Information Sciences 743 (2026) 123359

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A qualitative and quantitative comparative study of VPK schemes for relational data watermarking

Maikel Lázaro Pérez Gort*, Agostino Cortesi

Ca' Foscari University of Venice, Via Torino 155, Mestre (VE), Venice, 30170, Italy

HIGHLIGHTS

- First comparative study of VPK schemes for relational data watermarking.
- PKs and VPKs do not have a fixed role in watermark synchronization.
- Duplicate VPKs do not necessarily disrupt watermark synchronization.
- Attribute deletion is not as damaging to watermark detection as it seems.

ARTICLE INFO

Keywords:
Digital watermarking
Virtual primary key
Relational data
Primary key
Duplicate problem
Deletion problem

ABSTRACT

In relational data watermarking, several virtual primary key (VPK) schemes have been proposed to address threats such as primary key (PK) deletion. These schemes face significant challenges and are often evaluated under heterogeneous assumptions and criteria. This paper provides a systematic qualitative and quantitative evaluation of existing VPK schemes, as well as alternative solutions that avoid relying on PKs for watermark synchronization. We standardize notation and terminology, organize the design space according to VPK generation and management strategies, and compare major scheme families using consistent criteria covering duplication behavior, deletion resilience, attribute involvement, and computational overhead. The analysis highlights progress achieved, remaining limitations, and open research challenges. Finally, the paper distills practical selection guidelines for PK-unavailable data-sharing and processing pipelines and motivates future work on realistic threat models and reproducible benchmarks.

1. Introduction

Digital watermarking is an information-hiding technique designed to protect digital assets without constraining their copying, deployment, management, or distribution. It has proven effective for ownership protection, data tampering detection, and copy tracing, among other applications. Digital watermarking typically works by embedding a copyright signature (a.k.a. the watermark) into the protected digital asset, and later extracting it on demand. The combination of watermark embedding and extraction is referred to as *watermark synchronization* [1].

The successful application of digital watermarking extends to content stored in multimedia data, documents, text, relational databases, compiled code, and many other forms of digital content. In domains such as medical image watermarking, hybrid approaches that combine deep-learning-based feature extraction with transform-domain embedding have emerged in recent years (e.g., Nawaz et al., 2024 [2]), reflecting a broader shift toward machine-learning-driven watermark generation. Watermarking is

* Corresponding author.
Email addresses: maikel.perezgort@unive.it (M.L. Pérez Gort), cortesi@unive.it (A. Cortesi).

https://doi.org/10.1016/j.ins.2026.123359
Received 24 November 2025; Received in revised form 9 March 2026; Accepted 9 March 2026
Available online 10 March 2026

particularly useful in healthcare, legal, and administrative data management, where provenance, ownership, or tamper evidence must be established while preserving data usability without relying solely on cryptographic protection.

Recent robust image watermarking models further illustrate this trend by combining multi-scale feature extraction with residual or attention mechanisms to improve imperceptibility and robustness [3]. Watermarking techniques have also expanded to generative AI, where latent-space watermarking frameworks have been proposed for diffusion-model-generated images to support authenticity and copyright protection [4].

For relational databases, techniques almost always use the primary key (PK) of the relation being watermarked to determine where the mark is embedded and which elements are involved in its generation. Given the uniqueness of the PK, it contributes to high detectability and reliable watermark synchronization. However, relying on the PK has its risks. If the protected data are distributed separately from the database, and there is no future need to reintegrate them, PKs can be deleted or replaced with other unique values. By doing so, an adversary can compromise watermark synchronization without affecting the quality of the data. Although the marks remain stored in the relation, detecting the watermark becomes very difficult.

This vulnerability is particularly relevant in data-sharing pipelines where the original PK is routinely unavailable, unstable, or intentionally removed. Representative scenarios include: (i) licensing or sharing relational datasets with external partners, in which identifier columns are stripped for privacy or contractual reasons; (ii) outsourcing analytics through Extract, Transform, and Load (ETL) workflows that deduplicate, merge, and reload relations using regenerated surrogate keys; and (iii) publishing de-identified microdata for research, where unique identifiers are omitted by design. In these settings, PK-based synchronization may fail even when the relation's informational content remains largely intact, thereby motivating PK-free synchronization mechanisms.

To address this issue, researchers have proposed schemes that create values known as virtual primary keys (VPKs), which can be used for watermark synchronization without relying on PKs. This paper presents a quantitative and qualitative evaluation of these schemes as the core of a comparative study that encompasses all aspects related to VPK generation and management. Since the published literature sometimes uses the same symbols to denote different concepts, we standardize notation in Tables 1 and 2.

Table 1
Notations of functions, operators, variables, and other elements (first part).

