# Query Preserving Relational Database Watermarking

S.A. Shah, Sun Xingming and Hamadou Ali
Network and Information Security Lab Hunan University, Changsha, Hunan, China
E-mail: saeed.arif@gmail.com; sunnudt@126.com: alihamadou@yahoo.fr

Majid Abdul
Pervasive Computing Lab Zhejiang University Hangzhou, Zhejiang P.R. China
E-mail: majedabbasi@yahoo.com

**Keywords:** right protection, relational data, watermarking

**Received:** March 25, 2010

*In order to preserve the query results after watermarking relational data, it is necessary to keep the semantic value of data intact during watermark embedding process. A query preserving relational database-watermarking scheme is proposed in this paper. For watermark embedding, we use alphanumeric data as new embedding channel. The scheme retains the semantic meaning of data after embedding, which makes this a distortion free and query preserving technique. Watermark Embedding is done by adjusting the case of securely selected text data. Our proposed method provides better relational database watermarking solution for the database, which either has no numeric attribute, or has data with zero resilience to data alteration. To make this scheme more secure tuples and attributes selection is done using a secret key known only to the owner. Later the same key is used in detection process. Moreover, there is no need of original data for watermark detection so it is a fully blind scheme. The method proved (through experiments) to be robust against various kinds of attacks. An SQL Server database implementation has shown that our algorithms can be used successfully in real world applications.*

*Povzetek: Predstavljeno je obvarovanje relacij pri povpraševanju v relacijskih bazah.*

## 1 Introduction

Easy modification and reproduction of digital data (software, images, video, audio, and text) without leaving any trace of manipulation makes it very easy victim of piracy. Number of watermarking based solutions proposed so far for copyright protection of relational data. Watermark is a secret code embedded in digital contents. This watermark can be extracted/detected from the watermarked contents and can be used to establish the ownership of data. Watermarking fails to prevent illegal copying but it can be an effective tool for establishing original ownership of pirated data. This discourages piracy and enables owners to prosecute copyright violators.

Growing use of outsourced relational data, especially availability of relational data over the internet, demanded an effective mechanism for copyright protection so that owner of the data can identify pirated copies of their data. Watermarking has proved to be an effective tool for multimedia data so researchers explored this technology for relational data also. Agrawal et al [1] pioneered research on relational database watermarking in 2002. Different schemes [1, 4, 3, 9, and 13] have been proposed after that. Most of the previous work in this area use numeric data as embedding domain for watermark insertion. All these schemes are based on the assumption that there are some data, which can tolerate small changes, without affecting its usability. Some of them use direct LSB domain [1] while other manipulates statistics of the data for watermark embedding [4, 13].

There are few schemes proposed for categorical data watermark but these schemes also introduce significant change to data [2, 15]. A large number of techniques available for multimedia watermarking [3, 5, 7, 10] which proved to be effective but these cannot be applied directly to database. For multimedia, there is a lot of room for embedding any extra information, as there is a large amount of redundant bits. One can play with these bits as long as these manipulations are imperceptible. For multimedia, the most important requirement is to avoid visual distortion whereas for relational data, preservation of semantic value of data is essential. Sometimes even a change of a single bit will change the meaning of data and thus affect query results. For example change of single bit in date like name, address, age, account number etc, will change the value and in turn the query result. Another important challenge that still needs attention is; what if there is no numeric data or, there is data which is not resilient enough for watermark embedding? All these factors lead to the need of scheme, which not only retain the semantic value of data but also preserve query results after watermark embedding. In this paper we are going to propose such a scheme which

works for non-numeric data and also preserve query results by introducing almost zero distortion to semantic value of data while watermark embedding.

The paper is organized as follows. In section 2 related work is discussed. Section 3 discusses our scheme in detail. Section 4 analyzes main features of our scheme. In section 5 different attacks are discussed. Experimental setup and results are also outlined in section 5.A multi-bit watermarking scheme; an extended version of the proposed solution is described in Section 6. Section 7 concludes.

## 2 Related Work

Work on database watermarking started in 2002 when Agrawal and Kiernan presented a robust watermarking scheme for databases [1]. The scheme focused on watermarking relational data with numeric attributes. It is assumed that these numeric attributes can tolerates small amount of modification. Using a secret key and secure hash algorithm, first tuple and then attribute within that tuple are randomly selected for watermark embedding. Finally, selected bits of that attribute are modified in order to embed watermark bits. Robustness of the scheme is shown through experiments and theoretical analysis against different kind of attacks including such as rounding attack, subset attack, and additive attack etc.

