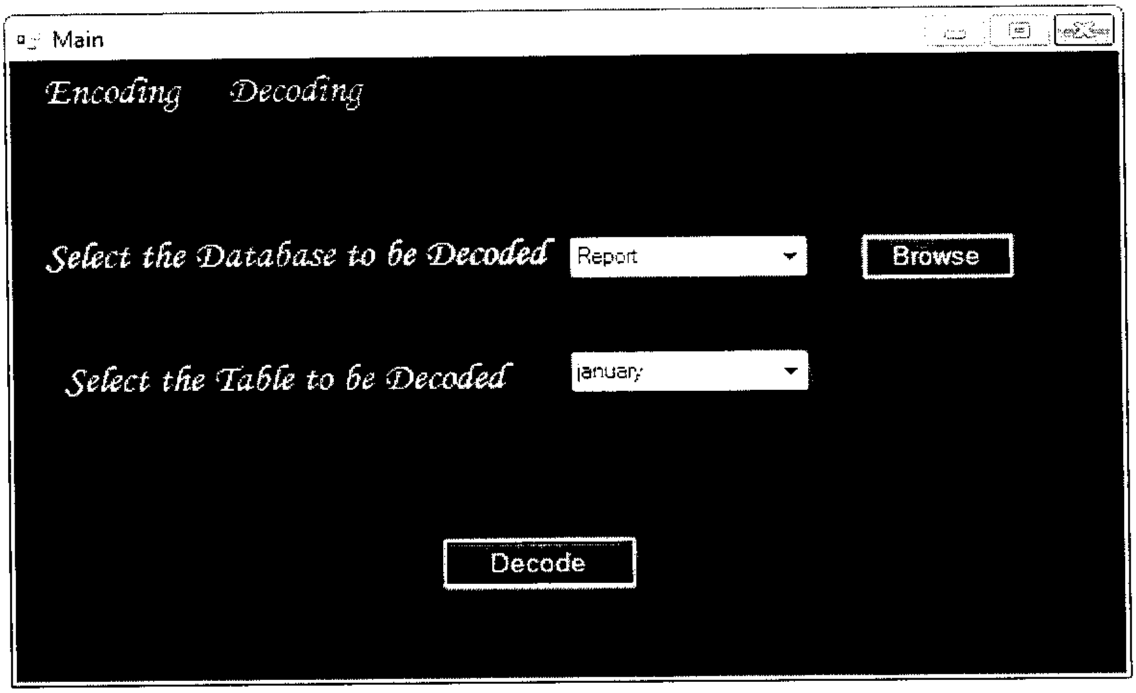
	product_id	product_code	stock_details	sales_details	profit	
▶	[illegible]	1243	77	53	111	0
	28	1262	8	9	66	0
	32	1267	12	13	144	0
	52	1290	29	30	57	0
	88	1327	81	3	33	0
	9	1243	21	17	55	1
	29	1262	24	21	156	1
	33	1268	24	21	58	1
	53	1291	45	6	57	1
	89	1327	65	59	134	1
	10	1243	21	8	99	2
	30	1267	32	21	50	2
	34	1271	28	24	78	2