Symb.	Meaning	Symb.	Meaning
$\mathcal{I}(\cdot)$	Watermark insertion function.	$\mathcal{E}(\cdot)$	Watermark extraction function.
R	Original unwatermarked relation.	R'	Watermarked version of R.
$\hat{R}$	Updated or attacked version of R'.	K_{P}	Primary key of a relation.
η	Number of tuples of a relation.	ν	Number of attributes of a relation.
r_j	j-th tuple of a relation, $j \in [0, \eta)$.	A_i	i-th attribute of a relation, $i \in [0, \nu)$.
W	Original watermark to embed in R.	W'	Watermark extracted from $\hat{R}$.
K_{S}	Data owner's secret key.	w_k	k-th mark of a watermark, $k \in [0, \lvert W \rvert)$.
S	VPK set used to embed W.	S'	VPK set to extract the watermark.
K_{VP}	Virtual primary key of a tuple.	γ	Tuple fraction, $\gamma \in [1, \eta]$.
G	Set of all duplicate values in S.	Φ	Set of null values in S.
D	Set of duplicate values in S (contains one value per duplicate group).	G_r	r-th group of duplicate values in S, $r \in [0, \lvert D \rvert)$.
E	Set of exclusive values in S.	e_x	x-th exclusive value in E, $x \in [0, \lvert E \rvert)$.
β	*Most significant bits* (*msb*) of $[r_j.A_i]_2$.	ξ	*Less significant bits* (*lsb*) of $[r_j.A_i]_2$.
β_f	Fraction of most significant bits.	$\hat{\beta}$	Temporary most significant bit.
ℓ'	Maximum number of attributes per tuple involved in VPK generation.	p	Minimum number of attributes between attributes selected for VPK generation.
b_{jih}	h-th bit of $[r_j.A_i]_2$.	s_j	j-th key of a VPK set.
A_{vk}	Attribute virtual key.	A_{vv}	Attribute virtual value.
A_{Mv}	Maximum value of A_{vv} per tuple.	A_{Av}	Average value of A_{vv} per tuple.
$\mathbf{R}$	Set of binary strings representing the selection ratio of each attribute.	A_{C}	Value created by cross-splicing the bits from A_{vv} with A_{Mv}.
$A_{\mathcal{V}}$	Set containing A_{C} values considered for VPK generation.	v_k	k-th bit of K_{VP}, $k \in [0, h)$
`mod`	Modulo operator.	$\equiv$	Equivalence relationship.
$\mathcal{H}(\cdot)$	Common one-way hash function.	$\circ$	Concatenation operator.
$\lvert\cdot\rvert$	Cardinality or length of a set or string.	$\mathcal{F}(\cdot)$	Function that generates a tuple identifier (not necessarily a VPK).
$[\cdot]_2$	Binary representation of a value.	$\mathcal{V}(\cdot)$	Function to compute a VPK.
$[\cdot]_{10}$	Decimal notation of a numerical value.	`VPK`$(\cdot)$	Function to extract the β bits of a given binary string.
$\lfloor\cdot\rceil$	Rounding function.	$\oplus$	`XOR` operator
$\lfloor\cdot\rfloor$	Floor function.	$\lceil\cdot\rceil$	Ceiling function.
$\mathcal{Z}(\cdot)$	Function that returns the number of zeros in the binary value.	`MLU`$(\cdot)$	Function that computes the machine learning utility of a dataset.
`sin`$(\cdot)$	Trigonometric function sine.	$\lVert\cdot\rVert_2$	The Euclidean norm (L_2−norm) function of a vector.
$\phi_x(\cdot)$	Function for mapping a tuple to a x-coordinate.	$\phi_y(\cdot)$	Function for mapping a tuple to a y-coordinate.
$\varphi(\cdot)$	Operation that maps a relation R into a discrete-time signal T.	$\hat{P}(\cdot)$	Spectrum power function that checks the presence of a sinusoidal signal at a given frequency in a discrete-time signal.

Table 2
Notations of functions, operators, variables, and other elements (second part).

Sym.	Meaning	Sym.	Meaning
M	Median set in the VT-Scheme	κ	Element involved in VPK generation using the HSR-Scheme.
$\vec{Z}$	Hidden layer vector in the latent space of a variational autoencoder model.	h	Dimension of the hidden layer vector. Length of $[K_{\text{VP}}]_2$
z_k	k-th item of $\vec{Z}$, $k \in [0, h)$	m_k	Median of the dimension z_k
A	Sets of attributes considered for VPK generation from R.	$\boldsymbol{\Phi}_A$	Set of attribute values concatenated with their corresponding labels.
$\tilde{P}$	Record table used by the HSR-Scheme to store auxiliary information for watermark detection.	Z	Set of points containing all mapped tuples from R.
A_i^1	Last word of a multi-word textual attribute used for mark embedding.	A_i^{-1}	Content of a multi-word textual attribute excluding the last word, used for VPK generation.
M_r	First fragment of a watermark W.	m_{rs}	Second fragment of a watermark obtained from M_r (i.e., a mark).
L	Value proportional to the sum of the cardinalities of all duplicate groups located before G_r.	seed_j	Seed generated for the tuple r_j used by the DF-Technique.
$\vec{\mathbf{e}}_x$	Orthogonal vector for mapping a tuple a x-coordinate.	$\vec{\mathbf{e}}_y$	Orthogonal vector for mapping a tuple a y-coordinate.
x_j	x-coordinate on which the tuple r_j is projected.	y_j	y-coordinate on which the tuple r_j is projected.
B_{h}	Set of points having the same x-coordinate.	h	Value of the x-coordinates of the point in B_{h}
$\bar{x}$	Value summarizing the x-coordinates of the points in h.	$\bar{y}$	Mean of the y-coordinates of the points in B_{h}
b	Width of a bin used to project a tuple.	τ	Divisor used to map a bin index into an x-coordinate.
T	Discrete-time signal.	λ	Amplitude of a sinusoidal signal embedded into T.
θ	Frequency of a sinusoidal signal embedded into T.	r'_j	Tuple containing a sinusoidal signal of θ frequency and λ amplitude.
T'	Discrete-time signal obtained from R'	$\hat{T}$	Discrete-time signal obtained from $\hat{R}$
δ	Duplicate index.	$\omega_{\min}$	Minimum number of times the same mark is embedded.
$\omega_{\max}$	Maximum number of times the same mark is embedded.	ρ	Exclusiveness index.
u_r	Payload of the duplicate VPK group G_r	a_r	Number of VPKs within the group G_r affected by the attribute deletion.
φ_c	Degree of calibration of the exclusiveness index.	m	Number of VPKs matching in index and values between S and S'
ρ_O	Exclusiveness index calculated without checking index-value matching.	ρ_C	Calibrated exclusiveness index.
I_{emb}	Image generated from the embedded watermark.	I_{ext}	Image generated from the extracted watermark.
h^I, w^I	Height and width of the image used as watermark source.	∂	Qualitative difference between two VPK sets.
CF	Correction Factor	SSIM	Structural Similarity Index

The main motivation behind this work is that despite more than two decades of work on VPK-based synchronization, existing schemes are often presented under heterogeneous assumptions and evaluated under non-uniform experimental setups, with notation and terminology that differ substantially across papers. This fragmentation makes it difficult to (i) compare schemes fairly, (ii) understand which design choices most strongly affect duplication, deletion resilience, and attribute involvement, and (iii) select a suitable VPK mechanism for practical deployments in PK-unavailable settings.