Another significant contribution in this area is by Radu Sion et al. [2]. Sion presented different schemes for numeric data and categorical data [4]. For the first time Sion proposed a subset based watermarking method for numeric data. According to this method first all the numeric data is partitioned into subsets using some secret key and then a single bit is embedded within each subset by playing with its statistics. The scheme claimed to be robust against variety of attacks including subset attack, data resorting and transformations attacks. However, it does not seem effective for database which need frequent update, as it requires re watermarking of all the data.

In [14] Yingjiu Li et al proposed a fragile watermarking scheme for relational data authentication. This is a group based technique. Watermark calculated from message digest of the group, which is then embedded in the same group. Since the message digest is fragile even for single bit change, it can be used for authentication of relational data.

In [13] M. Shehab et al presented optimization based watermarking for numerical data. The relational data watermarking is first formulated as constrained optimization problem then solution to this problem is sought either using Genetic algorithms or Pattern search. Here again data is partitioned into subsets and distribution of each partitions is modified to embed a single bit, but embedding is done by solving the optimization problem either for maximization or minimization. This technique is state of the art for numerical data as it introduces minimum distortion to data and is more robust against different attack than earlier schemes.

The above mentioned schemes work only for numeric data. In [2] a robust watermarking proposed by Sion for categorical data copyright protection. In the scheme first tuples are securely selected using secret key then values of categorical attributes of selected tuples are changed to some other values from available pool valid values based on the watermark to be embedded (e.g. change city from New York to Washington). The values are changed by satisfying some constraints so that this change does not affect data usability.

David Gross proposed a scheme for query preserving relational data watermarking in [15]. Scheme claimed to be robust against local queries. For watermark embedding first local queries are identified and then selected data values are modified while preserving these queries. However, the changes made to the data values are significant to most of the applications, which limits its scope.

| Notation | Description |
|---|---|
| r | Row or record of a relation |
| K | Secret key known only to owner |
| n | Number of tuples in the relation |
| v | Number of attributes in the tuple |
| t | No of tuples to be marked |
| $1/m$ | Fraction of tuples to be watermarked |
| s | Size of watermark |
| $K_w$ | Selected watermark key |
| $E_b$ | Existing bit pattern |
| W | Embedded Watermark |
| G | Pseudo random sequence generator |

Table 1: Notations and parameters.

## 3 Our Scheme

Most of the above mentioned schemes [1, 2, 4, 9] are based on the manipulation of numeric data (which must have some margin of error), thus have limited domain. In addition, the schemes discussed above including those for categorical data, introduce distortion in the contents by changing meaning of attribute values which is often not desirable.

In our scheme we introduce a new embedding channel by embedding watermark in non numeric data or more precisely the alphabetic data attributes. Since the database, queries are case insensitive so it will not affect the semantic meaning of data if the case is changed from small to capital or vice versa. We are going to exploit this inherent property present in such kind of data attributes. The proposed method is applicable to all the languages with upper/lower character cases. This work is actually an extension of our previous work for copyright protection of relational data [16]. We now present our technique for watermarking relational database. This technique marks only alphabetic attributes without introducing any change in their semantic meaning. Not all attributes need to be watermarked. Data owner will decide which attributes are more suitable for watermarking. Let R be the database relation with schema R (P, $A_0 \ldots A_{v-1}$) where P is the primary key attribute. Table 1 shows the important parameters used in

our algorithms. For simplicity assume that all the attributes are candidate for watermarking. *m* is used to determine the number of tuple to be watermarked. If *t* denotes the number of tuples to be marked then

$$t \approx n / m$$

r.Ai is used to denote the value of attribute Ai. In this technique, we are using one way hash function H for hash value calculation. There are number of hash function like MD4, MD5 SHA1 SHA256 etc.

A Message Authentication Code (MAC) is computed using a one way hash function that depends on a key[11]. If K denotes the key then H will randomize the input primary key ***"r.P"*** of relation R when H is seeded with **K** known only to owner. The Following MAC is used in our Scheme

$$MAC=H(K, r.P)$$

Where we are using SHA-1 as one way hash function H.

