IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 15, 2022, accepted March 3, 2022, date of publication March 8, 2022, date of current version March 17, 2022.

*Digital Object Identifier 10.1109/ACCESS.2022.3157866*

# Comparative Analysis of Relational Database Watermarking Techniques: An Empirical Study

**SAPANA RANI AND RAJU HALDER, (Member, IEEE)**
Indian Institute of Technology Patna, Patna 801106, India

Corresponding author: Sapana Rani (sapana.pcs13@iitp.ac.in)

**ABSTRACT** Digital watermarking is considered one of the most promising techniques to verify the authenticity and integrity of digital data. It is used for a wide range of applications, e.g., copyright protection, tamper detection, traitor tracing, maintaining the integrity of data, etc. In the past two decades, a wide range of algorithms for relational database watermarking has been proposed. Even though a number of surveys exist in the literature, they are unable to provide insightful guidance to choose the right watermarking technique for a given application. In this paper, we provide an exhaustive empirical study and thorough comparative analysis of various relational database watermarking techniques in the literature. Our work is different from the existing survey papers as we consider both distortion-based and distortion-free techniques along with a rigorous experimental analysis demonstrating a detailed comparison on robustness, data usability, and computational cost with considerable empirical evidence.

**INDEX TERMS** Digital watermarking, relational database, empirical study, robustness, data usability.

## I. INTRODUCTION

Digital watermarking is considered one of the most promising techniques to verify the authenticity and integrity of digital data. It is used for a wide range of applications, e.g., copyright protection, tamper detection, traitor tracing, maintaining the integrity of data, etc. For several decades, relational databases are at the heart of many information systems. As they contain crucial information, they must be protected before sharing them to the world of the internet. Although encryption is used to protect the data stored in a relational database from being accessed by individuals with malicious intent, but it is very restrictive in nature. Since the first proposal in 2000 in [1] that used digital watermark for protecting a database of map information, various relational database watermarking techniques have been proposed in the literature thereafter. Among them, the first and most significant one is proposed by Agrawal and Kiernan in [2]. The database watermarking techniques embed a piece of information (known as watermark) in an underlying data and extract it later from any suspicious content in order to verify the absence or presence of any possible attacks. The former phase is known as Embedding phase, whereas the later phase is known as Detection or Verification phase. In general, these database watermarking techniques are classified as (i) distortion-based techniques that embed the watermark into the underlying content of the data and (ii) distortion-free techniques that generate the watermark based on various characteristics of the data.

A number of survey papers [3]–[11] already exist in the literature, which provides a comprehensive summary of different techniques and their comparison. Authors in [3] elaborated the features of the relational databases, application of digital watermarking, attack analysis of the then existing distortion-based and distortion-free watermarking techniques. A survey of reversible watermarking approaches has been proposed in [4], [5]. A holistic study of distortion-based watermarking techniques has been proposed in [6]. A recent survey on multimedia and database watermarking is reported in [7] where, in addition to different multimedia artifacts, a comparative summary of only nine existing database watermarking techniques is presented. Other significant works related to the survey of relational database watermarking include [8]–[11].

Despite this, the existing survey papers do not carry the following insights that may provide an appropriate guidance to choose the right watermarking technique for a given application: (i) what should be the criteria to compare different categories of watermarking techniques, (ii) how to show empirically that a particular watermarking technique is better than the other techniques, (iii) lack of emphasis towards distortion-free techniques.

The associate editor coordinating the review of this manuscript and approving it for publication was Gianmaria Silvello.

To fill this knowledge gap and to provide a well-informed guidance to the users for a wise decision on choosing right watermarking technique, in this paper, we provide an exhaustive empirical study and thorough comparative analysis of various relational database watermarking techniques. Our work is different from the existing survey papers as we consider both distortion-based and distortion-free techniques along with a comprehensive experimental analysis of robustness, data usability, and computational cost, and their comparisons with considerable empirical evidence.

In order to achieve these objectives, our major contributions in this paper are as follows:

1) We classify the distortion-based and distortion-free techniques in various categories on the basis of the algorithmic steps adopted as well as the type of the watermark information used in the algorithm.
2) We perform an empirical study on a selected number of algorithms, each representing the class of algorithm it belongs to. In particular, we perform a rigorous experimental analysis demonstrating a detailed comparison on robustness, data usability, and computational cost.
3) Our empirical analysis provides a well-informed guidance to the users for a wise decision on choosing right watermarking technique.

The structure of the rest of the paper is as follows: Section II explains the research methodology we adopted. Section III and IV provide the detailed comparative performance analysis of distortion-based and distortion-free algorithms respectively. Section V discusses our evaluation-results w.r.t. the existing experimental observations. Section VI provides a guidance to the users for choosing the right watermarking technique for a given application. Finally, we conclude our work in Section VII.

## II. RESEARCH METHODOLOGY

### A. PRIMARY STUDY SELECTION

We perform the primary study by searching the major online scientific repositories (depicted in Table 1) using the following search queries: ''relational database watermarking'', ''watermarking of relational databases'', and ''copyright protection of relational databases''. In all cases, we set as a filter the years from 2002 to 2022.

**TABLE 1.** Online scientific repositories.

| Digital Library | URL |
|---|---|
| Google Scholar | https://scholar.google.com/ |
| IEEE Explore | https://ieeexplore.ieee.org/Xplore/home.jsp |
| ACM digital library | https://dl.acm.org/ |
| Science Direct | https://www.sciencedirect.com/ |
| MDPI | https://www.mdpi.com/ |
| Springer | https://www.springer.com/gp |

We carefully analyze each and every publication obtained in the search result by following the inclusion and exclusion criteria mentioned in the subsequent subsection.

### B. INCLUSION/EXCLUSION CRITERIA

In this study, we consider research works published in journal, conference, symposium, or workshop and we exclude other kinds of works such as books, newsletters, magazines, technical reports, Ph.D. thesis, and undergraduate/master project documents. These criteria are depicted in Table 2.

**TABLE 2.** Inclusion and exclusion criteria.

| Criteria | Explanation |
|---|---|
| Exclusion | The research works related to other databases watermarking like XML, JSON, etc.<br>It is a patent.<br>It is not published in the English language.<br>It is a Ph.D. thesis or undergraduate/master project document.<br>It is a book, newsletter, magazine, or technical report. |
| Inclusion | It is a journal, conference, symposium, or workshop paper and the title, keywords, and abstract explicitly indicate that the paper is related to relational database watermarking. |

### C. SELECTION RESULTS

Considering the above-mentioned search queries, we obtain the following results:

- Google Scholar: 497 results
- IEEE Xplore Digital Library: 24 results
- ACM Digital Library: 6 results
- Science Direct: 16 results
- MDPI: 2 results
- Springer: 68 results

**TABLE 3.** Summary of articles by the type of the publication.

| Sites | #Articles | % Article |
|---|---|---|
| Journal | 46 | 48.93 |
| Conference | 34 | 36.17 |
| Symposium | 7 | 7.44 |
| Workshop | 7 | 7.44 |

As these search results overlap, we remove the duplicate entries and obtain 416 publications. Finally, after applying the inclusion and exclusion criteria, we obtain 94 publications that we consider in our paper. The summary of the articles by type of publication and the temporal trend of these research publications under consideration are depicted in Table 3 and Figure 1 respectively. We analyze these 94 papers on the basis of a brief overview of the watermarking technique, the data set used in the experiment, and the attacks performed.

### D. QUALITY ASSESSMENT

We subsequently check the quality of the research works. The studies were classified on the basis of the types of algorithms adopted as follows: The distortion-based techniques are classified in the following six categories:

- Meaningless bit-pattern as the watermark.
- Virtual primary key based.
- Image as the watermark.

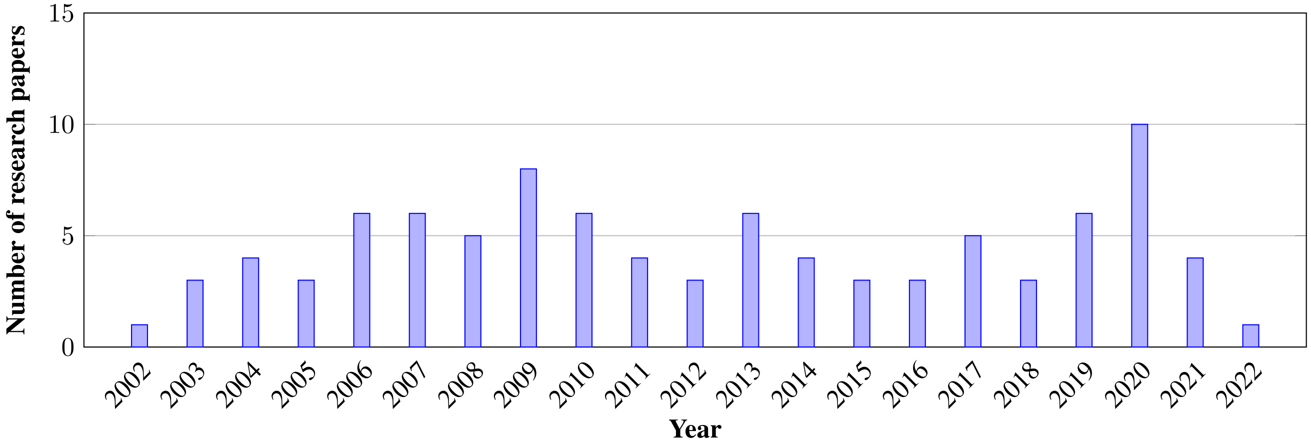


**FIGURE 1.** Temporal distribution of the 94 research works considered in this paper.

- Partitioning based.
- Fake tuple or fake attribute insertion.
- Fingerprinting techniques.
- Other Meaningful watermark information.

Whereas, the distortion-free techniques are classified as:

- Permutation of tuples.
- Conversion of the database into binary form.
- Attribute reordering.
- Content characteristics based.
- Others.

### E. RESULTS

We examine the motivations, contributions, future works of the papers which passed the quality assessment. We select one algorithm in each category that follows any one of the following criteria for the experimental analysis:

1) Criteria 1: Select the pioneer work if the other recent works are minor variants of the pioneer work and there is no significant improvement.
2) Criteria 2: If there is a significant improvement in the recent work compared to the previous works then select the recent one.
3) Criteria 3: Select a work in a category if the work is published in a publication having a higher core ranking and h-5 index.

Let us discuss the research works under each category of distortion-based and distortion-free techniques in detail.

#### 1) DISTORTION-BASED TECHNIQUES

The distortion-based watermarking techniques are classified on the basis of the algorithmic steps adopted as well as the type of the watermark information, described below:

##### a: MEANINGLESS BIT-PATTERN AS THE WATERMARK

Authors in [2], [12]–[20] propose the watermarking algorithms that embed a meaningless bit pattern of the watermark into the data set. Authors in [2] have proposed the algorithm in which hash function is used to decide the marking of a particular tuple. Authors in [12], [14]–[16], [21] extend the proposal of [2]. For example, in [12] the pseudo-random number generator is used instead of a hash function. In [14] chaotic random number generator is used instead of the hash value. Gupta *et al.* in [15] extend the proposal of Agrawal *et al.* in [2] and propose a reversible watermarking algorithm. Authors in [16] use the similar approach of [2] but instead of flipping the least significant bits (LSB) they embed random digits (0 to 9) at LSB of the attribute values. Authors in [17] apply data flow analysis to identify the variant and non-variant parts of the relational database, and then apply the watermarking algorithm in [2] to embed the watermark.

##### b: VIRTUAL PRIMARY KEY BASED

In most of the watermarking algorithms, it is assumed that the primary key exists and is not distorted by the attackers. However, it may not be always true. To deal with this situation, various techniques have been proposed to generate and use Virtual Primary Key (VPK) instead of a primary key. Authors in [2] propose an extended proposal named as S-Scheme in [22]. In S-scheme, one attribute is used to generate the VPK and the remaining attributes are used for watermark embedding. Authors in [22] propose E-scheme and M-scheme. The VPKs generated in E-scheme is similar to the S-scheme, but it considers all of the attributes. M-scheme considers more than one attribute per tuple to generate the VPK. In this approach, each time a different attribute is selected and hence is more resilient towards the delete problem. Other approaches based on virtual primary keys are proposed in [23]–[27]. Approaches in [23], [24] are similar to the M-Scheme. Two attributes having hash values near zero are considered. In [23], the textual attributes are considered. The VPK is generated based on two numeric attributes in [24]. Different attributes are selected each time in [25]. The HQR-Scheme [26] generates one VPK per tuple based on the cyclic model of the attribute.

### c: IMAGE AS THE WATERMARK

Various watermarking approaches [28]–[36] embed images as the watermark into the relational databases. All these approaches first group the tuples and then embed the bit string of the image watermark. Authors in [28] insert a binary image watermark into a relation. In the case of text data, the carriage return character represents 1 and the linefeed character represents 0 of the watermark bits. In the case of numeric data, the watermark bits are embedded in the LSBs of the attribute value.