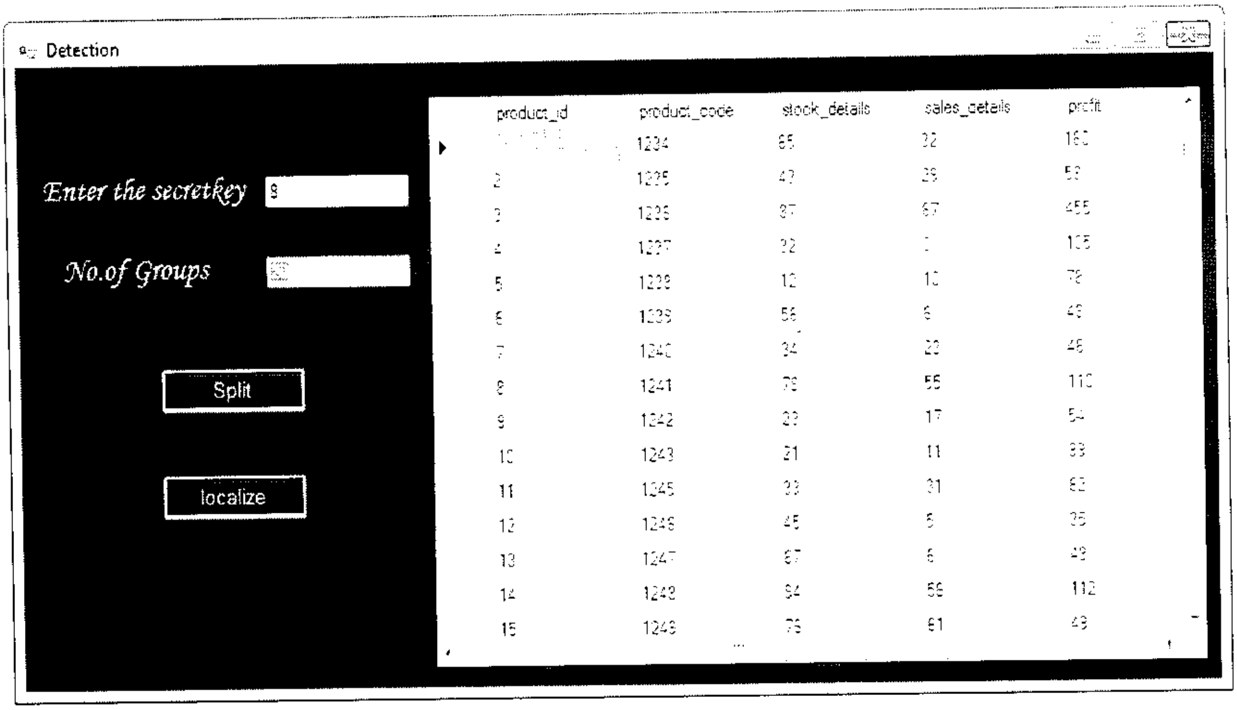
Watermark Embedding Completed


Main
Encoding

DECODING


Main
Encoding
Decoding
Select the Database to be Decoded
Report
Browse
Select the Table to be Decoded
january
Decode