## 3.1 Watermark embedding

Watermark embedding algorithm is given in table 2. Given the relation *R* (P, A1, A2……$A_{v-1}$) with primary key P. Lines 1 through 6 determine tuples and attribute to be marked respectively, using primary key Hash value. Selection of both depends on secret key **K** known to the owner, so only the owner can identify which tuple and which attribute of that tuple is to be marked. An attacker has to guess the tuple as well as attribute within the tuple to destroy the watermark. In line 8 existing bit sequence Eb is extracted by inspecting the text case of selected data. Bit 1 or 0 is extracted following same rules outlined in detection algorithm. Lines 9-11 generate L number of candidate watermark sequences Cj, each of which is then compared to existing bit pattern Eb. Keys used to generate these candidate watermarks may be obtained by seeding G with secret key K and index j of the watermark.

| // Secret keys **K, Kw** and parameters *m, v* are private to the owner. |
|---|
| 1. foreach tuple *r* ∈ do step 2 to 5 |
| 2. Calculate PrimaryKeyHash pHash=H (K, r.P) |
| 3. Select tuple with (pHash mod *m* ==0) |
| 4. Select attribute with index: i= pHash mod *v* |
| 5. selected attribute array: SelectedValue[i]=r.Ai |
| 6. Sort Selected tuples using pHash |
| 7. watermark size *s*= length of SelectedValue[ ] |
| 8. Extract Existing Bit ($E_b$) pattern From selected data values |
| 9. Generate random candidate watermarks Cj each of length *s* using G and keys $K_j$ |
| Cj= G($K_j$) where 1<=j<=L |
| 10. a) Select watermark W= Cj with minimum hamming distance from $E_b$ |
| b) $K_w$ = Key of selected wm sequence |
| 11. Call embedwm( SelectedValue[ ], W)//Embed Watermark in Selected Data |

Table 2: Watermark embedding algorithm.

Finally the bit sequence having minimum hamming to Eb is selected for embedding as watermark W and its key isrecorded as Kw which will be used later for watermark extraction. *embedwm* actually embeds the watermark depending on the corresponding watermark bit. It adjusts the case of selected attribute value according to the conditions laid down in Algorithm 1. Case is adjusted so as to follow most common practice e.g. if watermark bit is 1 then case is converted to title case and for 0 to sentence case.

Sometimes database contain null values in that case mark is not applied. In addition, when there is text such as abbreviation, where a standard is there, no change is applied.

| //Given *R*, K, Kw and parameters m, *v* |
|---|
| 1. foreach tuple r *r* ∈ do steps 2 to 4 |
| 2. PrimaryKeyHash pHash=H(K, r.P) |
| 3. if (pHash mod *m* ==0) then Select This tuple |
| 4. attribute_index i= pHash mod *v* //Select Attribute Ai |
| 5. Sort Selected tuples using pHash |
| //Extract Watermark $W_e$ |
| 6. Repeat 7 to 10 for selected tuples |
| 7. Case-I: When r.Ai is single word |
| 8. If (r.Ai has title case) then |
| $W_e$ [i]=1 |
| else If (r.Ai has all caps ) then |
| $W_e$ [i]=0 |
| 9. Case-II: When r.Ai is multi word |
| 10. If (whole text of r.Ai has Title case ) then |
| $W_e$ [i]=1 |
| else $W_e$ [i] =0 |
| // Verify Watermark |
| 11. Generate W using key $K_w$ |
| 12. result_vector=W XOR $W_e$ |

Table 3: Watermark detection algorithm.

## 3.2 Watermark detection algorithm

Let Alice be the owner of the database and Mallory another person with pirated copy of Alice's Data. We assume that the primary Key is intact because dropping it may cause loss of important data. The algorithm for watermark detection is given in the table 3. In line 3 the tuple is selected where watermark is supposed to be embedded. Line 4 determines attribute marked. Both of these are selected using same secret key K used during embedding. In lines 6 to 10, watermark bits are extracted using the predefined conditions.

When the attribute value consists of a single word then extracted watermark bit is 1, if it has Title case and 0 otherwise. For attribute having multiple words, the watermark bit is 1 if whole text has title case and 0 for sentence case. Watermark verification is done in lines 11 & 12, where, first the original watermark is generated using same secret key and then compared with the extracted watermark.

# 4 Discussion

In this section, we discuss some important features of our scheme related to security, detection, query preservation and case alterations.

## 4.1 Security

Security of our schemes can be defined in terms of difficulty for a malicious attacker to recover, locate or even guess originally embedded watermark. For watermark generation a secure pseudorandom sequence generator G is used. It is computationally infeasible to predict the next number in the sequence [6]. Statistically, the numbers generated by G appear to be a realized sequence of independent and identically distributed random variables, in the sense that the numbers pass standard statistical tests for these properties [8]. A seed value is used to generate the random number.