In most of the techniques, the partitioning of tuples is based on hashing. However, in the case of Huang *et al.* in [29], the tuples are clustered into equivalent classes by using the k-means algorithm. The parity of the watermark bit is compared with the LSB of the candidate attribute for embedding the watermark bit. The location of the embedded watermark is assured by the clustering method. In [30], the original image of size $N \times N$ is converted into a binary string of length $L = N \times N$. The tuples are grouped into $L$ groups based on the hash function, and an $i^{th}$ bit of the binary string is embedded into the bit positions of a fixed attribute in the $i^{th}$ group. The authors in [31] follow the same algorithm as in [30] but they do not consider a fixed attribute and they do not consider the order of image during computing the bit position. After marking, the usability constraints are also checked. The approach in [32] is also similar to [31]. The difference is that they have divided the image into two parts: *header* and *image data*. The header is used for the grouping of the tuples and the image data is converted into a binary string and embedded into these groups.

### d: PARTITIONING BASED

The partitioning based watermarking techniques [37]–[44] partition the data into various groups and embed the watermark into these groups independently. In [37], a marker tuple is used for partitioning and one watermark bit is embedded into one group maintaining the usability constraints. In [38], instead of marker tuple, the hash function is used for partitioning the tuples into groups, and in each group, the watermark bit is embedded by altering the group statistics satisfying the usability constraints. In [40] also, the hash function is used for partitioning. The changes are minimized by selecting a few tuples for watermarking and the watermark (generated from date-time) bit is embedded in each of the selected tuples. In [43], the tuples are partitioned and in each partition, two types of watermark, attribute watermark $W_1$ and tuple watermark $W_2$ are embedded.

### e: FAKE TUPLE/ATTRIBUTE INSERTION

The watermarking techniques in this category insert a new tuple or a new attribute into the database relation as a watermark. In [45], probability distributions are used to determine the properties of the new tuple inserted as a watermark. In [46], a new attribute is inserted into the existing relation. Parity checks are calculated from each attribute and appended to that attribute. The new attribute also has a value from the aggregate function of any of the attributes for all tuples.

### f: FINGERPRINTING TECHNIQUES

A fingerprint is a piece of meaningful information, e.g. social security number that is used as a watermark. Authors in [48] extend the proposal of [2] but embed a fingerprint of length $L$ computed from a hash function taking input as secret key $K$ and user identifier $n$. Liu *et al.* in [47] propose a block-oriented fingerprinting technique. The hash function is based on a secret key and the buyer's ID is used to generate the fingerprint. Authors in [49] propose a twice-embedding watermarking scheme. In the first process, the fingerprint value is used to select the position and the embedding value for every group. In the second process, a pattern is embedded using the fingerprint as the secret key. Authors in [22] also extend the proposal of [2] but they embed fingerprint instead of meaningless bit pattern and they propose schemes named as E-scheme and M-scheme for constructing the virtual primary keys. In [50], watermarking is based on integer linear programming constraint solving. In [51], a buyer's "thumb impression" is used for embedding the fingerprints.

### g: OTHER MEANINGFUL WATERMARK INFORMATION BASED

In [41], the database tuples are partitioned based on the hash function, and meaningful information is embedded in a single attribute as the watermark. Authors in [54] use a pseudorandom sequence number to know the attribute and bit position where the watermark is to be embedded. Similarly [52], [53], [55], [56] also embed meaningful information as the watermark.

A brief overview of different distortion-based watermarking techniques within each category is depicted in Tables 4, 5, and 6.

## 2) DISTORTION-FREE TECHNIQUES

The distortion-free techniques can be classified into following categories:

### a: PERMUTATION OF TUPLES

In these techniques, the order of the tuples is arranged based on secret parameters without causing any distortion in the data values. The significant proposals that perform tuple-reordering based watermarking are proposed in [57]–[61]. In [57], some secure parameters are used to partition the tuples into groups. The order of two tuples is changed based on the hash values of the tuple and the watermark bit. In [62], the value of some critical attribute(s) is used to re-order the tuples relative to a secret initial order, e.g., ascending. The proposed schemes in [58]–[61] are also similar to the approach as proposed in [57] as they also partition the tuples into groups and the tuples are re-ordered in a group that corresponds to the watermark.

TABLE 4. Distortion-based Watermarking Techniques ('-' indicates '*not mentioned*').

| Category | Paper | Brief Overview | Data Set used | Attacks analysed | Selected for experiment | Selection Criteria |
|---|---|---|---|---|---|---|
| Meaningless bit-pattern as the watermark | Agrawal et al. [2] | Hash function is used for selecting the attribute value and the bit position to be marked. | Forest CoverType | Bit flipping, mix and match, additive, invertibility, subset | Yes | Criteria 1 |
| | Agrawal et al. [12] | Uses pseudo-random sequence generator instead of hash function. | same as [2] | same as [2] | No | |
| | Agrawal et al. [13] | Uses pseudo-random sequence generator instead of hash function. | same as [2] | same as [2] | No | |
| | Lafaye [20] | Describes the security analysis of [2] | Random databases and keys were generated | - | No | |
| | Gupta et al. [15] | Watermark bits replace the bit of the integer part of the attribute values and it is inserted in the fraction part of the attribute value. | Generated 270 sets of random documents | mix and match | No | |
| | Qin et al. [14] | Uses chaotic random series based on the Logistic chaos equation to avoid collision of hash function. | A table of retail data set | subset extracting, subset addition, randomized data substituting, additive | No | |
| | Xiao et al. [16] | Digits (0,1,...,9) are embedded at the LSB bits of the candidate attribute values. | railway freight information | massive deletion, random deletion, incremental update, epsilon, collusion attack | No | |
| | Rani et al. [17] | Applies data flow analysis to identify variant and invariant part and embeds the watermark into invariant part applying [2]. | same as [2] | Update attack | No | |
| | Zhang et al. [18] | Based on LSB modification for numerical data and space embedding, symbol modification, text modification for textual data. | 1. German credit risk dataset, 2. Dow Jones Industrial Average (DJIA) stock dataset, 3. Reddit World News Channel historical news headlines dataset | subset deletion, subset modification, subset addition | No | |
| | Melkundi et al. [19] | Watermarked both textual and numerical data and for numerical data, LSB bit flipping is performed. | real time database relation related to ticket assignment. | subset insertion, subset deletion, subset alteration | No | |
| Virtual Primary Key Based | Agrawal et al. [2] | The VPK is generated by using only one attribute, and the rest of the attributes are selected to perform the WM embedding. | No experiments performed | - | No | Criteria 1 |
| | Li et al. [22] | MSB of more than one attribute of each tuple is used to generate virtual primary key. | same as [2] | bit flipping, subset, superset, attribute(addition, deletion, modification), collusion, invertibility, additive | Yes | |
| | Chang et al. [23] | Textual attributes are used to generate virtual primary keys. | Artificially generated database | Tuple deletion, tuple alteration, tuple insertion | No | |
| | Khanduja et al. [24] | Similar to M-scheme of [22] but only two specific attributes are used for VPK generation. | No experiments performed | subset deletion, subset alteration, attribute attack, tuple sorting, additive, invertibility, linear transformation | No | |
| | Gort et al. [25] | Focused on selecting each time different attributes and different bit for VPK generation. | same as [2] | Attribute deletion | No | |
| | Gort et al. [26] | Based on the cyclic model of the attribute | same as [2] | same as [25] | No | |
| | Gort et al. [27] | Proposes double fragmentation of the watermark by using the existing redundancy in the set of virtual primary keys | same as [2] | attribute deletion, tuple addition, tuple deletion | No | |

**TABLE 5.** (Continuation of Table 4) Distortion-based Watermarking Techniques ('-' indicates *'not mentioned'*).

| Category | Paper | Brief Overview | Data Set used | Attacks analysed | Selected for experiment | Selection Criteria |
|---|---|---|---|---|---|---|
| Image as Watermark | Hu et al. [31] | Similar to [30] but not considered a fixed attribute to embed the watermark bits | Generated synthetic data | subset addition, subset selection, subset alteration | No | Criteria 2 |
| | Zhang et al. [28] | Insert a binary image watermark W into a relation in form of 0,1 sequences | Real life KDD Cup 98 data set | subset addition, subset alteration, subset deletion | Yes | |
| | Sardoudi et al. [34] | image watermark embedded into numerical attributes | same as [2] | modification attack | No | |
| | Wang et al. [30] | Bits of a watermark image is embedded into bit positions of a fixed attribute of selected tuples in each group | Randomly generated tuples | subset selective, subset additive, subset modification, subset reverse order | No | |
| | Huang et al. [29] | Uses k-means clustering algorithm to locate the embedded watermark | Meteorological database | subset update, subset delete | No | |
| | Zhou et al. [32] | Similar to [31] but image is divided into two parts, one part is used in grouping of tuples and another is used in embedding | - | selection, addition, alteration | No | |
| | Al-Haj et al. [35] | Based on inserting binary image watermarks in non-numeric multi-word attributes of selected database tuples. | Constructed own database | subset deletion, subset addition, subset alteration, subset selection | No | |
| | Yige et al. [36] | Transformed frequency coefficients to embed image watermark | same as [2] | tuple deletion, tuple addition, tuple modification | No | |
| | Gort et al. [33] | Spatial image watermarking technique is used | Same as [2] | tuple addition, tuple deletion | No | |
| Fingerprinting techniques | Liu et al. [47] | Hash function is used to generate the fingerprint. | - | bit flipping, subset, attribute (insert, delete, modify), collusion, additive | No | Criteria 3 |
| | Li et al. [48] | Extends the proposal in [2] but a fingerprint is embedded | Same as [2] | bit flipping, invertibility, subset and superset, collusion | Yes | |
| | Guo et al. [49] | Proposes twice-embedding scheme. The fingerprint bits are embedded in the first round. In second round a pattern is embedded using the fingerprint as secret key. | Same as [2] | subset selection, subset addition, subset alteration | Yes | |
| | Lafaye et al. [50] | Watermarking is based on integer linear programming constraint solving. | 1. Synthetic data 2. Forest CoverType data set | subset attacks, data alteration | No | |
| | Solami et al. [51] | A buyer's "thumb impression" is used as the fingerprint to be embedded. | Rail system ticket pricing related data set | collusion, tuple deletion, tuple alteration, insertion | No | |
| | Li et al. [22] | Extends the proposal in [2] but embed meaningful fingerprint. | same as [2] | bit flipping, subset, superset, attribute (add, delete, modify), collusion, invertibility, additive | No | |
| Fake tuple or attribute insertion | Pournaghshband [45] | Inserts new fake tuples as watermarks and properties of these watermarks are determined by probability distributions. | Flight scheduling database | subset of attribute, update, alteration | No | Criteria 2 |
| | Prasannakumari [46] | A virtual attribute is added as the watermark. The aggregate function and parity checks is used to calculate the values of that attribute. | Sample data set | Insertion, alter, deletion | Yes | |

**TABLE 6.** (Continuation of Table 4) Distortion-based Watermarking Techniques ('-' indicates '*not mentioned*').

| Category | Paper | Brief Overview | Data Set used | Attacks analysed | Selected for experiment | Selection Criteria |
|---|---|---|---|---|---|---|
| Partitioning Based | Kamran et al. [39] | Data is partitioned by using hash function satisfying the usability constraints and each bit of the watermark is embedded in each partition. | biomedical and biomedicine data sets | deletion | No | Criteria 3 |
|  | Kamran et al. [40] | Data is partitioned by using hash function and watermarks bits are embedded in each selected tuple. | Real life data set that shows power consumption rates | deletion, insertion, alteration, multifaceted, collusion, additive | Yes |  |
|  | Shehab et al. [38] | Data is partitioned by using hash function and watermark bit is embedded by altering the partition statistics. | same as [40] | deletion, alteration, insertion | No |  |
|  | Huang et al. [41] | Data partitioned using hash function and single bit of owner's watermark is embedded in one group. | - | subset selection, subset alteration, subset addition, attribute cutting | No |  |
|  | Sion et al. [37] | Data partitioned based on marker tuple and one bit of the watermark is embedded into one partition. | Walmart Sales database | Data loss attack, data alteration attack | No |  |
|  | Rao et al. [42] | Data partitioned based on marker tuple and one bit of the watermark is embedded into one partition. | same as [2] | Attribute cutting, subset addition, alteration | No |  |
|  | Guo et al. [43] | Data partitioned by using hash function and two kind of watermarks: attribute watermark and tuple watermark. | same as [2] | Attribute value modify, tuple insertion (single and multiple), tuple deletion (single and multiple), tuple modification(single and multiple) | No |  |
|  | Khanduja et al. [44] | improved hash partitioning using bacterial foraging algorithm. | National geochemical survey database of US. | attribute reordering, subset deletion, subset alteration, subset insertion, linear transformation. | No |  |
| Other meaningful watermark information | Huang et al. [41] | Partitions tuple based on hash function and embeds watermark bits in single attribute. | - | subset selection, subset alteration, subset addition, attribute cutting | Yes | Criteria 3 |
|  | Guo et al. [52] | meaningful watermark information embedded. | Wisconsin Diagnostic Breast Cancer data set | subset selection, subset addition, subset alteration | No |  |
|  | Cui et al. [53] | meaningful watermark information embedded. | Synthetic data | Same as [52] | No |  |
|  | Hu et al. [54] | Random pseudo sequence number is used to identify the candidate attribute and the bit to be embedded. | Synthetic data | tuple deletion, tuple insertion, tuple alteration | No |  |
|  | Cui et al. [55] | meaningful watermark information embedded. | Synthetic data | attribute subsetting, attribute flipping, mix and match | No |  |
|  | David Gross-Amblard [56] | Query preserving approach | RDBMS and XML doc | - | No |  |