DETECTION

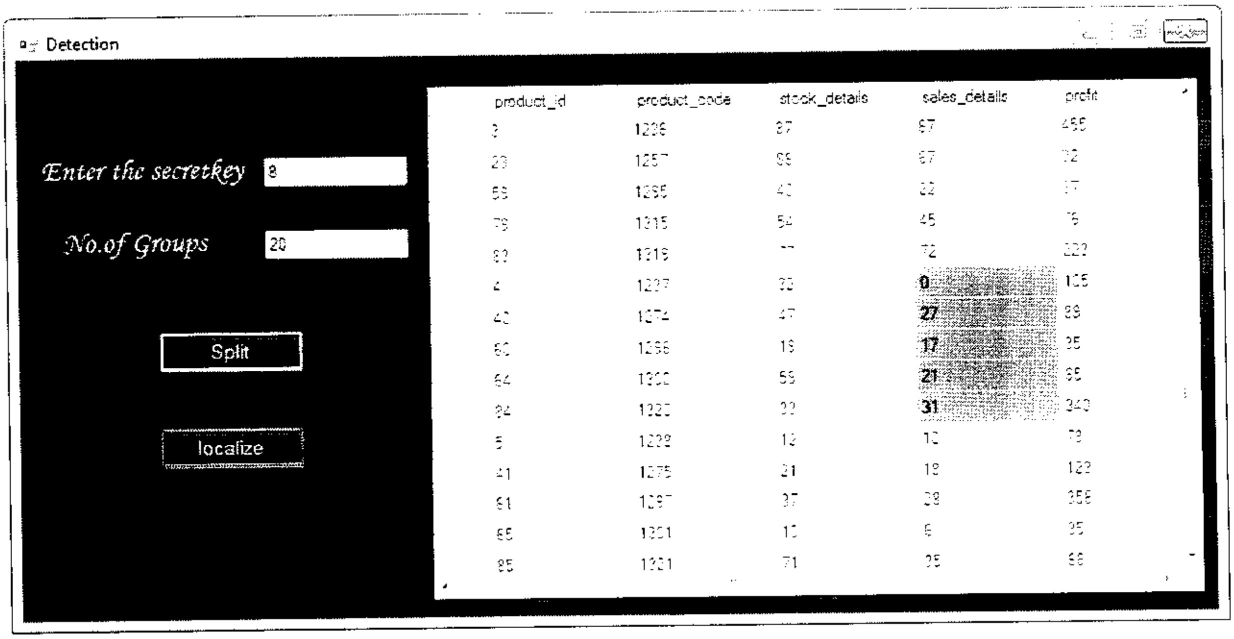


Detection

Enter the secretkey 8

No.of Groups [illegible]

product_id	product_code	stock_details	sales_details	profit
[illegible]	1234	65	32	160
2	1235	47	29	53
3	1236	87	67	455
4	1237	22	[illegible]	105
5	1238	12	10	78
6	1239	56	6	49
7	1240	34	22	46
8	1241	78	55	110
9	1242	23	17	54
10	1243	21	11	33
11	1245	33	31	63
12	1246	45	5	35
13	1247	67	6	49
14	1248	94	59	112
15	1249	78	61	49