Same sequence will be generated every time with repeated execution of G with given seed value [6]. We used the secret key K as seed and only the owner knows it. Selection of tuples and attributes is purely key based. Moreover, use of a secure one-way hash function makes this scheme more secure.

## 4.2 Blind detection

There are two types of watermarking methods, one which require original data for the detection of embedded watermark called non-blind and other which don't called blind [10]. Since there is no need of original database to recover/decode embedded watermark, so we can claim that our scheme is Blind in nature.

The watermark $W_e$ is extracted from watermarked relation, which is then verified. It is difficult to keep original version of distributed copy of database because it requires frequent updates, so a blind technique is very helpful.

## 4.3 Query preserving watermark

Watermark embedding is done by adjusting the case of selected data according to predefined rules, which does not change the meaning of data so queries result will not be affected even after embedding.

## 4.4 Reduced number of case alterations

For watermark, a number of random sequences are generated and one of them is selected for embedding. This selected sequence has minimum hamming distance to existing bit sequence, which is extracted from current text case of selected attribute values before embedding. Applying hamming distance for final watermark selection reduces number of text case alteration leading to low text case distortion. This is one of the important contributions.

# 5 Experiments and Attacks Analysis

In this section, we will discuss the survival of embedded watermark against common database attacks. Watermark that survives when its host data is exposed to attacks is called robust watermark. The watermarked data can be attacked in various ways through malicious attacks and benign updates. The most common attacks are:

i. Tuple deletion
ii. Modifying attributes values
iii. Case alteration

The first two attacks may affect the usability of data, so, for an attacker these kinds of attack are often less desirable. The third one is actually a legal attack, which does not affect the usability of data so is most important to study it. We analysed our scheme against these attacks through experiments, and results show that it is robust against the above attacks; means there is very high probability of correctly decoding the embedded watermark even after these attacks.

For experimental purpose, we used SQL server database of more than half million records on Windows Xp platform. The value of "$m$" is kept 10 and number of watermarkable attributes "$v$" is 3 in our sample database. In the following, we present experiment involving different attacks (Data loss, Data Alterations, Change in Case). Experiments were performed repeatedly and their results are averaged over multiple runs.

## 5.1 Tuple deletion attack

In this experiment, randomly selected tuples are deleted and after deletion of every few tuples, the watermark is extracted and compared with originally embedded watermark. The experiment performed many times and average behaviour is plotted in figure 1. It shows that even after deleting 35-40% of tuples, distortion in decoded watermark is up to 12%. In our experiments, we used binary watermark so in this case 88% bits of decoded watermark matched with the actual embedded watermark so distortion (damage/loss in watermark) is only 12%.

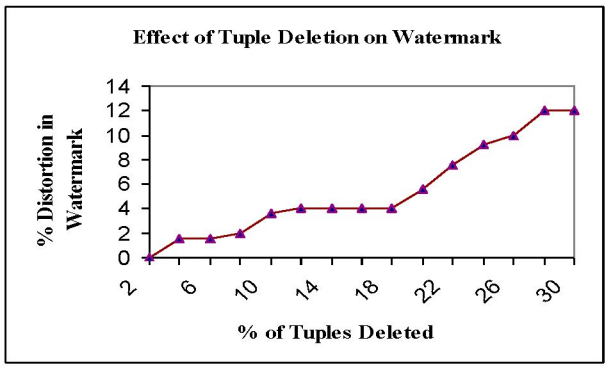


Figure 1: Tuple deletion attack.

## 5.2 Data modification attack

Mallory's (The attacker) priority would be to destroy watermark, while preserving the data. Given no

knowledge of secret key or the original data, the attacker may try to make random modifications to watermarked data values, thus hoping to destroy watermark at some point. In this experiment, we analyse the sensitivity of our scheme to random updates of watermarked data. The demonstrated behaviour is shown in the figure 2. The results show that only 6% distortion in watermark is observed if 35-40% data values are randomly modified. Hence, it is more robust against such kind of attacks as compared to other attacks.