**TABLE 7.** Distortion-free Watermarking Techniques ('-' indicates '*not mentioned*').

| Category | Paper | Brief Overview | Data Set used | Attacks analysed | Selected for experiment | Selection Criteria |
|---|---|---|---|---|---|---|
| Tuple Reordering | Li et al. [57] | Tuples are divided into groups by using hash function and for each tuple pair in a group, the order of the tuples are changed according to their tuple hash values and the watermark bit | - | insert, delete, value modification | Yes | Criteria 1 |
| | Bhattacharya et al. [58] | Extends the proposal in [57] | - | tuple update, tuple deletion, tuple insertion | No | |
| | Kamel [59] | Extends the proposal in [57] | Real life data sets representing roads, regions and streams in US | Deletion, insertion, modification | No | |
| | Li et al. [60] | Extends the proposal in [57] | real database related to baseball | - | No | |
| | Arun et al. [61] | Extends the proposal in [57] | - | - | No | |
| | Kamel et al. [62] | The tuple ordering is done on the basis of value of critical attribute(s) and re-arrangement is done relative to a secret initial order. | Synthetic data | random value updation | No | |
| Binary Form Relation | Li et al. [63] | MSBs of attribute values are used to generate the watermark. | - | attribute insertion, tuple insertion, value modification, deletion | Yes | Criteria 3 |
| | Bhattacharya et al. [64] | Extends the proposal in [63] but it considers private key instead of public. | - | tuple deletion, insertion | Yes | |
| | Halder et al. [65] | Extends the proposal in [63] | - | - | No | |
| | Halder et al. [66] | Extends the proposal in [63] | - | - | No | |
| | Bhattacharya et al. [67] | A fixed number of MSBs and LSBs of the selected filed are used for generating the watermark. | - | subset deletion, subset addition | No | |
| Attribute Reordering | Hamadou et al. [68] | Attributes are virtually sorted on hash values of attribute names to define a secret initial order of attributes. | same as [2] | attribute value updation, tuple insertion, tuple deletion | Yes | Single work in this category. |
| Content characteristics Based | Camara et al. [69] | Partitions the data into square matrices and matrix operations are used to generate the watermark. | same as [2] | Tuple deletion, tuple insertion, attribute insertion, multifaceted | Yes | Both are significant works in this category. |
| | Khan et al. [70] | Based on local characteristics like frequency distribution of digits, lengths and ranges of data values. | same as [2] | Insertion, update, deletion | Yes | |
| Others | Darwish et al. [71] | After grouping of tuples, three fake tuples are generated for each groups: first tuple is generated based on hash function and to other tuples are created using genetic algorithm. | real database | subset deletion, subset addition, subset modification | No | Criteria 3 |
| | Rani et al. [72] | Adapted MapReduce paradigm for watermarking large relational databases. | same as [2] | - | No | |
| | Siledar et al. [73] | Generates an image from the database content. | same as [2] | insertion, deletion, alteration | No | |
| | Kamel et al. [74] | Each column is organized into groups and the data values are reordered corresponding to a watermark value. | Synthetic data and Forest CoverType data set | Modification, superset, deletion | No | |
| | Naz et al. [75] | Data values are grouped and the group watermark is generated by extracting $\mu$ MSBs of hash of attribute names. | Patient's medical record | update, insert, delete | Yes | |
| | Shah et al. [76] | watermarking is done by adjusting the text case of selected data values | US Medicare Plan 2008 database | multiple value modification, multiple tuples insertion and multiple tuples deletion | No | |

#### b: CONVERTING DATABASE RELATION INTO BINARY FORM

These techniques convert the database relation into a binary form. In [63], the watermark is generated from the most significant bits (MSBs) of the attribute values and can be verified publicly. In [64], the watermark can not be verified publicly as it uses a private key. The approaches in [65], [66] also, extend the approach of [63]. In [67], tuples are first grouped, then a fixed number of MSBs and LSBs of the selected attribute value are used to generate the watermark.

#### c: ATTRIBUTE REORDERING

Authors in [68] have proposed a fragile distortion-free watermarking technique based on the attribute reordering method. First, a secret initial order of attributes is defined by virtually sorting the attributes based on the hash of attribute names. Thereafter, the MSBs are extracted for generating the watermark.

#### d: CONTENT CHARACTERISTICS BASED WATERMARKING

The watermarking approach in [70] generates the watermark based on the local characteristics like frequency distribution of various digits, lengths, and ranges of data values. In [69], the data set is grouped as the square matrices and the watermark is generated using the determinant and the minor of those square matrices.

#### e: OTHERS

We have classified the recent works in distortion-free watermarking in this category. The significant recent research works are proposed in [71]–[76]. In the proposed scheme in [71], some secure parameters are used to partition the tuples and three fake tuples are generated for each partition. A hash function is used to generate the first tuple. For the other two tuples, a genetic algorithm is used for numeric attributes, and for non-numeric attributes, the most frequent value is selected. These fake tuples are stored in a separate file, not inside the database, therefore making this approach distortion-free. Authors in [72] have adapted the MapReduce paradigm for watermarking of relational databases to decrease the computational cost and have implemented distortion-free algorithms in both sequential and MapReduce form. The proposal in [73] generates an image as a watermark from the database content. In [74], each column (attribute) is organized into groups, each having $g$ data elements. The data elements in each group are re-ordered based on a watermark value. In [75], the data elements are grouped and the group watermark is generated by extracting $\mu$ MSBs of the hash of attribute names. They present the proposed watermarking as a service (WaaS) scheme.

A brief overview of different distortion-free watermarking techniques within each category is depicted in Table 7.

It is to observe that the reversible database watermarking techniques [4], [5], [77]–[105] as depicted in Table 8 have a wider scope of research and we would like to explore these techniques in the future separately.

Figure 2 provides a quick reference on the classification of different relational database watermarking algorithms.

## III. COMPARATIVE PERFORMANCE ANALYSIS OF DISTORTION-BASED WATERMARKING TECHNIQUES

We select the distortion-based algorithms proposed in [2], [22], [28], [40], [41], [46], [48], [49] for the experimental analysis. We implement all the algorithms using Java. The experiments are performed on a server equipped with six core Intel Xeon Processor, 2.4 GHz Clock Speed, 128 GB RAM and Linux Operating System. We use benchmark data sets obtained by modifying the Forest CoverType data set[1] into data sets of size 276MB, 532MB, 888MB, 1124MB, 1338MB, 1692MB, and 2237MB and perform the following analysis:

1) We analyze the usability of the watermarked databases in terms of differences between the mean and variance of attribute values, before embedding of watermark and after embedding.
2) We also analyze the watermark embedding and detection time by increasing the data set size.
3) We perform the robustness analysis of these techniques over various attacks, e.g. insertion, update, delete, zero out, and multifaceted attack.

The prime reason behind choosing this data set is its wide consideration by the majority of the proposals in the literature. In particular, 33 out of 94 research works considered this data set as their benchmark, whereas the rest of the proposals used either a different kind of real-world data or self-generated data which differ from one proposal to another. This makes it difficult to compare them empirically uniformly. In order to unify the comparative analysis, in this paper, we consider this most popular Forest CoverType data set as the benchmark for all the proposals under our consideration.

### A. COMPUTATIONAL TIME

In database watermarking, the time spent during watermark generation and detection is an important factor to consider. The watermark embedding and detection time for various approaches is shown in Table 9. The comparison of watermarking time for these techniques is depicted in Figure 3.

From Table 9 and Figure 3, we have the following observations:

1) For all algorithms, watermark embedding and detection time increase as the data size increases.
2) The watermark embedding and detection time are least in the case of [48].
3) The watermark embedding and detection time are highest in the case of [22].
4) The order of computational cost from lowest to highest is: [48] < [28] < [2] < [49] < [46] < [40] < [41] < [22].

[1] https://kdd.ics.uci.edu/databases/covertype/covertype.html

**TABLE 8.** Reversible Watermarking Techniques ('-' indicates '*not mentioned*').

| Paper | Brief Overview | Data Set used | Attacks analysed |
|---|---|---|---|
| Siledar et al. [77] | Based on quadratic difference expansion | Indian Liver Patient data set | Insertion, deletion, modification |
| Hou et al. [78] | Quality of watermarked data is used to claim copyright | Generated data | Tuple deletion, tuple addition, tuple alteration |
| Lin et al. [79] | Two different secret embedding keys are generated | Generated data | Alteration, deletion, mix-match, sorting, combination |
| Shen et al. [80] | Clustering-based and difference expansion technique is used | Generated data | Tuple delete, tuple modification |
| Li et al. [81] | Embeds the watermark bit by bit on the basis of grouping | Same as [2] | Insertion, deletion, modification |
| Lian et al. [82] | Differential expansion technology based on ant colony algorithm | Same as [2] | Subset deletion, modification |
| Li et al. [83] | Based on continuous columns in histogram | Same as [2] | Insert, delete, alter |
| Hamadou et al. [84] | Prediction-error expansion method | Same as [2]. | Attribute Alteration, tuple deletion, tuple insertion |
| Ge et al. [85] | Histogram shifting watermarking method | Wisconsin breast cancer diagnosis data set | Tuple addition, tuple deletion, attribute value modification |
| Tufail et al. [86] | Binary Bat Algorithm used for watermark creation | Heart disease medical data set | Insertion, deletion, alteration |
| Chai et al. [87] | Based on clustering grouping | Same as [2] | Attribute modification or deletion, subset deletion, subset addition, subset alteration |
| Chai et al. [88] | Based on erasure code | Same as [2] | Attribute modification or deletion, tuple deletion, tuple addition, subset alteration |
| Wu et al. [89] | Difference-expansion reversible data hiding method is used | Protected numeric data | - |
| Li et al. [90] | Based on histogram gap low distortion | Same as [2] | Insertion, deletion, modification |
| Hu et al. [91] | Genetic Algorithm with a new proposed Histogram Shifting of prediction error Watermarking (HSW) method to minimize distortion | Same as [2] | Insertion, deletion and alteration |
| Imamoglu et al. [92] | Difference expansion watermarking (DEW) with Firefly Algorithm is used to embed watermark | Same as [2] | Addition, deletion, bit-flipping, tuple-wise-multifaceted, attribute-wise-multifaceted, sorting |
| Chang et al. [93] | Watermark is embedded into the textual relational database | Textual relational database | Sorting, deletion, modification, addition |
| Chang et al. [94] | The content of textual attributes are used to generate the virtual primary attribute | Synthetic data | Tuple deletion, tuple alteration, tuple insertion |
| Iftikhar et al. [95] | Genetic Algorithm is used for getting optimal watermark information | Heart disease medical data set | Insertion, deletion, alteration |
| Chang et al. [96] | Embeds the watermark into the fractional portion of the numerical attributes to minimize the distortion | Generated database | Alteration, deletion, mix-match, sorting |
| Jawad et al. [97] | Genetic algorithm is used to improve the capacity of difference expansion based watermarking in databases | Same as [2] | Addition, deletion, bit flipping, sorting, tuple-wise-multifaceted, attribute-wise-multifaceted, secondary watermarking |
| Farfoura et al. [98] | An identification image is converted into a stream of bits 0's, and 1's and embedded into numeric attributes | Synthetic data | Deletion, insertion, modification |
| Farfoura et al. [99] | Time-stamping protocol is used | Synthetic data | Tuple alteration, tuple deletion, mix and match, attribute deletion |
| Contreras et al. [100] | Based on a circular representation of a bijective transformation | Medical database | Modification of attribute values, elimination or insertion of tuples |
| Gupta et al. [101] | Difference expansion on integers is used to achieve reversibility | Generated database | Random bit wise flipping, subtractive, sorting, secondary watermarking |
| Gupta et al. [102] | Query-preserving watermarking scheme is proposed. | - | - |
| Zhang et al. [103] | Based on expansion on data error histogram | Generated database | - |
| Gupta et al. [104] | Based on difference expansion | Same as [101] | Same as [101] |
| Unnikrishnan et al. [105] | Based on orthogonal learning particle swarm optimization | Synthetic data | insertion, deletion and alteration |