This survey focuses on VPK-based and PK-free synchronization mechanisms, rather than on relational database watermarking techniques in general. As a result, the number of works reviewed here is smaller than in broader watermarking surveys. Papers were included if they satisfied at least one of the following criteria: (i) explicit introduction of a virtual primary key or PK-free synchronization strategy; (ii) proposal of a distinct technical principle for VPK generation or management; (iii) representation of a key evolutionary step or use as a baseline in subsequent VPK-related studies; or (iv) introduction of recent adaptive or learning-based approaches to VPK construction. Using these criteria, we ensured coverage of all major stages in the evolution of VPK schemes, from early bit-extraction methods to recent data-adaptive and learning-driven designs, while excluding PK-dependent watermarking approaches that do not address synchronization under PK removal. The selected papers are analytically evaluated in Section 5 by clustering them according to the approaches they adopt.

The main contributions of this work are as follows:

i. a compilation and systematization of the complete body of VPK-based synchronization schemes proposed to date, covering more than two decades of research;
ii. the introduction of a standardized notation and terminology framework (Tables 1 and 2) that resolves symbol reuse and enables consistent cross-scheme discussion;
iii. a structured analysis of VPK-generation principles and management choices, including how schemes address (or exacerbate) duplication, deletion resilience, and attribute involvement;

iv. a rigorous qualitative and quantitative evaluation methodology that compares schemes under consistent criteria and performance metrics; and
v. practical guidance on the trade-offs among robustness, computational cost, and sensitivity to data modification, together with consolidated challenges and opportunities for future research.

This survey advances the state of the art by providing the first unified and systematic comparison of all VPK schemes proposed to date, covering more than two decades of research. It introduces a standardized notation framework, a structured analysis of VPK-generation principles, and a rigorous evaluation methodology that jointly examines qualitative design aspects and quantitative performance metrics. The study also clarifies long-standing issues related to duplication, deletion resilience, and attribute involvement, integrating both classical schemes and more recent adaptive or learning-based approaches within a consistent analytical perspective.

The experimental evaluation highlights clear distinctions among the major families of VPK schemes. Early bit-extraction methods exhibit high duplication and fragile behavior under attribute deletion. Adaptive schemes such as the Ext- and HQR-Schemes reduce duplication and improve robustness, while more recent proposals, including the FRBBM- and VT-Schemes, provide stronger deletion resilience and more balanced VPK distributions. The hybrid HSR-Scheme produces the highest-quality VPK sets but requires auxiliary structures and higher computational effort. These findings reveal a well-defined evolution from simple PK-free mappings toward more robust, adaptive, and data-aware synchronization strategies.

The results of this survey offer practical guidance for the selection and design of VPK mechanisms in real-world watermarking applications. By characterizing the trade-offs between robustness, computational cost, and sensitivity to data modification, the analysis enables practitioners to choose the most appropriate VPK strategy for environments such as secure data sharing, tamper detection, and copyright protection in relational databases. The study also identifies scenarios in which particular schemes are more effective, providing insights that can inform the deployment, tuning, and further development of VPK-based synchronization solutions.

The rest of this paper is organized as follows. Section 2 presents the preliminaries of VPK schemes, including the core definitions and notation used throughout the survey. Section 3 addresses the main challenges when performing VPK-based watermark synchronization, with emphasis on the duplication and deletion problems and the design factors that influence VPK quality. Section 4 reviews the schemes proposed to date and other solutions designed to enable PK-free watermark synchronization, highlighting their core principles and evolution over time. Section 5 presents a qualitative and quantitative comparison of the works discussed in the previous section, using consistent criteria to summarize the resulting trade-offs and practical implications. Section 6 explores potential opportunities for further research, drawing from the comparative findings, and Section 7 concludes.

2. Preliminaries

There are two kinds of watermarking approaches: fragile and robust. Fragile approaches are mainly designed to detect data tampering. If the watermark is not detected, tampering is usually assumed, indicating the compromised quality of the digital asset. Fragile watermarking for relational databases has been investigated since the early 2000s, with Guo et al. [5] proposing one of the first schemes capable of detecting and localizing tuple-level modifications in a relation through fragile marks embedded at group granularity.

On the other hand, robust approaches are designed to provide copyright protection by reporting ownership if the watermark is detected in a copy of the data subject to ownership claims. Robust approaches are also used for fingerprinting, where different copies of the same data are marked with a distinct watermark to detect the origin of the copies, thereby tracing buyers who might have violated contract acquisition terms. Given the high stakes, the watermark is meant to resist any intentional or unintentional operation that might compromise its detection. In addition, reversible watermarking techniques have been proposed to enable perfect recovery of the original data after watermark removal, which is particularly important in sensitive domains such as medical or scientific data [6].

2.1. Structure of a database relation

According to the relational model, a relational database comprises relations linked to one another, each representing an entity in the modeled reality. Database relations are modeled as tables and have a primary key, which is used to identify each instance stored in them and link the relations to the rest of the database. In the table, the columns represent the attributes (including those comprising the PK) and the rows represent the tuples (or records) storing data instances [7].

Fig. 1 depicts the table format of a generic relation R and introduces the formal notation to refer to any relation without relying on the attribute names. We denote the primary key as K_P and the attributes as $A_i : i \in [0, \nu)$, where ν is the number of attributes available for marking. The tuples of the relation are identified by $r_j : j \in [0, \eta)$, where η is the number of tuples stored in the relation. The entire relation is denoted as $R(K_P, A_0, \ldots, A_{(\nu-1)})$, the value stored in the *i-th* attribute of the *j-th* tuple is denoted as $r_j.A_i$, and the primary key of each tuple is given by $r_j.K_P$.

Although the PK is often composed of a single attribute, it can also be composed of multiple attributes combined, if this results in unique values, allowing for the identification of each tuple. Nevertheless, for the sake of simplicity, in this work, we always refer to a single relation with PK composed of a single attribute. As long as R is managed inside the database, the PK is typically stable because it preserves referential integrity. When relations are released outside the database context (e.g., for sharing or processing), identifiers may be removed or regenerated, as discussed in Section 1, which motivates PK-free synchronization mechanisms.