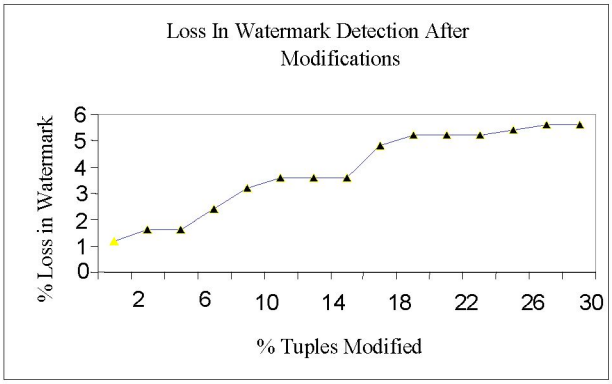


Figure 2: Data modification attack

### 5.3 Tuple sorting attack

During embedding, we sort the selected tuples before embedding watermark, using primary key hash value, and the same process is repeated during detection, so any kind of sorting attack will not harm the detection of watermark.

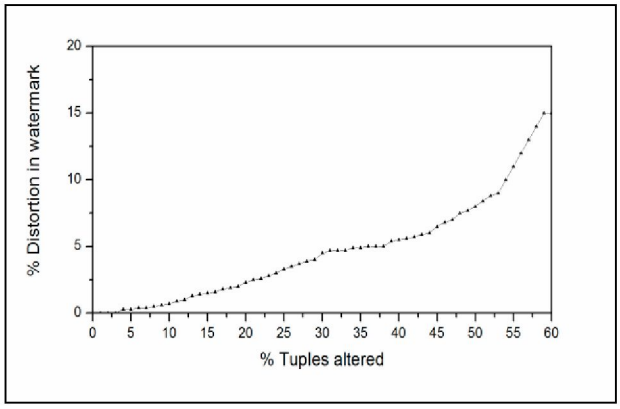


Figure 3: Case alteration attack.

### 5.4 Case alteration attacks

Since we are playing with the case of attributes values so this kind of attack is only specific to our scheme and it is a legal attack so robustness against it must be checked. In this experiment, we randomly (and repeatedly) change the case of attributes values, then extract watermark, and compare it with embedded one. The results are averaged our various runs. As shown in the figure 3. It is observed that our scheme is robust against this kind of attack. After changing the case of up to 60% tuples, the watermark distortion is less than 20%.

It is evident from the results of above experiments, that we can correctly decode the embedded watermark with very high ratio (up to 80%) even after different attacks. Hence we can claim that our scheme is robust against common database attacks.

## 6 Multi-bit watermarking

In this section, an extension to multi-bit watermarking of the proposed scheme is presented. For this purpose, the watermark embedding and detection algorithms are modified. For embedding first all the tuples are securely divided into partitions. A single bit embedded into each partition, which requires that the number of partitions should be much greater than the watermark size so that a single bit can be embedded multiple times.

The simplified versions of embedding and detection algorithms are given in the table 4 and 5 respectively. For sake of simplicity only overview of embedding and detection process is given.

**E1. Secret Grouping:**
All tuples are securely divided into "g" number of groups. Grouping is done as proposed by Shehab in [13]

**E2. Tuple Sorting:**
All the tuples are sorted based on secure hash value of each tuple's primary key

**E3. Secure Tuple and Attribute Selection:**
Within each group, first a tuple then an attribute is randomly selected for marking (same as in algorithm 1)

**E4. Watermark Embedding:**
The case of selected attribute data in a specific group is adjusted to represent the embedded watermark bit

Table 4: Multi-bit embedding algorithm.

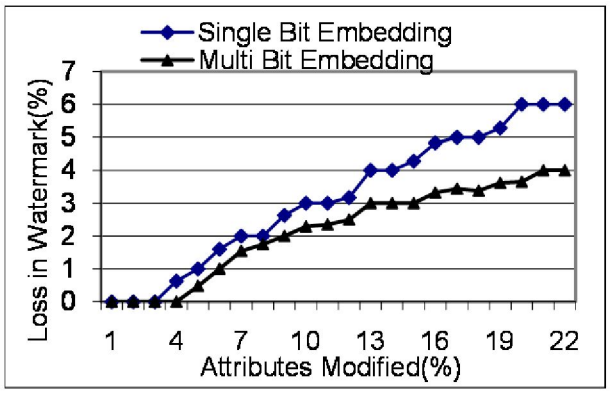


Figure 4: Robustness against value modification attack.