**TABLE 9.** Watermark generation and detection time (in milliseconds).

| *Size of Data (in MB)* | AHK 2002 [2] | | Li 2003 [22] | | Prasanna. 2009 [46] | | Zhang 2011 [28] | | Kamran 2013 [40] | | FieGuo 2006 [49] | | Li 2005 [48] | | Huang 2004 [41] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ |
| 276 | 44593 | 36333 | 238693 | 225158 | 51900 | 57727 | 38693 | 29089 | 94193 | 87239 | 44270 | 42005 | 16589 | 5677 | 107848 | 107039 |
| 532 | 101587 | 72430 | 455016 | 429327 | 113053 | 92188 | 71165 | 57295 | 193080 | 169147 | 93525 | 72687 | 29355 | 9897 | 314846 | 276820 |
| 888 | 142363 | 119988 | 613255 | 586367 | 213112 | 146547 | 138523 | 100428 | 320350 | 287711 | 146400 | 91233 | 50078 | 21219 | 359679 | 332385 |
| 1124 | 178702 | 158994 | 887012 | 859342 | 213136 | 181424 | 139666 | 104371 | 367192 | 336507 | 198813 | 151873 | 63113 | 19883 | 519823 | 525728 |
| 1338 | 234311 | 174010 | 945584 | 883639 | 285000 | 219363 | 169044 | 128484 | 502728 | 427423 | 252000 | 176136 | 70664 | 33010 | 536742 | 530287 |
| 1692 | 292721 | 228411 | 1318398 | 1292265 | 312628 | 268167 | 205991 | 166133 | 543450 | 482110 | 291490 | 229154 | 83886 | 27686 | 926221 | 898700 |
| 2237 | 354479 | 287743 | 2168703 | 1430286 | 464093 | 370848 | 278627 | 214460 | 818662 | 694515 | 696066 | 422730 | 118508 | 51731 | 933961 | 910995 |

There are many operations that may affect the computational time. We identify these operations as partitioning, hash calculation, random number generation, virtual primary key generation, updating the attribute value. Other parameters

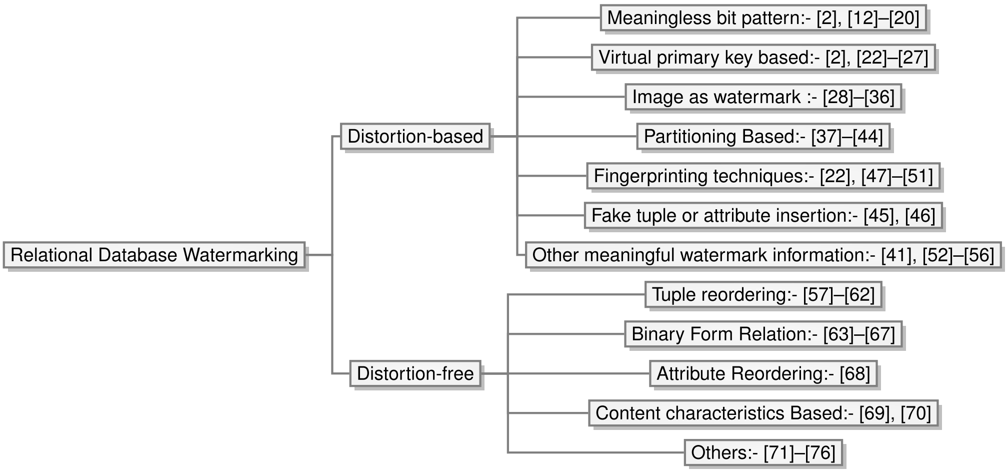
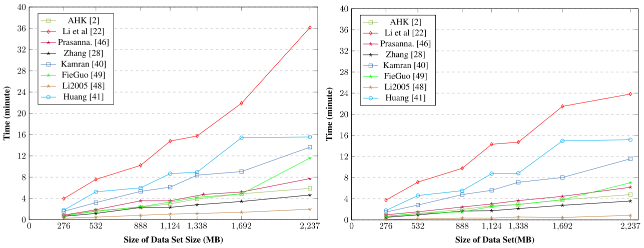

Relational Database Watermarking
Distortion-based
Meaningless bit pattern:- [2], [12]–[20]
Virtual primary key based:- [2], [22]–[27]
Image as watermark :- [28]–[36]
Partitioning Based:- [37]–[44]
Fingerprinting techniques:- [22], [47]–[51]
Fake tuple or attribute insertion:- [45], [46]
Other meaningful watermark information:- [41], [52]–[56]
Distortion-free
Tuple reordering:- [57]–[62]
Binary Form Relation:- [63]–[67]
Attribute Reordering:- [68]
Content characteristics Based:- [69], [70]
Others:- [71]–[76]

**FIGURE 2.** Classification of relational database watermarking techniques.

**TABLE 10.** Changes in variance after watermarking.

| Attri-butes | Vari-ance | AHK 2002 [2] | | | Li 2003 [22] | | | FieGuo 2006 [49] | | | Zhang 2011 [28] | | | Huang2004 [41] | | | Kamran 2013 [40] | Prasanna [46] | Li 2005 [48] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\xi$=3 | 5 | 8 | 3 | 5 | 8 | 3 | 5 | 8 | 3 | 5 | 8 | 3 | 5 | 8 | | | |
| A1 | 27 | | | +1 | -3 | -3 | $-17$ | | | | +281 | +143 | +168 | | | | | No Change | No Change |
| A2 | 96 | | | | | | | | | | | | | | | | | | |
| A3 | 150 | | | | | | | | | | | | | | | | | | |
| A4 | 67 | | | | | | | | | | +1 | +3 | +23 | | | | | | |
| A5 | 235 | | | | | | | | | | | | | | | | | | |
| A6 | 2 | | | | +7 | +2 | +16 | +14 | +19 | +99 | +347 | +202 | +336 | +20 | +20 | +171 | +222 | | |
| A7 | 19 | | | | | | | | | | | | | | | | | | |
| A8 | 201 | | | | | | | | | | | | | | | | | | |
| A9 | 255 | +1 | | +1 | +1 | | +1 | | | | | | | | | | | | |
| A10 | 192 | | | | | | | | | | -84 | -104 | +142 | | | | | | |



(a) Watermark Generation Time (b) Watermark Detection Time

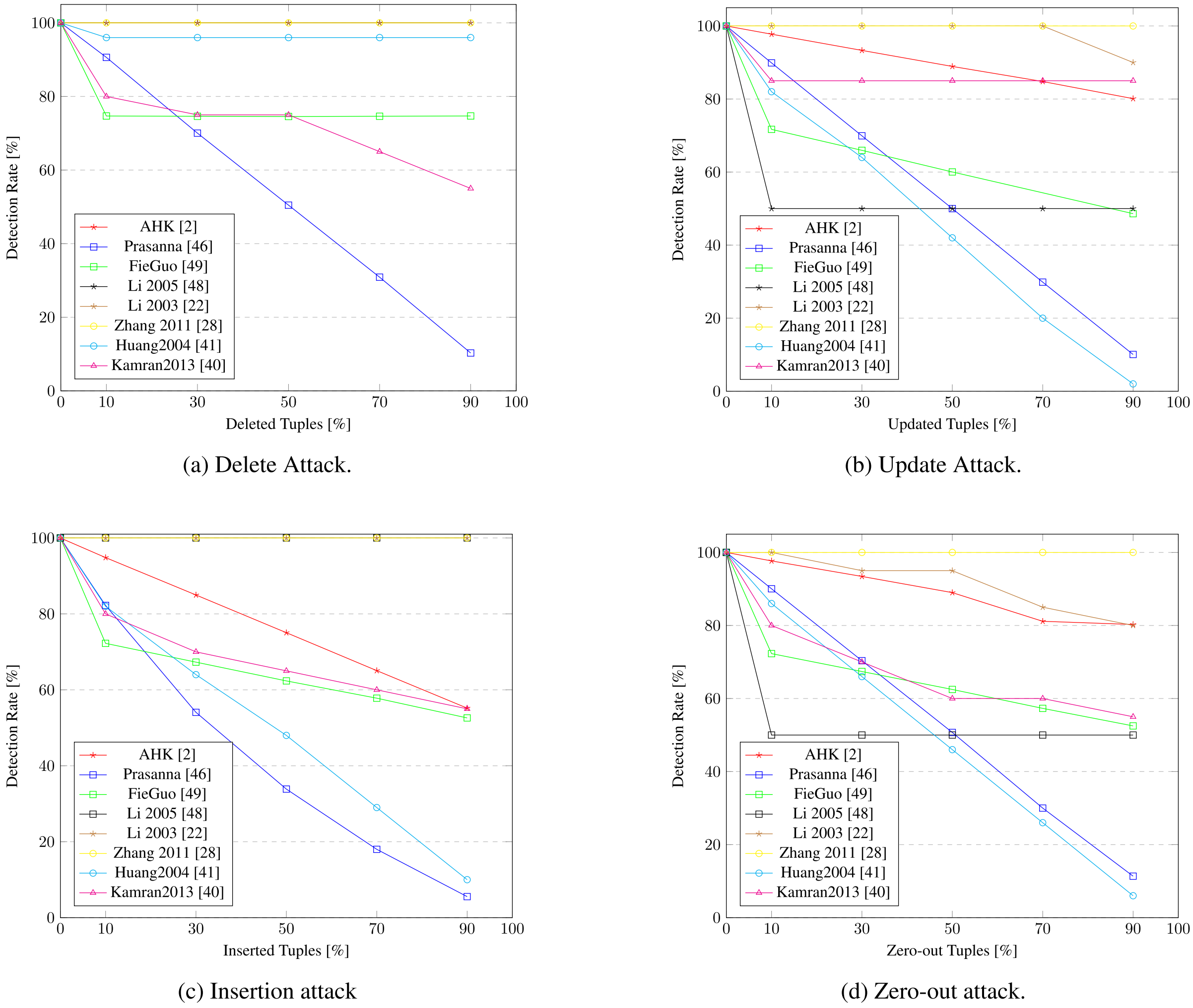
**FIGURE 3.** Comparison of watermark generation and detection time for distortion-based techniques.

like the number of attributes and the number of tuples considered for watermark embedding also affect the computational time. We observe that the computational cost is highest in the case of [22] because it generates a virtual primary key for each of the tuples in the data set and therefore it takes more time. Whereas, the approach in [48] has least computational cost as it generates some random numbers instead of hash computation. The computational cost is less in the case of [2], [28]. In the case of [28], after partitioning only some of the tuples and attributes satisfying a criterion are considered for embedding the watermark. In the case of [2], the tuples are not partitioned, but only a fraction ($\gamma$) of tuples satisfying a particular condition are considered. The LSBs of one attribute in each selected tuple are flipped based on the watermark bits. The approach in [49] also considers only one fixed attribute in each partition to embed the watermark. Partitioning is a common operation in case of [40], [41], [46], [49] and all the approaches are having more computational cost after the approach in [22]. Therefore, the partitioning operation is affecting the computational cost.

### B. USABILITY OF DATA AFTER WATERMARK EMBEDDING

The usability of the database is based on the domain, e.g., a minor change in a voter database can create a problem, and hence the watermarking should not cause any changes to the voter database, whereas, minor changes in a forest survey database can be tolerated. Therefore, it is difficult to generalize the criteria for usability. However, Table 10 will help the users to understand the effect of watermark embedding on the mean and variance values of the attributes and give them an idea about whether watermarking causes more changes in the underlying data or not. Table 10 shows the change in variance of database values before embedding of watermark and after embedding. The watermark embedding algorithms in [2], [22], [28], [41], [49] embed the watermark bit in a particular bit of the attribute value. The number of bits available for watermark embedding is denoted by the variable $\xi$. We compute the variance of each attribute by varying the value of $\xi$ to 3, 5, and 8. We observe that there is no change in mean after watermark embedding.

In distortion-based watermarking, it is assumed that a certain level of distortion is tolerable. In the case of [2], there



(a) Delete Attack.

(b) Update Attack.

(c) Insertion attack

(d) Zero-out attack.

**FIGURE 4.** The rate of detection after various attacks in case of distortion-based techniques.

are small changes in variance when the bits available for embedding are increased to 8 bits. In the case of [46] and [48], there is no change in the variance at all, whereas in the case of [22] and [28], the variance is highly affected after the watermark embedding.