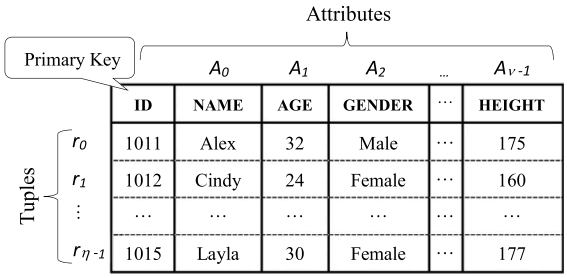


Fig. 1. Illustrative structure of a generic relation R: tuples r_j with primary key K_P and attributes A_i, as used throughout the paper to describe PK-based and VPK-based synchronization [8].

2.2. Architecture of relational data watermarking

Relational data watermarking implements the same processes as any other watermarking technique: watermark insertion and extraction. Both processes utilize a cryptographic secret key K_S, which is known only to the data owner. First, the insertion process generates the watermark W and then embeds it in R, producing a watermarked version of the relation denoted as R'. We formally represent watermark insertion as $R' = \mathcal{I}(K_S, W, R)$, where $\mathcal{I}(\cdot)$ constitutes the insertion function. The watermark W is composed of a set of bits identified as the marks, formally $w_k : k \in [0, |W|)$, where $|\cdot|$ denotes the cardinality operator. The watermark can be generated from an external and independent source, which is the case for *meaningful watermarking*. There are also watermarks consisting of meaningless bit-patterns, being classified as *meaningless watermarking* [9][1].

After watermark insertion, R' can be deployed on a public server and be subjected to data access and *benign updates* comprising tuple insertion, tuple deletion, attribute updates, and any other processing operations formally defined in the business model. Moreover, adversaries can target the relation seeking to compromise the watermark detection through attacks or any other kind of *malicious operations* [10].

The aftermath of *benign updates* and *malicious operations* transforms R' into $\hat{R}$. If robustness is guaranteed, the watermark should still be present in $\hat{R}$. Watermark extraction is performed whenever it is necessary to verify false ownership claims or any other unauthorized activity on the protected data. We formally denote watermark extraction as $W' = \mathcal{E}(K_S, \hat{R})$, where $\mathcal{E}(\cdot)$ is the extraction function and W' is the extracted watermark.

After watermark extraction, if $W \equiv W'$, the final assessment of the rightful data owner can be given. We use $\equiv$ to refer to the equivalence between the two watermarks, considering that an exact match is neither required nor expected. Instead, a certain degree of similarity must be present to assess the watermark detection. This assessment is made assuming that no secondary watermark embedding was performed, given that *additive attacks* [10] are out of the scope of this paper.

2.3. Threat model and assumptions

We consider a standard robust relational watermarking setting in which an adversary obtains the released watermarked relation R' and may know the watermarking and VPK algorithms, but does not know the secret key K_S. The adversary's goal is to prevent reliable watermark detection (false negatives) while preserving data utility, i.e., without arbitrarily destroying attribute semantics or violating basic domain constraints.

We model two classes of transformations that may be applied to R' before detection, producing $\hat{R}$. *Benign updates* capture routine data-management operations (e.g., ETL processing, de-identification, tuple insertion/deletion, and attribute updates). *Malicious operations* aim specifically at breaking synchronization, including PK deletion, replacement, or regeneration, tuple and attribute reordering, attribute deletion, and content modifications that affect the values used in VPK construction. These are the scenarios emphasized in Section 5 because they determine whether the detector can reproduce an equivalent synchronization set ($S \equiv S'$) and recover W'.

We assume the attacker does not compromise K_S and does not perform secondary watermark embedding (additive attacks), which is outside the scope of this paper. When a scheme relies on auxiliary protected structures (e.g., side tables), we assume they remain available to the legitimate detector but are not accessible to the adversary.

2.4. Watermark synchronization and tuple identification

Most robust relational watermarking schemes follow a common synchronization principle: the data owner deterministically derives, for each tuple r_j, an identifier that drives (i) whether the tuple is selected for embedding/extraction, and (ii) which attributes within the tuple are used. In PK-dependent schemes, this identifier is computed from the primary key value $r_j.K_P$ combined with K_S through a one-way hash, yielding $\mathcal{F}(r_j.K_P)$. In VPK-based schemes, the same role is played by a content-derived substitute $\mathcal{V}(r_j)$, from which the identifier is computed as $\mathcal{F}(\mathcal{V}(r_j))$ (see Section 4.2).

Given the identifier, tuple selection is typically performed through a modulo test, e.g., $\mathcal{F}(\cdot) \bmod \gamma = 0$, which selects an expected fraction $1/\gamma$ of tuples. For each selected tuple, additional modulo operations are used to select the carrier attribute and the bit position

[1] Some approaches generating W from fragments of R may be considered meaningful as well, as the watermark is not a meaningless sequence of bits.

(or other embedding location) in a deterministic way. During extraction, the same computations are re-run so that the detector visits the same tuple subset and carriers, enabling the reconstruction of W' from $\hat{R}$.

This generic pipeline also explains the two central challenges of PK-free synchronization. First, when identifiers are derived from tuple content, different tuples can map to the same VPK value, which yields duplicates in the VPK set S (the *duplicate problem*) and reduces synchronization entropy. Second, if the attributes involved in $\mathcal{V}(r_j)$ are deleted or modified, the recomputed identifiers may change between embedding and extraction, leading to inconsistent tuple identification and misalignment (the *deletion problem*). The remainder of this paper formalizes these issues and evaluates how different VPK schemes trade off duplication, deletion resilience, and practical constraints.

Tables 1 and 2 introduce the symbols used in this paper, along with the description of their roles. We explicitly adopt this notation, considering that the published literature presenting the works addressed in this paper may occasionally use the same symbols to denote different concepts.

3. Challenges

Building on the synchronization pipeline introduced in Section 2.4 and the motivation discussed in Section 1, this section summarizes the main technical challenges that govern PK-free watermark synchronization. We focus on limitations imposed by relation content and ordering, and on the two central PK-free issues, duplication and deletion, that affect the quality and stability of the VPK set.