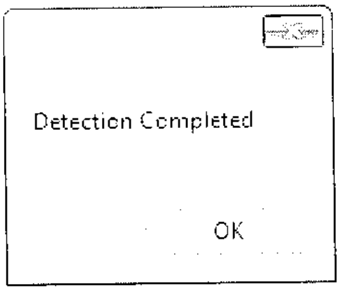


Detection

Enter the secretkey 8

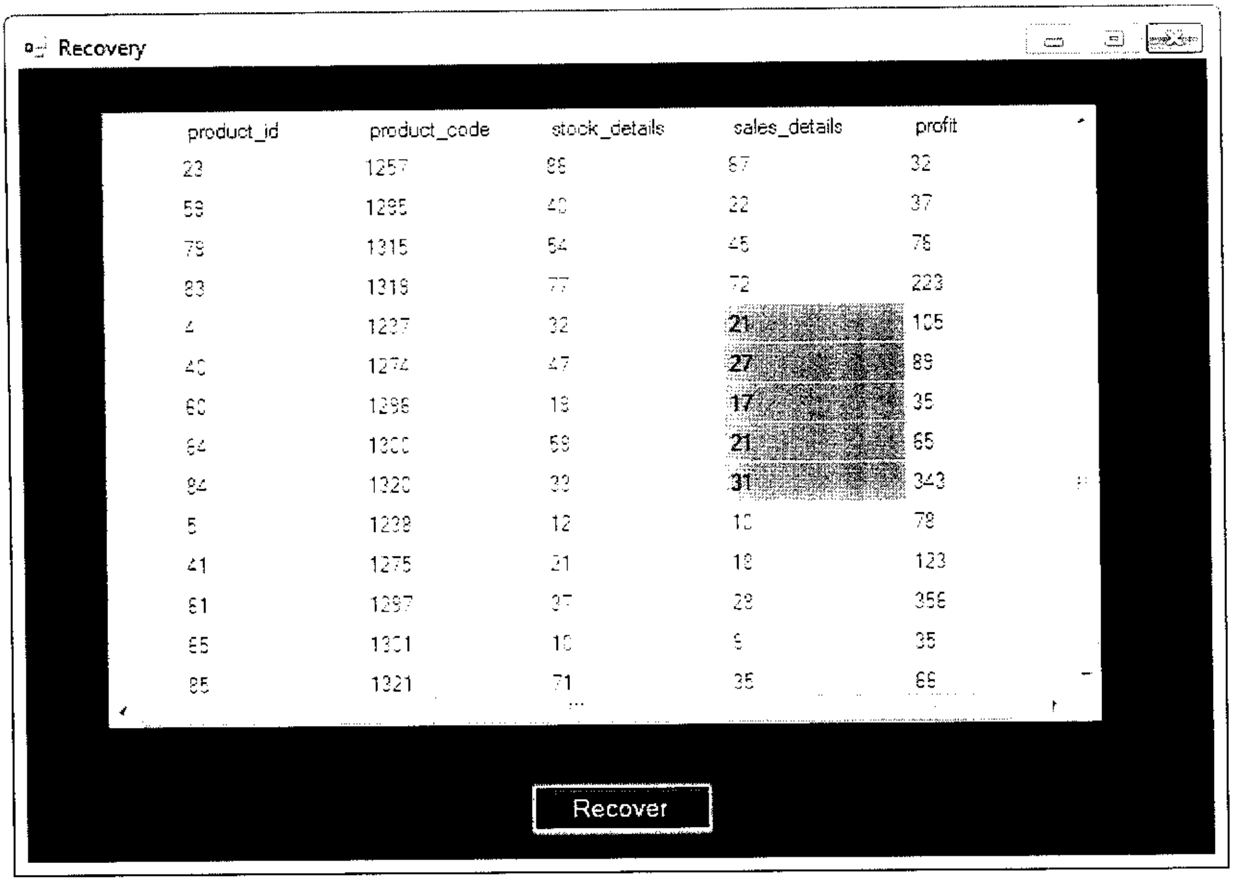
No.of Groups 20

product_id	product_code	stock_details	sales_details	profit
3	1236	87	67	455
23	1257	99	67	72
59	1295	40	22	17
79	1315	54	45	76
83	1319	77	72	222
4	1237	22	**0**	105
40	1274	47	**27**	99
60	1296	19	**17**	35
64	1300	59	**21**	95
84	1320	22	**31**	340
5	1238	12	10	78
41	1275	21	19	122
61	1297	37	38	358
65	1301	10	6	35
85	1321	71	25	66

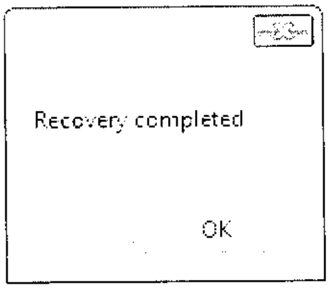


Detection Completed

RECOVERY



product_id	product_code	stock_details	sales_details	profit
23	1257	88	67	32
58	1295	40	22	37
79	1315	54	45	76
83	1319	77	72	223
4	1237	32	**21**	105
40	1274	47	**27**	89
60	1298	19	**17**	35
64	1300	59	**21**	65
84	1320	39	**31**	343
5	1238	12	10	78
41	1275	21	19	123
61	1297	37	28	356
65	1301	10	9	35
85	1321	71	35	69



Recovery completed

REFERENCES

FERENCES

] Guo.H, Li.Y, Liu.A, Jajodia.S., (2006). A fragile watermarking scheme for detecting licious modifications of database relations. Information Sciences 176, pp 1350–1378.

.] Agrawal.R, Kiernan.J., (2002). Watermark relational databases of the 28th Int nference on Very Large Data Bases.

3] Devanbu.P, Gertz.M, Martel.C, Stubblebine.S., (August 2000). Authentic data blication over the internet, in: Proc. 14th IFIP 11.3 Working Conf. in Database ecurity, pp. 102–112.

4] Gross-Amblard.D., (June 2003). Query-preserving watermarking of relational atabases and xml documents, in: Proc. of the 22nd ACM SIGMOD-SIGACT-SIGART ymp. On Principles of Database, pp 191–201.

5] Li.Y, Swarup.V, Jajodia.S., (October 2003). Constructing a virtual primary key for ingerprinting relational data .ACM Workshop on Digital Rights Management, pp 133– 41.

[6] Li.Y, Swarup.V, Jajodia.S., (December 2003). A robust watermarking scheme for elational data, in: Proc. The 13th Workshop on Information Technology and Engineering, pp 195–200.

[7] Pang.H, Tan.K., (March 2004). Authenticating query results in edge computing. The IEEE Int Conf on Data Engineering.

[8] Sion.R, Atallah.M, Prabhakar.S., (2003). Rights protection for relational data. ACM SIGMOD, pp 98–109.