From figure 10, we observe that in the case of approaches in [40], [41], [49] the variance of only one attribute is affected after watermark embedding because they only consider a single attribute to embed the watermark. In the case of [46], there is no change or negligible change in the variance as it inserts a new tuple into the database relation. Therefore, it does not cause any change in the attribute values and the variance of attributes is not affected. Similarly, in the case of [48], there is no change in the variance and in the case of [2], there is negligible change as both of the algorithms embed the watermark into a fraction of tuples. The usability is highly affected in the case of [28] because an attribute is selected for embedding if the length of the attribute value is greater than a particular value. This causes the watermark to be embedded in more than one attributes.

### C. ROBUSTNESS ANALYSIS

We perform the robustness analysis of the watermarking techniques over various attacks, e.g. insertion, update, delete, zero out, and multifaceted attack. We analyze the rate of detection by varying the intensity of the attacks as 10%, 30%, 50%, 70%, and 90%.

#### 1) DELETE ATTACK

In delete attack, the attacker deletes some of the tuples of the watermarked database in order to distort the embedded watermark. Though the attacker is supposed to delete the tuples keeping in mind the usability of the data, we analyze the detection rate by varying the attack percentage from 10% to 90%. The rate of detection for various distortion-based techniques after delete attack is shown in Figure 4(a). From Figure4(a), we can observe that the rate of watermark detection remains more than 90% even after 90% delete in case of [2], [22], [28], [41], [48].

#### 2) UPDATE ATTACK

In an update attack, the attacker randomly updates some of the values of the watermarked database with his own values

**TABLE 11.** Intensity of attacks.

| Multifaceted attack | Delete Attack | Update attack | Insertion attack |
|---|---|---|---|
| **10%** | 3% | 3% | 4% |
| **30%** | 10% | 10% | 10% |
| **50%** | 16% | 16% | 18% |
| **70%** | 20% | 20% | 30% |
| **90%** | 30% | 30% | 30% |

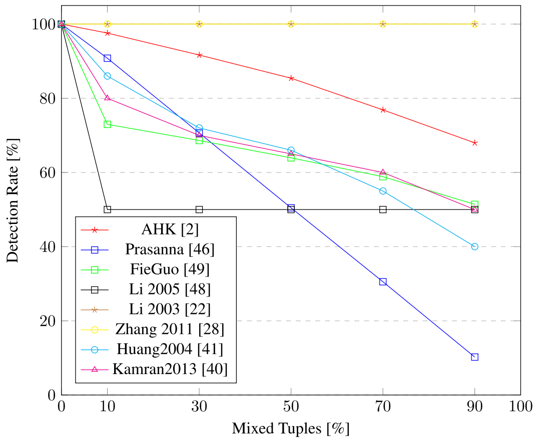


**FIGURE 5.** Rate of detection after multifaceted attack in case of distortion-based techniques.

and try to claim ownership of the database. We analyze the detection rate by varying the update percentage from 10% to 90% as depicted in Figure 4(b).

We can observe from Figure 4(b) that the rate of detection is more than 80% in case of [2], [22], [28], [40] even after 90% update.

### 3) INSERTION ATTACK

In an insertion attack, the attacker removes a particular number of tuples from the watermarked database and inserts the same number of tuples into the database to destroy the embedded watermark. The rate of watermark detection for various techniques after insertion attack is shown in Figure 4(c). We can observe from Figure 4(c) that the rate of detection is 100% in the case of [22], [28], [48] even after a 90% attack.

### 4) ZERO-OUT ATTACK

In this attack, some tuple values are selected randomly and updated with zero by the attacker to destroy the embedded watermark. We analyze the rate of watermark detection by varying the attack percentage as shown in Figure 4(d).

We can observe from Figure 4(d) that the rate of detection is more than 80% in case of [2], [22], [28] even after 90% attack.

### 5) MULTIFACETED ATTACK

This is the combination of delete, update, and insertion attacks. The attacker randomly updates some of the tuple values, deletes some of the tuples, and inserts his own tuples to destroy the embedded watermark.

**Algorithm 1** GEN_WM(R)

```
1: for each tuple $t \in$ R do
2:     Compute $g_t = f(t)$.
3: end for
4: for each group $g_i \in$ G do.        // G = total number of groups
5:     T = set of all tuples $t_i \in$ group $g_i$.
6:     Compute watermark $W_i$ for group $g_i$ using T.
7: end for
8: Compute W = $||W_i, \quad \forall\, i = 1$ to $|G|$.
```

The data usability is highly impacted by this attack. The intensity of these attacks that we have considered is shown in Table 11. The rate of detection after the multifaceted attack is depicted in Figure 5. We can observe from Figure 5 that the rate of detection is 100% in case of [22], [28] even after 90% attack.

The robustness against various attacks is more in the case of [2] and [48] since the detection in [2] is based on the match counts that are computed on the remaining watermarked tuples after the attack. Similarly, in the case of [48], the detection is based on the majority voting for each fingerprint bit and form the remaining watermarked tuples after attacks, the fingerprint can be recovered.

## IV. COMPARATIVE PERFORMANCE ANALYSIS OF DISTORTION-FREE WATERMARKING TECHNIQUES

The data values in the database are not changed in the case of distortion-free watermarking techniques. These techniques mainly generate the watermarks from the database contents. The primary phases in these techniques are (i) Partitioning of tuples into groups, and (ii) Watermark generation from each group. The watermarks for each group can be combined together to generate the watermark for the whole database.

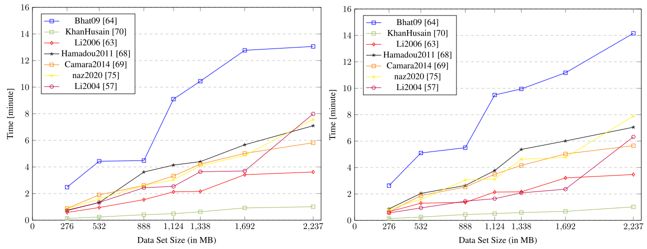
A generic distortion-free watermarking algorithm GEN_WM is shown in Algorithm 1. The database relation R is taken as input. Steps 1 to 3 compute the group id $g_t$ to which a tuple $t$ belongs by applying function $f$ (e.g. a hash function). In Steps 4 to 6, a group watermark $W_i$ for the group $g_i$ is generated by using the tuples belonging to group $g_i$. Step 8 computes the overall watermark W by performing a suitable operation || (e.g. a concatenation operation) to the group watermarks.

Authors in [57] proposed the first work in this domain. We classify the distortion-free watermarking techniques in the following categories: (i) permutation of tuples, (ii) converting database relation into binary form, (iii) attribute reordering, (iv) content characteristics based, and (v) others. We analyze these techniques and select the algorithms for experimental analysis on the basis of the same criteria as discussed in Section II.

We consider the distortion-free watermarking algorithms proposed in [57], [63], [64], [68]–[70], [75] and perform the robustness analysis of these techniques over various attacks,

**TABLE 12.** Watermark generation and detection time (in milliseconds).

| *Size of Data(in MB)* | **Li 2004 [57]** | | **KhanHusain [70]** | | **Li Deng 2006 [63]** | | **Hamadou2011 [68]** | | **Camara2014 [69]** | | **Bhat2009 [64]** | | **Naz 2019 [75]** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ | $T_e$ | $T_d$ |
| 276 | 43650 | 33602 | 7999 | 8028 | 34700 | 39179 | 48708 | 52609 | 53473 | 38106 | 149124 | 158174 | 52364 | 50258 |
| 532 | 79851 | 56447 | 14592 | 15170 | 57305 | 78222 | 109594 | 122486 | 113383 | 112954 | 265919 | 306047 | 95609 | 95484 |
| 888 | 146475 | 87114 | 24915 | 26482 | 92723 | 82163 | 217412 | 158838 | 156892 | 151359 | 269187 | 330152 | 156754 | 183296 |
| 1124 | 152515 | 98501 | 29704 | 30907 | 128262 | 128955 | 248636 | 226474 | 199280 | 209280 | 546589 | 569417 | 181938 | 187764 |
| 1338 | 218518 | 125221 | 38276 | 35905 | 129633 | 129667 | 264569 | 322607 | 253458 | 249865 | 627592 | 597004 | 245983 | 278160 |
| 1692 | 222053 | 141930 | 43006 | 40907 | 204966 | 192796 | 340214 | 360626 | 301493 | 301982 | 766220 | 670200 | 292440 | 285779 |
| 2237 | 479433 | 379725 | 60723 | 61550 | 217652 | 208221 | 426348 | 423525 | 349532 | 339458 | 783832 | 849842 | 454534 | 473980 |



(a) Watermark Generation Time (b) Watermark Detection Time

**FIGURE 6.** Comparison of watermark generation and detection time for distortion-free techniques.

e.g. insertion, update, delete, zero out, and multifaceted attack. We also analyze the computational cost. We implement all the algorithms using Java. The experiments are performed on a server equipped with six-core Intel Xeon Processor, 2.4 GHz Clock Speed, 128 GB RAM, and Linux Operating System. We use benchmark data sets obtained by modifying the Forest CoverType data set[2] into data sets of size 276MB, 532MB, 888MB, 1124MB, 1338MB, 1692MB and 2237MB. The reason for choosing this data set is discussed in Section III.

### A. COMPUTATIONAL TIME

In database watermarking, the time spent during watermark generation and detection is an important factor to consider. The watermark generation and detection time for various approaches is shown in Table 12. The comparison of watermarking time for these techniques is depicted in Figure 6.

Following are the observations from Table 12 and Figure 6:

1) For all the watermarking approaches, watermark embedding and detection time increases as the data size increases.
2) Watermark generation and detection time is least in case of [70] and highest in case of [64].

Authors in [72] adapted the MapReduce paradigm to watermark relational databases. They have implemented the algorithms proposed in [57], [64], [67], [69], [70] in sequential as well as MapReduce form and it was observed that as the data size increases, the percentage reduction in watermarking time increases from sequential to MapReduce.

In the case of distortion-free watermarking techniques, there are various operations that affect the computational cost, e.g. hash computation, partitioning, watermark generation, pseudo-number generation, matrix operations, etc. The number of attributes, tuples, and the bit positions available for watermark generation also affects the computational cost. From Figure 6, we can observe that the computational time is highest in the case of [64] since it partitions the database relation based on the hash function and uses all attributes of all tuples for generating a binary form of the relational database. The computational cost is least in case of [70], since it does not partition the database relation. The watermark is generated by considering all attributes of all tuples and by generating digit, length, and frequency sub-watermarks. The basic step in the case of distortion-free technique is partitioning. For example, the approaches in [57], [64], [68], [69], [75] partition the data based on either hash function, pseudo-random number, etc. The group watermarks are then generated independently.

### B. USABILITY OF DATA AFTER WATERMARK GENERATION

In the case of distortion-free watermarking approaches, the watermark is generated from the underlying content of the data and there is no distortion in the data itself, hence the data usability is not affected.

### C. ROBUSTNESS ANALYSIS

We perform the robustness analysis of the watermarking techniques over various attacks, e.g. insertion, update, delete, zero out, and multifaceted attack. We analyze the rate of detection by varying the intensity of the attacks from 10% to 90%.