The main goal of VPK schemes is to generate virtual primary keys to replace K_P in watermark synchronization. However, VPK-based synchronization introduces additional challenges. This section addresses the primary challenges and issues encountered when implementing VPK-based watermark synchronization, ranging from the design of VPK schemes to their application.

3.1. Limited content in R

The very structure of R poses a challenge for VPK generation. Database relations store limited content, constrained by the number of tuples and attributes. Attributes in a relation are usually few in number, and their values are restricted to their domains. For example, in a relation storing student data, an attribute for age is expected to contain integers within a specific interval, rather than arbitrary values that could describe a person's age in general. Tuples, on the other hand, while typically more numerous than attributes, represent occurrences corresponding to each entity reflected in the relation. Consequently, their number cannot grow without restrictions and must remain consistent with the business rules.

The issue regarding the limited content for generating VPKs worsens if we consider that only a fraction of the values stored in R can be utilized. Any value that might change due to *benign updates*, *malicious operations*, or the very embedding of the watermark should be ignored by VPK schemes. Otherwise, the VPK set generated to embed the watermark (denoted as S) can differ from the one computed to detect it (denoted as S'). This is risky because watermark synchronization requires $S \equiv S'$. The two sets may be slightly different, but they must remain equivalent to a degree that guarantees watermark synchronization.

3.2. Lack of fixed positions in R

When analyzing the values involved in VPK generation and watermark embedding, sequential selection is not recommended, considering that attributes and tuples in the relation do not present a fixed order. Their visual location in R, when using the table structure as a reference, constitutes a representation of the relation's design, enabling us to communicate effectively while designing, programming, and managing the database. Nevertheless, attributes and tuples can be reordered without affecting the results of the queries performed on the database. As a result, if sequential watermark embedding is used, adversaries can reorder the values in R, compromising watermark synchronization when the same sequential order is used for watermark detection. This is defined in the related literature as the *subset-reverse order attack* [11].

Researchers have proposed *pseudo-random* selection to avoid relying on sequential processes for watermark synchronization. However, *pseudo-random* selection can drive watermarking techniques to select the same attributes for different tuples multiple times while ignoring others, compromising the scattering of the marks in R'.

This problem also affects the elements considered for mark generation when using external sources in meaningful watermarking approaches. For example, when utilizing a binary image as a source, such as in [11], some pixels may be selected multiple times to generate the marks while others may be ignored, resulting in partial use of the source, and producing a low-quality watermark that may not even be recognizable after the embedding is concluded.

Low-quality watermark embedding is an issue that also occurs for PK-based techniques using *pseudo-random* selection. Still, watermark synchronization is expected to degrade further when it is not possible to rely on the uniqueness of the values stored in S.

3.3. The duplicate problem

When tuple identifiers are derived from content, distinct tuples can map to the same VPK value, producing duplicates in the synchronization set S (see Section 2.4). This *duplicate problem* [12] reduces the effective identifier space and can cause repeated selection of the same watermark positions while other positions remain underused, ultimately degrading synchronization quality.

Usually, embedding the same marks multiple times is considered a strength of a watermarking technique that helps withstand attacks based on updating or deleting data from R. The damage caused by those attacks is restored when a majority vote is performed after watermark extraction, thereby resolving inconsistencies that may result from extracting different values corresponding to the

same mark. Nevertheless, embedding the same marks multiple times at the expense of excluding others from the process compromises watermark synchronization.

3.4. The deletion problem

PK-free synchronization requires that identifiers computed at embedding time can be recomputed at detection time. If attributes involved in $\mathcal{V}(r_j)$ are deleted or modified, the recomputed set S' may diverge from S, leading to inconsistent tuple identification and misalignment (see Section 2.4). This divergence is the *deletion problem* [12].

Performing watermark synchronization with $S \not\equiv S'$ results in the addition of noise to W' as an outcome of computing incorrect values for some marks. Thus, the chances of the extracted watermark not matching the embedded one (i.e., $W \not\equiv W'$) drastically increase.

4. Schemes evolution

Watermarking techniques that use K_P to synchronize W are referred to as PK-dependent mechanisms. Representative examples include approaches based on the AHK algorithm, where a keyed hash of $r_j.K_P$ is used as a deterministic pseudo-random identifier to drive tuple, attribute, and bit selection during embedding; and to reproduce the same selection during detection (see Section 4.2). The main practical limitation of PK-dependent synchronization is that it fails as soon as the original PK is not preserved in the released relation. In such cases, the keyed hash values change for essentially all tuples, so the detector recomputes a different selection pattern than the one used at embedding time, resulting in global misalignment and failure to synchronize the watermark even if the embedded marks remain present. This limitation aligns with the malicious PK deletion/replacement scenario explicitly included in our threat model (see Section 2.3).

To address this limitation, researchers have proposed VPK schemes that construct a synchronization set S from content-derived identifiers rather than from $r_j.K_P$, enabling watermark detection when the original PK is unavailable. This design shift, however, moves the synchronization burden from PK uniqueness to the quality and stability of the generated identifiers: VPK sets may contain duplicate values and may change when attributes are deleted or updated. These issues and their impact on synchronization robustness are examined throughout this survey. Beyond VPK schemes, other approaches have been introduced to ensure watermark synchronization, either by employing low-quality VPK sets or by avoiding the use of both PK and VPK altogether. Fig. 2 presents a chronological view of VPK-related solutions proposed so far, identified according to the screening criteria described in Section 1. It highlights representative schemes across successive evolutionary stages, which are reviewed in detail in this section.

4.1. Structure of the VPK set

Knowing the VPK set structure is critical to understanding how S can contribute to watermark synchronization. By knowing the quality of S before performing watermark synchronization, it is possible to prevent watermarking attempts until a proper VPK set is chosen. This saves resources and time that would otherwise be wasted on obtaining mediocre results. Furthermore, a formal definition of S facilitates the design of more functional VPK schemes, addressing the challenges presented in Section 3 in the most effective manner.