**D1. Grouping**:
All the tuples are securely divided into g groups using the same secret key K and number of group's g

**D2. Extract Watermark $W_e$**:
Sort tuples within each group

**(a)**. Determine marked tuples and attributes from groups (same as in encoding)

**(b)**. Extract watermark bit from the Case of selected data

**(c)**. Apply majority voting method for final watermark extraction

**D3. Watermark verification**:
**(a)** Bit matching of $W_e$ with actual watermark W (Generated using same key K)
**(b)** If match_count/total_count > τ then watermark detected

Table 5: Multi-bit detection algorithm

Experiments show that robustness of the scheme can be improved significantly by using multi-bit embedding. Figure 4 shows the robustness of the multi-bit scheme against data modification attack.
In our experiments we used value of threshold $\tau = 0.8$. We suggest its value within $0.6 \leq \tau \leq 0.8$. If an image is embedded as watermark then τ can be set to even lower value.

# 7 Conclusions and Future Work

In this paper, we introduced a new scheme for watermarking relational database for owner verification and copyright protection. A solution is proposed by:

i. Discovering a new embedding channel for watermarking.
ii. Building an algorithm for watermarking such that it preserves data integrity by introducing zero distortion to its semantic meaning.
iii. Improving robustness by multi-bit embedding

We thus provided an efficient watermarking technique for copyrights protection of relational data. Through experiments, we proved that our scheme is robust against common database attack as well attacks specific to our scheme. In future, we intend to analyze and improve this scheme against other attacks such as subset and partitioning attacks. Another research direction may be to investigate a method for data authentication using fragile watermarks.

# Acknowledgment

This work is supported by the National Basic Research Program of China (973 Program) under grant No. 2006CB303000, the National High Technology Research and Development Program of China (863 Program) under grant 2007AA010404, NSFC Key Project grant No. 60736016, NSFC grants No. 60702065, 60873198, 60973113 and 60973128, Science and Technology Program of Hunan Province grants No.2008FJ4221, Special for National Basic Research Program of China (973 Program) grants No. 2009CB326202 and Higher Education Commission Pakistan through University of AJ&K FDP 2008.

# References

[1] R. Agrawal and J. Kiernan. Watermark relational databases. In *Proc. of the 28th International . Conference. On Very Large Data Bases*, 2002.
[2] R.Sion. Proving ownership over categorical data. In *Proceedings of ICDE* 2004.
[3] I. J. Cox, M. Miller, and J. Bloom. Watermarking applications and properties. In *Proc. International Conference on Information Technology: Coding and Computing*, 2000.
[4] R.Sion, M. Atallah, and S. Prabhakar. Rights protection for relational data. In *Proceedings of ACM SIGMOD 2003*, 2003.
[5] C. Rey and J. Dugelay, A Survey of Watermarking Algorithms for Image Authentication In JASP, No.6, pp. 613-621, 2002.
[6] Schneider B. Applied cryptography", 2nd ed. (1996) Wiley, New York
[7] I. J. Cox, J. Kilian, T. Leighton and T. Shamoon, Secure Spread Spectrum Watermarking for Multimedia, *IEEE Trans. on Image Processing, 6, 12, 1673-1687,* (1997).
[8] Knuth D. Semi numerical algorithms. In: The art of computer programming, vol 2.Addison-Wesley, Reading, MA, 1981
[9] Y. Li, V. Swarup, and S. Jajodia, A Robust Watermarking scheme for relational data. In *Proc. The 13th workshop on information technology and engineering*, pages 195–200, December 2003.
[10] Ingemar Cox, Matthew Miller, Jeffrey Bloom. Digital Watermarking Morgan Kaufmann .2002
[11] N. Ferguson and B. Schneider, Practical Cryptography. Wiley & Sons, 2003.
[12] J. Brassil, L. O'Gorman Watermarking Document Images with Bounding Box Expansion, in Proc. of 1st Int'l Workshop on Information Hiding, Newton Institute, Cambridge UK, May 1996, pp. 227-235.
[13] Mohammad Shehab, Elisa Bertino, Arif Ghafoor Watermarking Relational Data using Optimization Based Techniques *IEEE Transactions on Knowledge and Data Engineering Volume 20 , Issue 1 (January 2008) Pages 116-129*
[14] Y. Li, H. Guo, S. Jajodia, Tamper detection and localization for categorical data using fragile Watermarks in: *4th ACM Workshop on Digital Rights Management, CCS04, October 2004.*
[15] D. Gross-Amblard, Query-preserving watermarking of relational databases and xml documents, in: *Proc. of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database, June 9–12, 2003, pp. 191–201.*
[16] S. A. Shah, S. A. M. Gilani, I. A. Awan: Owner Verification and Copyright Protection of Relational Data. in: *Proc. Of IMEC Jun 2006 252-257 Hong Kong.*