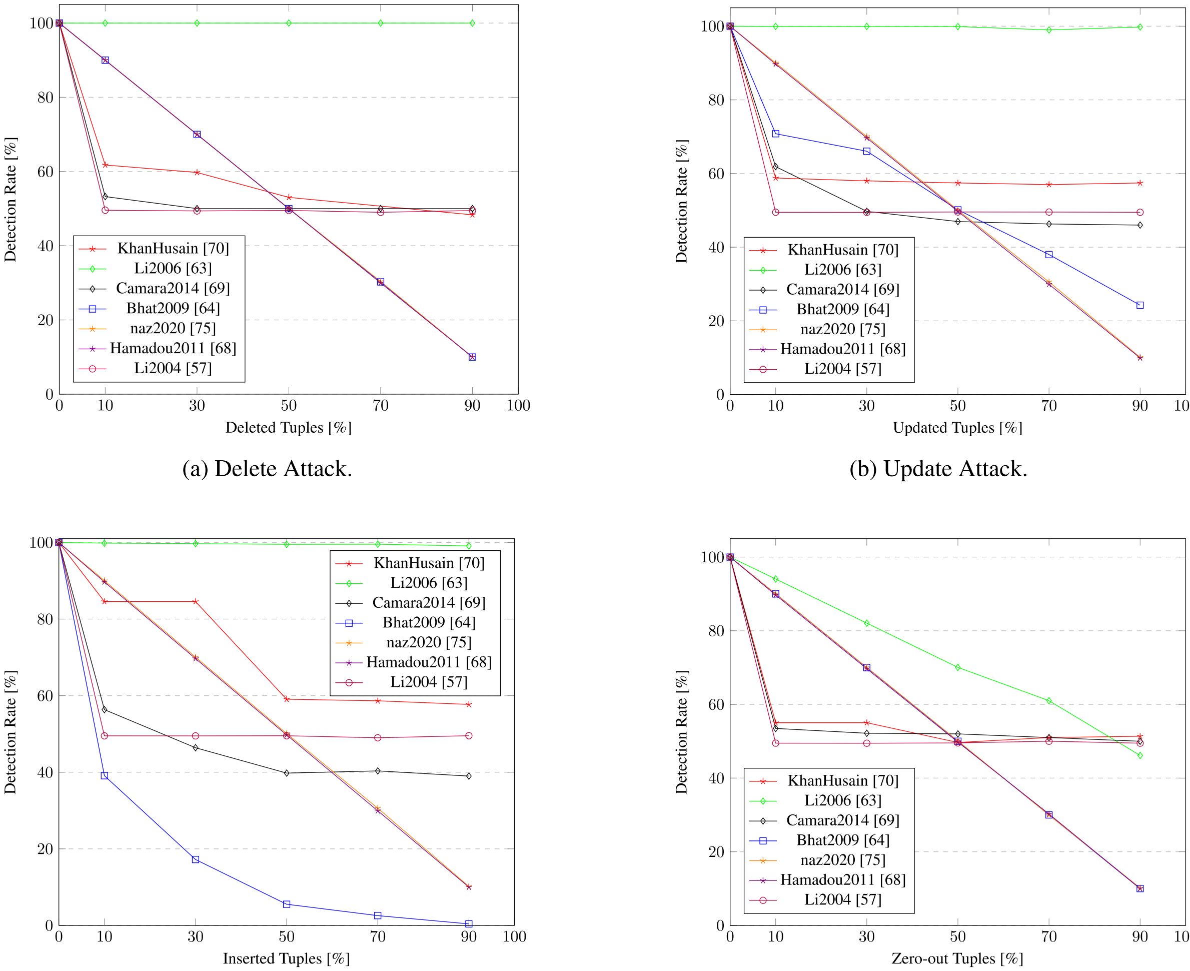
#### 1) DELETE ATTACK

In a delete attack, some of the tuples of the watermarked database are deleted by the attacker in order to distort the watermark. Though the attacker is supposed to delete the tuples keeping in mind the usability of the data, we analyze the detection rate by varying the attack percentage from 10% to 90%. The rate of detection for various distortion-based techniques after delete attack are shown in Figure 7(a). From Figure 7(a), we observe that the rate of detection remains 100% in case of [63] even after 90% attack.

#### 2) UPDATE ATTACK

In an update attack, the attacker randomly updates some of the values of the watermarked database with his own values and tries to claim ownership of the database. We analyze the

[2] https://kdd.ics.uci.edu/databases/covertype/covertype.html



(a) Delete Attack.

(b) Update Attack.

(c) Insertion attack

(d) Zero-out attack.

**FIGURE 7.** The rate of detection after various attacks in case of distortion-free techniques.

detection rate by varying the update percentage from 10% to 90% as shown in Figure 7(b). We observe that the rate of detection remains 100% in the case of [63] even after a 90% attack.

### 3) INSERTION ATTACK

In an insertion attack, the attacker removes a particular number of tuples from the watermarked database and inserts the same number of tuples into the database to destroy the watermark. The rate of watermark detection for various techniques after insertion attack is depicted in Figure 7(c). We observe that the rate of detection remains 100% in the case of [63] even after 90% attack.

### 4) ZERO OUT ATTACK

Some of the tuple values of the watermarked database are randomly selected by the attacker and updated with zero to destroy the watermark. We analyze the rate of watermark detection by varying the attack percentage as shown in Figure 7(d). The rate of detection even after a 90% attack is highest in the case of [63].

### 5) MULTIFACETED ATTACK

This is the combination of delete, update, and insertion attacks. The attacker randomly updates some of the tuple values, deletes some of the tuples, and inserts his own tuples to distort the watermark.

The intensity of the update, delete, and insertion attacks are taken as shown in Table 11. We analyze the rate of detection after the multifaceted attack in Figure 8. The rate of detection remains near 100% in the case of [63] even after a 90% attack.

From Figure 7 and Figure 8, we observe that the approach in [63] has the highest robustness against four types of attacks since it considers the number of attributes as that of binary attributes ($\gamma$) present in the database relation. It generates the watermark bits from the MSB positions of the attribute values. If the value of $\gamma$ is increased, though it will increase the robustness, the computational cost will be increased.

## V. DISCUSSION

While comparing our evaluation-results with the results reported in the existing papers, we draw the following observations:

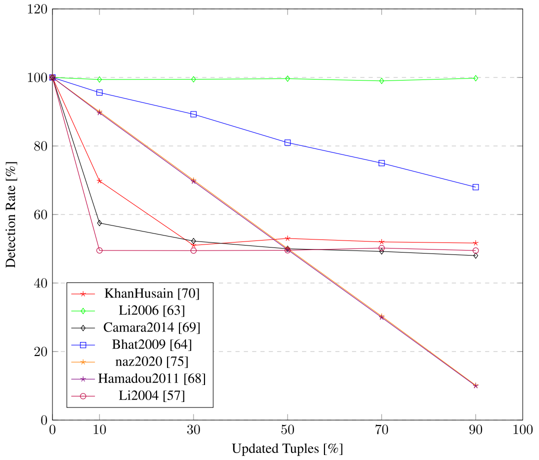


**FIGURE 8.** Rate of detection after multifaceted Attack in case of distortion-free algorithms.

### A. COMPUTATIONAL COST

Although watermark-embedding and detection times have significance when to apply in case of large-scale data set, none of the existing proposals (except [2]) under distortion-based approaches performs this evaluation. To the best of our knowledge, this paper first reports a detailed comparative study on the computational costs incurred by different algorithms under consideration. Under distortion-free approaches, only one proposal [75] evaluates its performance on patient's medical data achieving the watermark embedding and detection time of 13 and 21.1 seconds respectively. Even though [75] considers data set different from CoverType, we observe a linear growth of computational time in both embedding and detection phases similar to [2], [75].

### B. DATA USABILITY

Experimental evaluation on data usability using CoverType data set is being conducted by the authors in [2], [48], [49], whereas [40] considers a data set comprising consumers' power consumption rates. Like the results reported in [2], [48], [49], our evaluation results also reveals the similar fact that there is no notable change in the mean value of the data after watermark embedding, while very little change in the range 1-99 is observed in case of variance when more number of LSBs (e.g., 8 bits) for watermark embedding is considered. On the other hand, when we conduct experiments for the algorithms in [40] on CoverType data set, we observe a significant increase in the variance and little decrease in the mean values than that reported in [40] on power consumption data. This is due to the difference in the semantic domains of the attributes used for watermark-embedding in case of two different data sets. Note that distortion-free approaches do not suffer from this issue.

### C. ROBUSTNESS

Attack analyses to manifest the robustness of the algorithms are being conducted over CoverType data set in [2], [22], [48], [49], [68]–[70]. Interestingly, we gain a similar experience in our results also. To be more precise, in both the cases, the results show that the watermark can be detected even after a 90% attack. On the other hand, the attack analysis of the algorithms in [28], [40], [75] are performed on data sets different from CoverType. This is worthwhile to mention that the result reported in [28], [40] is similar to the result obtained using CoverType in our case, which shows that the watermark detection rate is above 70% even after a 90% attack. Similarly, the attack result reported in [75] exhibits similar trend as we observe in our case (on CoverType data), which show that the watermark-detection rate drops below 20% after a 90% attack.

## VI. OPEN ISSUES AND FUTURE DIRECTIONS

In this section, we discuss in detail the guidance to the users for a wise decision on choosing the right watermarking technique. We observe that various operations and parameters (such as the number of attributes, tuples, and bit positions for embedding or generation) in the watermarking algorithms impact the computational cost, data usability, and robustness. Few observations are listed below:

### A. THE NUMBER OF ATTRIBUTES INVOLVED IN WATERMARK EMBEDDING

Some algorithms embed the watermark in all attributes of the database relation. Even though this increases the robustness, this may cause more distortion and may affect the usability with increased computational time.

### B. THE NUMBER OF TUPLES CONSIDERED FOR WATERMARK EMBEDDING

If all of the tuples are considered for embedding the watermark, then it will increase the computational cost. It will also affect the usability more, though the robustness may be increased. Whereas, some watermarking algorithms consider a fraction of tuples for embedding the watermark. This will decrease the computational cost and the data usability will be less affected.

### C. THE NUMBER OF BITS AVAILABLE FOR WATERMARKING

If the number of bits considered for embedding watermarks is increased, it will increase the distortion. The computational time and robustness will not be affected much by this.

### D. PARAMETERS PARTICULARLY AFFECTING THE COMPUTATIONAL COST

There are many operations that may affect the computational time. We identify these operations as: partitioning, hash calculation, random number generation, virtual primary key generation, matrix operations, updating the attribute value.

Although we can not generalize, we categorize the usability, computational time, and robustness towards attacks for the relative comparison of various watermarking techniques

**TABLE 13.** A comparative summary of distortion-based techniques.

| Approach | Usability Affected | Computational Cost | Robustness Against Attacks | | | | |
|---|---|---|---|---|---|---|---|
| | | | Update | Delete | Insertion | Zero-out | Multifaceted |
| AHK [2] | Less | Less | Very High | Very High | High | Very High | High |
| Prasanna [46] | Less | Less | Less | Less | Less | Less | Less |
| Guo2006 [49] | High | Less | High | High | High | High | High |
| Li2005 [48] | Less | Less | Very High | Very High | Very High | High | High |
| Li2003 [22] | Very High | Very High | Very High | Very High | Very High | Very High | Very High |
| Zhang2011 [28] | Very High | Less | Very High | Very High | Very High | Very High | Very High |
| Kamran [40] | High | High | Very High | High | High | High | High |
| Huang2004 [41] | High | High | Less | Very High | Less | Less | Less |

**TABLE 14.** A comparative summary of distortion-free techniques.

| Approach | Usability Affected | Computational Cost | Robustness against various attacks | | | | |
|---|---|---|---|---|---|---|---|
| | | | Update | Delete | Insertion | Zero-out | Multifaceted |
| Li2004 [57] | No | Less | High | High | High | High | High |
| Bhat09 [64] | No | Very High | Less | Less | Less | Less | Less |
| Khan [70] | No | Less | High | High | High | High | High |
| Li2006 [63] | No | Less | Very High | Very High | Very High | High | Very High |
| Hamadou [68] | No | Less | Less | Less | Less | Less | Less |
| Naz2020 [75] | No | Very High | High | High | High | High | High |
| Camara [69] | No | Very High | High | High | Less | High | High |

in the following groups:

$$\texttt{Usability Affected}$$

$$= \begin{cases} \text{Very High}, & \text{if } \Delta\text{Variance} > \pm 10 \text{ in} \\ & \qquad > 2 \text{ attribute} \\ \text{High}, & \text{if } \Delta\text{Variance} \in (\pm 5 \text{ to } \pm 10) \\ & \qquad \text{in 1 or 2 attribute} \\ \text{Less}, & \text{if } \Delta\text{Variance} = 0 \text{ or } < \pm 5 \\ & \qquad \text{in 1 or 2 attribute}\} \end{cases}$$

$$\texttt{Computational Cost}$$

$$= \begin{cases} \text{Very High}, & \text{if computational time} > 10 \text{ minute} \\ \text{High}, & \text{if computational time} \in (\text{5-10 minute}) \\ \text{Less}, & \text{if computational time} < 5 \text{ minute} \end{cases}$$

$$\texttt{Robustness}$$

$$= \begin{cases} \text{Very High}, & \text{if rate of detection} > 80\% \\ \text{High}, & \text{if rate of detection} \in (50\text{-}80\%) \\ \text{Less}, & \text{if rate of detection} < 50\% \end{cases}$$

The $\Delta$Variance represents the change in variance of the attribute values after the embedding of the watermark. A comparative summary of the distortion based algorithms that we have considered for the experimental analysis is shown in Table 13.

The best algorithm should affect the usability "Less" after watermark embedding, have "Less" computational cost, and have "Very High" robustness against various attacks. In case of the distortion-based algorithms, if the usability is the main concern then the approaches in [2], [46], [48] are the better options since the attributes are having no change or negligible change in the variance after watermark embedding. If we consider the robustness and computational cost, then [2], [48] are better, but the approach in [46] has less robustness against all types of attacks. If only computational cost is considered, then the approach in [48] is having the least computational cost. If only robustness is considered, then the approach in [22] is the most robust, but the usability is highly affected after embedding. The computational cost is also highest in the case of [22] since it computes a virtual primary key for each of the tuples.

From Table 13, we can observe the following in the case of both [2] and [48]:

- The data usability is least affected after the watermark embedding.
- The computational cost is "Less".
- "Very High" robustness against three kinds of attacks.

Therefore, considering the usability constraints as defined, the computational cost and the robustness towards various attacks, we can say that the watermarking algorithms in [2] and [48] perform better than the other distortion-based watermarking algorithms we have considered for experimental analysis.