The structure of S was formally proposed in [1]. It comprises different groups of VPK values, including duplicate ones, which are a direct consequence of the *duplicate problem*. The entire set of duplicate values is denoted as G, and comprises smaller groups, each composed of the different values of repeated VPKs. In the representation of S, those groups are shown in order of cardinality, from the lowest to the largest.

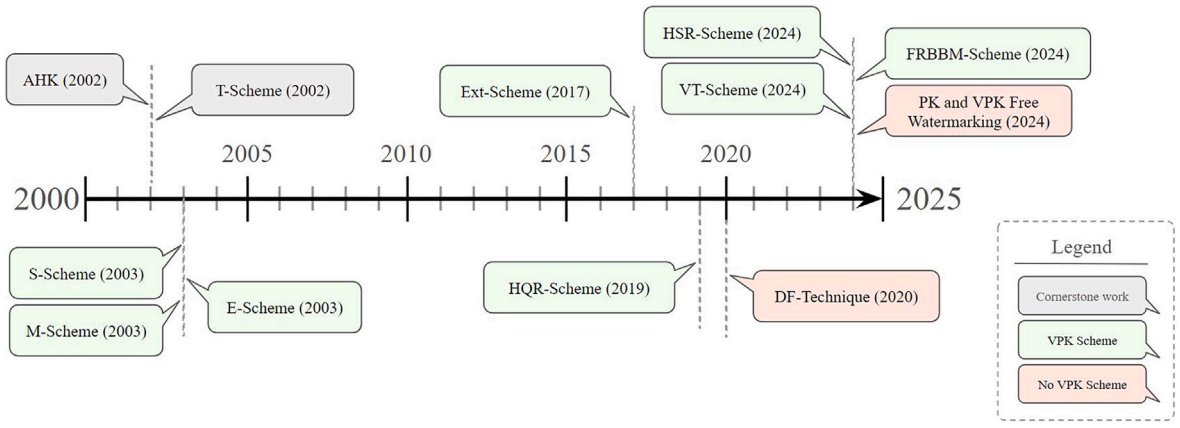


Fig. 2. Chronological view of VPK-related solutions for relational data watermarking proposed so far, including VPK schemes and related non-VPK approaches reviewed in this survey.

Table 3
Examples of G and D in S.

S	$\|G\|$	$\|D\|$
{34, 34, 34, 34, 75, 75, 75, 75, 75, 75}	10	2
{15, 15, 34, 34, 75, 75, 75, 75, 75, 75}	10	3
{15, 15, 34, 34, 41, 41, 41, 75, 75, 75}	10	4

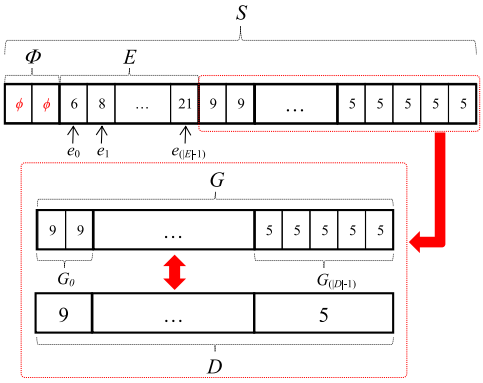


Fig. 3. Graphical representation of S structure depicting exclusive values E, duplicate groups G_r (summarized by D), and null values Φ [1].

Another set representing duplicate values is D, which is composed only of one item per duplicate group. The main reason to define D is to know how many distinct values are contained in G, which allows for computing the quality of S. Conceptually, D is contained in G. Therefore, the cardinality of G will always be higher than the cardinality of D (formally, $|G| > |D|$).

Table 3 contains some examples of S with their respective cardinalities of G and D. In the table, each group of duplicate values is highlighted with a different color. To identify each group inside G, it is utilized G_r, formally defined as $G_r \subset G : r \in [0, |D|)$. There is no need for a notation to identify a specific item within a duplicate group, since all keys in the group store the same value.

Ideally, S should consist only of unique VPKs, defined as exclusive values, but the *duplicate problem* complicates this goal. Nevertheless, some exclusive values can always be found in S. The set of exclusive values is denoted by E, where each value is represented as $e_x : x \in [0, |E|)$. The keys in E are presented in ascending order by their values.

There is another set in S, denoted as Φ, that is composed of all VPKs with *null* values. This set is used for cases in which a VPK should have been generated, but the values considered in R did not fulfill the conditions defined by the VPK scheme. An example of this is the use of most significant bits (*msb*) values that occasionally might be higher than the number of bits of some attribute values. Notice that Φ is composed of elements storing *null* values and does not represent the empty set (i.e., $\Phi \neq \emptyset$). The presence of Φ is not usual for all VPK schemes, but is formalized to represent a standard structure of S.

Fig. 3 depicts the graphical view of S structure, highlighting the differences between G and D. Formally, G, D, E, and Φ, are subsets of S.

4.2. Timeline of VPK schemes

In 2002, Agrawal & Kiernan proposed the first watermarking technique for relational data [9]. Their approach targeted numeric attributes to embed the marks in one of the ξ-th least significant bits (*lsb*). The core of this technique was named the AHK algorithm, and has been used as the backbone of most relational watermarking techniques ever since.

Unless otherwise stated, once a scheme defines the tuple identifier (PK or VPK), watermark embedding and detection follow the generic AHK-style synchronization pipeline described in Section 2.4.

The AHK algorithm enables the marking of each tuple independently. It selects the tuples, the attribute within the tuples, and the *lsb* used to embed the mark, by combining K_S and K_P according to (1). In the equation, $\mathcal{F}(\cdot)$ stands for the function that generates a tuple identifier, $\circ$ is the concatenation operator, and $\mathcal{H}(\cdot)$ is a standard one-way hash function, such as SHA-256 or SHA-512.

$$\mathcal{F}(r_j.K_P) = \mathcal{H}(K_S \circ \mathcal{H}(K_S \circ r_j.K_P)) \tag{1}$$

This equation maps the primary key value $r_j.K_P$ into a deterministic numeric identifier that seems random due to hashing. The secret key K_S acts as a hidden component: anyone with K_S can recompute the same identifier for the same tuple, while changing K_S yields a completely different identifier. The inner hash binds $r_j.K_P$ to the secret, and the outer hash further mixes the result to reduce structural patterns in the output.

The AHK algorithm takes as input R, K_S, ξ, and the tuple fraction $\gamma \in [1, \eta]$. The algorithm also utilizes ν and η, which can be obtained from R. Only tuples fulfilling the condition $\mathcal{F}(r_j.K_P) \bmod \gamma = 0$, where mod is the modulo operator, are selected for mark embedding. The closer γ is to 1, the higher the number of tuples selected. If $\gamma = 1$, all tuples are chosen.

For each one of the selected tuples, one attribute is chosen utilizing the expression $\mathcal{F}(r_j.K_P) \bmod \nu$, which returns the attribute index. For selected attributes, the position of the bit for mark embedding is obtained according to $\mathcal{F}(r_j.K_P) \bmod \xi$. In general, the AHK algorithm allows for checking all tuples, and for each tuple, all attributes.

4.2.1. T-scheme

The first VPK scheme was named T-Scheme (from Tuple-based scheme), considering how it performs tuple selection according to the AHK algorithm [12]. This scheme focuses on generating VPKs relying on K_P. Therefore, it remains vulnerable to watermark synchronization loss when the primary key is deleted or replaced, one of the main *malicious operations* introduced in our threat model (see Section 2.3).

The T-Scheme core is given by (1). The way it is created makes it possible to compute VPKs involving other elements of R rather than the primary key. This forms the first step toward the still limited but diverse set of VPK schemes proposed to date.

4.2.2. S-scheme

The S-Scheme (from Simple scheme) was proposed with the AHK algorithm in [9] as an alternative for watermarking relations that lack primary keys. It computes S by splitting the binary value of attributes into two groups: the most significant bits (*msb*) involved in VPK generation, and *lsb*, available for mark embedding. Usually, *msb* and *lsb* are selected from different attributes. Nevertheless, if R contains only one attribute, the S-Scheme uses it for both VPK generation and mark embedding, provided that *msb* and *lsb* do not overlap.

This scheme computes the VPKs according to (2), where β represents the most significant bits, $[r_j.A]_2$ is the binary representation of the value stored in the attribute A of the tuple r_j (assuming that R comprises only one attribute), and $[\mathtt{VPK}([r_j.A]_2, \beta)]_{10}$ is the integer value formed by the β bits of the attribute in decimal notation.

$$\mathcal{V}(r_j) = \left[\mathtt{VPK}([r_j.A]_2, \beta)\right]_{10} \tag{2}$$

This equation defines the virtual identifier $\mathcal{V}(r_j)$ by extracting the first β (most significant) bits from the binary representation of a chosen attribute value, and then interpreting those β bits as an integer. In effect, $\mathcal{V}(r_j)$ is the "prefix" of the attribute value, used as a surrogate identifier for the tuple.

When using AHK algorithm with S-Scheme, $r_j.K_P$ is replaced in (1) by $\mathcal{V}(r_j)$, which results in (3).

$$\mathcal{F}(r_j.K_P) = \mathcal{H}(K_S \circ \mathcal{H}(K_S \circ \mathcal{V}(r_j))) \tag{3}$$

Eq. (3) keeps the AHK identifier-generation structure unchanged, but replaces the PK input with the VPK value $\mathcal{V}(r_j)$. Thus, all subsequent AHK selection steps behave as if $\mathcal{V}(r_j)$ were the tuple key. This means that tuples with the same $\mathcal{V}(r_j)$ yield the same hashed identifier and follow the same selection decisions.

Despite the possibility of using different attributes for cases where R is composed of more than one, this scheme is highly vulnerable to the *deletion problem*. This fragility is especially relevant under the attribute deletion/modification operations considered in Section 2.3, since removing or altering an attribute involved in $\mathcal{V}(r_j)$ can desynchronize identifiers between embedding and detection. Furthermore, considering that β remains constant throughout the entire computation of S, the number of duplicate values it generates is exceptionally high.

4.2.3. E-scheme

The E-Scheme (from Element-based scheme) was created in 2003 by Li et al. [12] to compute a VPK per value stored in R. It is built very similarly to the S-Scheme and can be considered a high-scale application of it. The E-Scheme utilizes all η tuples and ν attributes of R, resulting in $|S| = \nu \times \eta$, contrary to $|S| = \eta$, which is the case for the S-Scheme.

The E-Scheme employs two distinct hash functions. The first one, denoted as $\mathcal{H}_1(\cdot)$, is used to decide if a value is marked, which occurs if $\mathcal{H}_1(K_S \circ \mathcal{V}(r_j.A_i)) \bmod (\gamma \times \eta) = 0$. Notice that $\mathcal{V}(r_j.A_i)$ is a variation of (2) for using all tuples and attributes. The second function, denoted as $\mathcal{H}_2(\cdot)$, is used to select the *lsb* to embed the mark, which is obtained according to $\mathcal{H}_2(K_S \circ \mathcal{V}(r_j.A_i)) \bmod \xi$.

This scheme performs the computation of S with the same β value throughout, resulting in the addition of too many duplicate values. Furthermore, it is critically affected by the *deletion problem*, considering that for any attribute deleted from R, η items from S are compromised. This matches the attribute deletion scenario considered in Section 2.3, where removing a single attribute can desynchronize all identifiers derived from that column.

4.2.4. M-scheme

The M-Scheme (from Multiple-elements scheme) was created in 2003 by Li et al. [12] to compute a VPK per tuple using different attributes whenever possible. This scheme also splits the binary value of each attribute into *msb* and *lsb*. Then, it computes the VPK by concatenating the two results of $\mathcal{H}(K_S \circ \mathcal{V}(r_j.A_i)) : i \in [0, \nu)$ that are closest to zero. In cases where more than two values match the same lowest result, they can also be utilized in the VPK generation.

The M-Scheme can be utilized considering more than two attributes. The initial condition of selecting only the two attributes that generate the hashes closest to zero is not mandatory. With this scheme, it is expected to obtain a VPK set that contains a high number of duplicate values, considering that β remains constant during S computation.