A comparative summary of the distortion-free algorithms that we have considered for the experimental analysis is shown in Table 14.

In the case of distortion-free watermarking techniques, if only computational cost is considered, then the approach in [70] is the best option as it takes the least watermarking time. If only the robustness against various attacks is considered, then the approach in [63] has a very high robustness in case of update, delete, insertion and multifaceted attacks. The usability is not affected, as the watermark generation process does not cause any distortion in the data.

From Table 14, we can observe the following in case of [63]:

- The usability of the data is not affected after the watermark generation.
- The computational cost is "Less".
- "Very High" robustness against four kinds of attacks.

Overall, considering the above-mentioned facts, the watermarking algorithm in [63] performs better than the other distortion-free watermarking algorithms in terms of computational-overhead and robustness.

## VII. CONCLUSION

In this paper, we perform a detailed comparative analysis of various relational database watermarking techniques empirically. We classify the existing distortion-based watermarking techniques into six categories, namely (i) meaningless bit-pattern as the watermark, (ii) virtual primary key based, (iii) image as watermark, (iv) partitioning based, (v) fake tuple/attribute insertion, (vi) fingerprinting techniques, and (vii) other meaningful watermark information. Similarly, the existing distortion-free techniques are classified as (i) permutation of tuples, (ii) conversion of the database into binary form, (iii) attribute reordering, (iv) content characteristics based, and (v) others. We perform an exhaustive empirical study and comprehensive analysis of a number of algorithms selected based on our quality-criteria. In particular, our evaluation focuses the following three crucial factors: computational cost, data usability, and robustness, as a way to provide an insightful guidance to choose the right watermarking technique for a given application.

## REFERENCES

[1] S. Khanna and F. Zane, "Watermarking maps: Hiding information in structured data," in *Proc. 11th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2000, pp. 596–605.

[2] R. Agrawal and J. Kiernan, "Watermarking relational databases," in *Proc. 28th Int. Conf. Very Large Databases*, 2000, pp. 155–166.

[3] R. Halder, S. Pal, and A. Cortesi, "Watermarking techniques for relational databases: Survey, classification and comparison," *J. Univ. Comput. Sci.*, vol. 16, no. 21, pp. 3164–3190, 2010.

[4] A. Khan, A. Siddiqa, S. Munib, and S. A. Malik, "A recent survey of reversible watermarking approaches," *Science*, vol. 279, pp. 251–272, May 2014.

[5] S. Iftikhar, M. Kamran, and Z. Anwar, "A survey on reversible watermarking techniques for relational databases," *Secur. Commun. Netw.*, vol. 8, no. 15, pp. 2580–2603, 2015.

[6] M.-R. Xie, C.-C. Wu, J.-J. Shen, and M.-S. Hwang, "A survey of data distortion watermarking relational databases," *Int. J. Netw. Secur.*, vol. 18, no. 6, pp. 1022–1033, 2016.

[7] S. Kumar, B. K. Singh, and M. Yadav, "A recent survey on multimedia and database watermarking," *Multimedia Tools Appl.*, vol. 79, pp. 20149–20197, Jul. 2020.

[8] M. Kamran and M. Farooq, "A comprehensive survey of watermarking relational databases research," 2018, *arXiv:1801.08271*.

[9] A. S. Alfagi, A. A. Manaf, B. Hamida, S. Khan, and A. A. Elrowayati, "Survey on relational database watermarking techniques," *ARPN-JEAS*, vol. 11, pp. 422–423, Oct. 2016.

[10] A. Alqassab and M. Alanezi, "Relational database watermarking techniques: A survey," *J. Phys., Conf. Ser.*, vol. 1818, no. 1, Mar. 2021, Art. no. 012185.

[11] V. Khanduja, "Database watermarking, a technological protective measure: Perspective, security analysis and future directions," *J. Inf. Secur. Appl.*, vol. 37, pp. 38–49, Dec. 2017.

[12] R. Agrawal, P. J. Haas, and J. Kiernan, "A system for watermarking relational databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2003, p. 674.

[13] J. Kiernan, R. Agrawal, and P. J. Haas, "Watermarking relational data: Framework, algorithms and analysis," *VLDB J. Int. J. Very Large Data Bases*, vol. 12, no. 2, pp. 157–169, Aug. 2003.

[14] Z. Qin, Y. Ying, L. Jia-jin, and L. Yi-shu, "Watermark based copyright protection of outsourced database," in *Proc. 10th Int. Database Eng. Appl. Symp. (IDEAS)*, Dec. 2006, pp. 301–308.

[15] G. Gupta and J. Pieprzyk, "Database relation watermarking resilient against secondary watermarking attacks," in *Proc. Int. Conf. Inf. Syst. Secur.* Berlin, Germany: Springer, 2009, pp. 222–236.

[16] X. Xiao, X. Sun, and M. Chen, "Second-LSB-dependent robust watermarking for relational database," in *Proc. 3rd Int. Symp. Inf. Assurance Secur.*, Aug. 2007, pp. 292–300.

[17] S. Rani, P. Kachhap, and R. Halder, "Data-flow analysis-based approach of database watermarking," in *Proc. Adv. Comput. Syst. Secur.* New Delhi, India: Springer, 2016, pp. 153–171.

[18] Y. Zhang, Z. Wang, Z. Wang, and C. Liu, "A robust and adaptive watermarking technique for relational database," in *Proc. 18th China Annu. Conf.* Singapore: Springer, Jul. 2021, pp. 3–26.

[19] S. Melkundi and C. Chandankhede, "A robust technique for relational database watermarking and verification," in *Proc. Int. Conf. Commun., Inf. Comput. Technol. (ICCICT)*, Jan. 2015, pp. 1–7.

[20] J. Lafaye, "An analysis of database watermarking security," in *Proc. 3rd Int. Symp. Inf. Assurance Secur.*, Aug. 2007, pp. 462–467.

[21] R. Halder, P. Dasgupta, S. Naskar, and S. S. Sarma, "An internet-based IP protection scheme for circuit designs using linear feedback shift register (LFSR)-based locking," in *Proc. 22nd Annu. Symp. Integr. Circuits Syst. Design Chip Dunes*, 2009, pp. 1–6.

[22] Y. Li, V. Swarup, and S. Jajodia, "Constructing a virtual primary key for fingerprinting relational data," in *Proc. ACM Workshop Digit. Rights Manage.*, 2003, pp. 133–141.

[23] C.-C. Chang, T.-S. Nguyen, and C.-C. Lin, "A blind robust reversible watermark scheme for textual relational databases with virtual primary key," in *Proc. Int. Workshop Digit. Watermarking*. Cham, Switzerland: Springer, 2014, pp. 75–89.

[24] V. Khanduja, S. Chakraverty, and O. Verma, "Ownership and tamper detection of relational data: Framework, techniques and security analysis," *Embodying Intell. Multimedia Data Hiding*, vol. 5, pp. 21–36, Dec. 2016.

[25] M. L. P. Gort, E. A. Diaz, and C. F. Uribe, "A highly-reliable virtual primary key scheme for relational database watermarking techniques," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2017, pp. 55–60.

[26] M. L. P. Gort, C. Feregrino-Uribe, and A. Cortesi, "HQR-scheme: A high quality and resilient virtual primary key generation approach for watermarking relational data," *Expert Syst. Appl.*, vol. 138, Dec. 2019, Art. no. 112770.

[27] M. L. P. Gort, C. Feregrino-Uribe, and A. Cortesi, "A double fragmentation approach for improving virtual primary key-based watermark synchronization," *IEEE Access*, vol. 8, pp. 61504–61516, 2020.

[28] L. Zhang, W. Gao, N. Jiang, L. Zhang, and Y. Zhang, "Relational databases watermarking for textual and numerical data," in *Proc. Int. Conf. Mech. Sci., Electric Eng. Comput. (MEC)*, Aug. 2011, pp. 1633–1636.

[29] K. Huang, M. Yue, P. Chen, Y. He, and X. Chen, "A cluster-based watermarking technique for relational database," in *Proc. 1st Int. Workshop Database Technol. Appl.*, Apr. 2009, pp. 107–110.

[30] C. Wang, J. Wang, M. Zhou, G. Chen, and D. Li, "ATBaM: An Arnold transform based method on watermarking relational data," in *Proc. Int. Conf. Multimedia Ubiquitous Eng.*, 2008, pp. 263–270.

[31] Z. Hu, Z. Cao, and J. Sun, "An image based algorithm for watermarking relational databases," in *Proc. Int. Conf. Measuring Technol. Mechatronics Autom.*, 2009, pp. 425–428.

[32] X. Zhou, M. Huang, and Z. Peng, "An additive-attack-proof watermarking mechanism for databases' copyrights protection using image," in *Proc. ACM Symp. Appl. Comput.*, 2007, pp. 254–258.

[33] M. L. Pérez Gort, C. Feregrino Uribe, and J. Nummenmaa, "A minimum distortion: High capacity watermarking technique for relational data," in *Proc. 5th ACM Workshop Inf. Hiding Multimedia Secur.*, Jun. 2017, pp. 111–121.

[34] H. M. Sardroudi and S. Ibrahim, "A new approach for relational database watermarking using image," in *Proc. 5th Int. Conf. Comput. Sci. Converg. Inf. Technol.*, Nov. 2010, pp. 606–610.

[35] A. Al-Haj and A. Odeh, "Robust and blind watermarking of relational database systems," *J. Comput. Sci.*, vol. 4, no. 12, pp. 1024–1029, 2008.

[36] S. Yige, L. Weidong, S. Jiaxing, and W. M. S. Angela, "DCT transform based relational database robust watermarking algorithm," in *Proc. 2nd Int. Symp. Data, Privacy, E-Commerce*, Sep. 2010, pp. 61–65.

[37] R. Sion, M. Atallah, and S. Prabhakar, "Rights protection for relational data," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 12, pp. 1509–1525, Dec. 2004.

[38] M. Shehab, E. Bertino, and A. Ghafoor, "Watermarking relational databases using optimization-based techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 116–129, Jan. 2008.

[39] M. Kamran and M. Farooq, "A formal usability constraints model for watermarking of outsourced datasets," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 6, pp. 1061–1072, Jun. 2013.

[40] M. Kamran, S. Suhail, and M. Farooq, "A robust, distortion minimizing technique for watermarking relational databases using once-for-all usability constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, pp. 2694–2707, Dec. 2013.

[41] M. Huang, J. Cao, Z. Peng, and Y. Fang, "A new watermark mechanism for relational data," in *Proc. 4th Int. Conf. Comput. Inf. Technol.*, Sep. 2004, pp. 946–950.

[42] B. V. Rao and M. V. Prasad, "Subset selection approach for watermarking relational databases," in *Proc. Int. Conf. Data Eng. Manage.* Berlin, Germany: Springer, 2010, pp. 181–188.

[43] H. Guo, Y. Li, A. Liu, and S. Jajodia, "A fragile watermarking scheme for detecting malicious modifications of database relations," *Inf. Sci.*, vol. 176, no. 10, pp. 1350–1378, 2006.

[44] V. Khanduja, O. P. Verma, and S. Chakraverty, "Watermarking relational databases using bacterial foraging algorithm," *Multimedia Tools Appl.*, vol. 74, no. 3, pp. 813–839, 2015.

[45] V. Pournaghshband, "A new watermarking approach for relational data," in *Proc. 46th Annu. Southeast Regional Conf.*, 2008, pp. 127–131.

[46] V. Prasannaku, "A robust tamperproof watermarking for data integrity in relational databases," *Res. J. Inf. Technol.*, vol. 1, no. 3, pp. 115–121, Mar. 2009.

[47] S. Liu, S. Wang, R. H. Deng, and W. Shao, "A block oriented fingerprinting scheme in relational database," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Berlin, Germany: Springer, 2004, pp. 455–466.

[48] Y. Li, V. Swarup, and S. Jajodia, "Fingerprinting relational databases: Schemes and specialties," *IEEE Trans. Dependable Secure Computing*, vol. 2, no. 1, pp. 34–45, Jan. 2005.

[49] F. Guo, J. Wang, and D. Li, "Fingerprinting relational databases," in *Proc. ACM Symp. Appl. Comput.*, 2006, pp. 487–492.

[50] J. Lafaye, D. Gross-Amblard, C. Constantin, and M. Guerrouani, "Watermill: An optimized fingerprinting system for databases under constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 4, pp. 532–546, Apr. 2008.

[51] E. A. Solami, M. Kamran, M. S. Alkatheiri, F. Rafiq, and A. S. Alghamdi, "Fingerprinting of relational databases for stopping the data theft," *Electronics*, vol. 9, no. 7, p. 1093, 2020.

[52] F. Guo, J. Wang, Z. Zhang, X. Ye, and D. Li, "An improved algorithm to watermark numeric relational data," in *Proc. Int. Workshop Inf. Secur. Appl.* Berlin, Germany: Springer, 2005, pp. 138–149.

[53] X. Cui, X. Qin, G. Sheng, and J. Zheng, "A robust algorithm for watermark numeric relational databases," in *Intelligent Control and Automation*. Berlin, Germany: Springer, 2006, pp. 810–815.

[54] T.-L. Hu, G. Chen, K. Chen, and J.-X. Dong, "GARWM: Towards a generalized and adaptive watermark scheme for relational data," in *Proc. Int. Conf. Web-Age Inf. Manage.* Berlin, Germany: Springer, 2005, pp. 380–391.

[55] X. Cui, X. Qin, and G. Sheng, "A weighted algorithm for watermarking relational databases," *Wuhan Univ. J. Natural Sci.*, vol. 12, no. 1, pp. 79–82, Jan. 2007.

[56] D. Gross-Amblard, "Query-preserving watermarking of relational databases and xml documents," *ACM Trans. Database Syst.*, vol. 36, no. 1, pp. 1–24, 2011.

[57] Y. Li, H. Guo, and S. Jajodia, "Tamper detection and localization for categorical data using fragile watermarks," in *Proc. 4th ACM Workshop Digit. Rights Manage.*, 2004, pp. 73–82.

[58] S. Bhattacharya and A. Cortesi, "A distortion free watermark framework for relational databases," in *Proc. ICSOFT*, 2009, pp. 229–234.

[59] I. Kamel, "A schema for protecting the integrity of databases," *Comput. Secur.*, vol. 28, no. 7, pp. 698–709, Oct. 2009.

[60] M. Li, W. Zhao, and J. Guo, "An asymmetric watermarking scheme for relational database," in *Proc. IEEE 3rd Int. Conf. Commun. Softw. Netw.*, May 2011, pp. 180–184.

[61] R. Arun, K. Praveen, D. C. Bose, and H. V. Nath, "A distortion free relational database watermarking using patch work method," in *Proc. Int. Conf. Inf. Syst. Design Intell. Appl.*, Visakhapatnam, India. Berlin, Germany: Springer, Jan. 2012, pp. 531–538.

[62] I. Kamel, M. AlaaEddin, W. Yaqub, and K. Kamel, "Distortion-free fragile watermark for relational databases," *Int. J. Big Data Intell.*, vol. 3, no. 3, pp. 190–201, 2016.

[63] Y. Li and R. H. Deng, "Publicly verifiable ownership protection for relational databases," in *Proc. ACM Symp. Inf., Comput. Commun. Secur.*, 2006, pp. 78–89.

[64] S. Bhattacharya and A. Cortesi, "A generic distortion free watermarking technique for relational databases," in *Proc. Int. Conf. Inf. Syst. Secur.* Berlin, Germany: Springer, 2009, pp. 252–264.

[65] R. Halder and A. Cortesi, "A persistent public watermarking of relational databases," in *Proc. Int. Conf. Inf. Syst. Secur.* Berlin, Germany: Springer, 2010, pp. 216–230.

[66] R. Halder and A. Cortesi, "Persistent watermarking of relational databases," in *Proc. IEEE Int. Conf. Adv. Commun., Netw., Comput. (CNC)*, 2010, pp. 46–52.

[67] S. Bhattacharya and A. Cortesi, "Distortion-free authentication watermarking," in *Proc. Int. Conf. Softw. Data Technol.* Berlin, Germany: Springer, 2010, pp. 205–219.

[68] A. Hamadou, X. Sun, and L. Gao, "A fragile zero-watermarking technique for authentication of relational databases," *Int. J. Digit. Content Technol. Appl.*, vol. 5, no. 5, pp. 189–200, May 2011.

[69] L. Camara, J. Li, R. Li, and W. Xie, "Distortion-free watermarking approach for relational database integrity checking," *Math. Problems Eng.*, vol. 2014, Dec. 2014, Art. no. 697165.

[70] A. Khan and S. A. Husain, "A fragile zero watermarking scheme to detect and characterize malicious modifications in database relations," *Sci. World J.*, vol. 2013, pp. 1–16, Oct. 2013.

[71] S. M. Darwish, "Distortion free database watermarking system based on intelligent mechanism for content integrity and ownership control," *J. Comput.*, vol. 4, pp. 1053–1066, Dec. 2018.

[72] S. Rani, D. K. Koshley, and R. Halder, "Adapting mapreduce for efficient watermarking of large relational dataset," in *Proc. Trustcom/BigDataSE/ICESS*, 2017, pp. 729–736.

[73] S. Siledar and S. Tamane, "A distortion-free watermarking approach for verifying integrity of relational databases," in *Proc. Int. Conf. Smart Innov. Design, Environ., Manage., Planning Comput. (ICSIDEMPC)*, Oct. 2020, pp. 192–195.

[74] I. K. W. Yaqub and Z. Aung, "Distortion-free watermarking scheme for compressed data in columnar database," in *Proc. ICETE*, Porto, Portugal, 2018, pp. 343–353.

[75] F. Naz, A. Khan, M. Ahmed, M. I. Khan, S. Din, A. Ahmad, and G. Jeon, "Watermarking as a service (WAAS) with anonymity," *Multimedia Tools Appl.*, vol. 79 no. 23, pp. 16051–16075, 2020.

[76] S. A. Shah, I. A. Khan, S. Z. H. Kazmi, and F. H. B. M. Nasaruddin, "Semi-fragile watermarking scheme for relational database tamper detection," *Malaysian J. Comput. Sci.*, vol. 34, no. 1, pp. 1–12, Oct. 2021.

[77] S. B. Siledar and S. Tamane, "Quadratic difference expansion based reversible watermarking for relational database," *J. Integr. Sci. Technol.*, vol. 9, no. 2, pp. 107–112, 2021.

[78] R. Hou and H. Xian, "A graded reversible watermarking scheme for relational data," *Mobile Netw. Appl.*, vol. 26, no. 4, pp. 1552–1563, 2021.

[79] C.-C. Lin, T.-S. Nguyen, and C.-C. Chang, "LRW-CRDB: Lossless robust watermarking scheme for categorical relational databases," *Symmetry*, vol. 13, no. 11, p. 2191, 2021.

[80] X. Shen, Y. Zhang, T. Wang, and Y. Sun, "Relational database watermarking for data tracing," in *Proc. Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discovery (CyberC)*, Oct. 2020, pp. 224–231.

[81] Y. Li, J. Wang, and X. Luo, "A reversible database watermarking method non-redundancy shifting-based histogram gaps," *Int. J. Distrib. Sensor Netw.*, vol. 16, no. 5, May 2020, Art. no. 155014772092176.

[82] J. Lian, "A new reversible database watermarking approach with ant colony optimization algorithm," *J. Phys., Conf. Ser.*, vol. 1616, no. 1, Aug. 2020, Art. no. 012040.

[83] Y. Li, J. Wang, and H. Jia, "A robust and reversible watermarking algorithm for a relational database based on continuous columns in histogram," *Mathematics*, vol. 8, no. 11, p. 1994, 2020.

[84] A. Hamadou, L. Camara, A. A. I. Hassane, and H. Naroua, "Reversible fragile watermarking scheme for relational database based on prediction-error expansion," *Math. Problems Eng.*, vol. 2020, May 2020, Art. no. 1740205.

[85] C. Ge, J. Sun, Y. Sun, Y. Di, Y. Zhu, L. Xie, and Y. Zhang, "Reversible database watermarking based on random forest and genetic algorithm," in *Proc. Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discovery (CyberC)*, Oct. 2020, pp. 239–247.

[86] H. Tufail, K. Zafar, and A. R. Baig, "Relational database security using digital watermarking and evolutionary techniques," *Comput. Intell.*, vol. 35, no. 4, pp. 693–716, 2019.

[87] H. Chai, S. Yang, Z. L. Jiang, and X. Wang, "A robust and reversible watermarking technique for relational dataset based on clustering," in *Proc. 18th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Aug. 2019, pp. 411–418.

[88] H. Chai, S. Yang, Z. L. Jiang, X. Wang, Y. Chen, and H. Luo, "A new robust and reversible watermarking technique based on erasure code," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.* Cham, Switzerland: Springer, 2019, pp. 153–168.

[89] G.-Y. Wu and C.-W. Lee, "Framework for cloud database protection by using reversible data hiding methods," in *Proc. IEEE Int. Conf. Consum. Electron.-Taiwan (ICCE-TW)*, May 2019, pp. 1–2.

[90] Y. Li, J. Wang, S. Ge, X. Luo, and B. Wang, "A reversible database watermarking method with low distortion," *Math. Biosci. Eng.*, vol. 16, no. 5, pp. 4053–4068, 2019.

[91] D. Hu, D. Zhao, and S. Zheng, "A new robust approach for reversible database watermarking with distortion control," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1024–1037, Jun. 2018.

[92] M. B. Imamoglu, M. Ulutas, and G. Ulutas, "A new reversible database watermarking approach with firefly optimization algorithm," *Math. Problems Eng.*, vol. 2017, Mar. 2017, Art. no. 1387375.

[93] C.-C. Chang, T.-S. Nguyen, and C.-C. Lin, "A virtual primary key for reversible watermarking textual relational databases," in *Proc. Intell. Syst. Appl.*, 2015, pp. 756–769.

[94] C.-C. Chang, T.-S. Nguyen, and C.-C. Lin, "A blind robust reversible watermark scheme for textual relational databases with virtual primary key," in *Proc. Int. Workshop Digit. Watermarking*. Cham, Switzerland: Springer, 2014, pp. 75–89.

[95] S. Iftikhar, M. Kamran, and Z. Anwar, "RRW—A robust and reversible watermarking technique for relational data," *IEEE Trans. Knowl. data Eng.*, vol. 27, no. 4, pp. 1132–1145, Aug. 2014.

[96] C.-C. Chang, T.-S. Nguyen, and C.-C. Lin, "A blind reversible robust watermarking scheme for relational databases," *Sci. World J.*, vol. 2013, pp. 1–12, Oct. 2013.

[97] K. Jawad and A. Khan, "Genetic algorithm and difference expansion based reversible watermarking for relational databases," *J. Syst. Softw.*, vol. 86, no. 11, pp. 2742–2753, Nov. 2013.

[98] M. E. Farfoura, S.-J. Horng, and X. Wang, "A novel blind reversible method for watermarking relational databases," *J. Chin. Inst. Eng.*, vol. 36, no. 1, pp. 87–97, Jan. 2013.

[99] M. E. Farfoura, S.-J. Horng, J.-L. Lai, R.-S. Run, R.-J. Chen, and M. K. Khan, "A blind reversible method for watermarking relational databases based on a time-stamping protocol," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 3185–3196, 2012.

[100] J. Franco Contreras, G. Coatrieux, E. Chazard, F. Cuppens, N. Cuppens-Boulahia, and C. Roux, "Robust lossless watermarking based on circular interpretation of bijective transformations for the protection of medical databases," in *Proc. Annu. Int. Conf. Eng. Med. Biol. Soc.*, Aug. 2012, pp. 5875–5878.

[101] G. Gupta and J. Pieprzyk, "Reversible and blind database watermarking using difference expansion," *Int. J. Digit. Crime Forensics*, vol. 1, no. 2, pp. 42–54, Apr. 2009.

[102] G. Gupta and J. Pieprzyk, "Reversible and semi-blind relational database watermarking," in *Proc. SIGMAP*, 2007, pp. 283–290.

[103] Y. Zhang, B. Yang, and X.-M. Niu, "Reversible watermarking for relational database authentication," *J. Comput.*, vol. 17, no. 2, pp. 59–66, 2006.

[104] G. Gupta and J. Pieprzyk, "Reversible and blind database watermarking using difference expansion," in *Proc. 1st Int. ICST Conf. Forensic Appl. Techn. Telecommun., Inf. Multimedia*, 2008, pp. 1–6.

[105] K. Unnikrishnan and K. Pramod, "Robust optimal position detection scheme for relational database watermarking through HOLPSOFA algorithm," *J. Inf. Secur. Appl.*, vol. 35, pp. 1–12, Aug. 2017.

**SAPANA RANI** received the B.Tech. degree in information technology from MIT, Muzaffarpur, India, in 2012. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, IIT Patna. Her research interests include database watermarking, information security, and big data.

**RAJU HALDER** (Member, IEEE) received the Ph.D. degree from Università Ca´ Foscari, Italy, in 2012. He is currently an Associate Professor with the Department of Computer Science and Engineering, IIT Patna. Before joining IIT Patna, he worked as a Postdoctoral Researcher with Macquarie University, Australia. He worked with the Robotics Team at HASLab (University of Minho), Portugal, in 2016. Prior to his Ph.D. degree, he had also worked as an Associate System Engineer at IBM India Private Ltd., from 2007 to 2008. His research interests include formal methods, blockchain technology, program analysis and verification, and data privacy and security.