Table 4
Criteria followed by the Ext-Scheme to compute β.

Inputs		Output
$b_{ji(\hat{\beta}_{ji}+1)}$	$b_{ji(\hat{\beta}_{ji}-1)}$	β
0	0	$\hat{\beta}_{ji}$
0	1	$\hat{\beta}_{ji} - -1$
1	0	$\hat{\beta}_{ji}$
1	1	$\hat{\beta}_{ji} + 1$

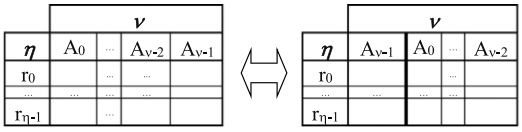


Fig. 4. Cyclic attribute-ordering model used by the HQR-Scheme to traverse attributes and diversify multi-attribute selection during VPK generation [8].

4.2.5. Ext-scheme

The Ext-Scheme (from Extended scheme) was created in 2017 by Pérez Gort et al. [13] to compute one VPK per tuple, aiming at varying the attributes considered each time. This scheme relies on the AHK algorithm to analyze all tuples and attributes of R, while trying to vary β each time using the binary length of the attribute value as a reference. For this, first, a temporary value $\hat{\beta}_{ji}$ is generated using (4). In the equation, $|[r_j.A_i]_2|$ represents the length of the binary value of $r_j.A_i$, β_f is the *msb* fraction (which remains the same during the entire computation of S), whereas $\lfloor\cdot\rfloor$ denotes the floor function.

$$\hat{\beta}_{ji} = \lfloor \frac{|[r_j.A_i]_2|}{\beta_f} \rfloor \tag{4}$$

This equation converts the bit-length of the attribute value into a candidate number of most significant bits to use: it takes the total number of bits in $r_j.A_i$ and divides it by a fixed fraction β_f, then rounds down. As a result, larger-magnitude values (longer binary representations) tend to yield larger $\hat{\beta}_{ji}$, so the scheme adapts the most significant bit count to the scale of the value rather than using a single constant β for all tuples and attributes.

After obtaining $\hat{\beta}_{ji}$, the scheme applies an *msb selector* (see Table 4) that adjusts $\hat{\beta}_{ji}$ by at most one bit based on two neighboring bit values, yielding $\beta \in \{\hat{\beta}_{ji} - 1, \hat{\beta}_{ji}, \hat{\beta}_{ji} + 1\}$. The criteria are user-configurable, but the default rules in [13] are reported here for completeness.

Once β is obtained, the attribute is considered for the VPK generation of its tuple if $b_{ji1} \oplus b_{ji(\beta-1)} = 1$. In the expression, b_{ji1} and $b_{ji(\beta-1)}$ denote the first and $(\beta-1)$-th most significant bits of $r_j.A_i$, and $\oplus$ represents the `XOR` operator. Selecting $r_j.A_i$ for the generation of VPK means adding $\texttt{VPK}([r_j.A]_2, \beta)$ to the values involved in computing $\mathcal{V}(r_j)$.

The Ext-Scheme generates a high number of exclusive values compared to previous schemes, and it is also more resilient to the *deletion problem*. Nevertheless, it does not always guarantee the embedding of enough marks, depending on the watermark length. Additionally, the process often wastes too many tuples by establishing too many conditions for the attribute's involvement in the VPK generation.

4.2.6. HQR-scheme

The HQR-Scheme (from High Quality and Resilient scheme) was proposed in 2019 by Pérez Gort et al. [8]. It relies on the cyclic model of attribute ordering to generate a VPK per tuple while varying the way attributes are parsed [8]. Fig. 4 shows the cyclic model of attribute ordering, which is based on the idea of making the first and last attributes of R neighboring elements.

The HQR-Scheme uses the Ext-Scheme as a base, considering a higher value of β by excluding from the binary string of $[r_j.A_i]_2$ only the *lsb* available for mark embedding. It focuses on varying for each tuple the position of the attribute selected to initiate the analysis, the direction of the analysis (by increasing or decreasing i in A_i), and the attributes considered.

This scheme introduces ℓ as the limit of attributes per tuple that can be involved in VPK generation. It defines the minimum distance between neighboring attributes, given by $p \in [1, \lfloor(\nu - \ell)/\ell\rceil]$, where $\lfloor\cdot\rfloor$ represents the rounding function. To choose the starting point of analysis, the attribute virtual key, denoted as A_{vk}, is computed using (5).

$$A_{\text{vk}} = [\texttt{VPK}([r_j.A_i]_2, |[r_j.A_i]_2| - \xi)]_{10} \tag{5}$$

In the equation, the term $|[r_j.A_i]_2| - \xi$ means "all bits except the ξ least significant ones." Thus, (5) builds A_{vk} by taking the binary representation of $r_j.A_i$, discarding the ξ *lsb* reserved for embedding, and converting the remaining prefix bits into an integer. In practice, A_{vk} is a per-attribute content-derived identifier designed to be stable under changes confined to the ξ embedding bits.

Then, A_{vk} is used to compute the attribute virtual value, denoted as A_{vv}. This is done according to $A_{\text{vv}} = \mathcal{H}(A_{\text{vk}} \circ K_S)$. This step maps each A_{vk} into a deterministic pseudo-random-looking number using the secret key K_S. The resulting A_{vv} values can be seen as secret, reproducible "scores" assigned to attributes within a tuple. Finally, the maximum value of A_{vv} and the average of all A_{vv}

values are obtained for the tuple. They are stored in A_{Mv}, and A_{Av}, respectively. These values summarize the computed attribute virtual values of the tuple into two reference points used by the subsequent selection rules.

The scheme initializes traversal using A_{Mv}: if $A_{\mathrm{Mv}} \bmod 2 = 0$, it moves right by steps of p; otherwise, it moves left by p along the cyclic attribute order. To further diversify attribute participation, it uses the parity of $\mathcal{Z}([A_{\mathrm{Mv}}]_2)$ to restrict candidates to attributes above or below A_{Av}. Attributes that satisfy these rules are accumulated until ℓ attributes have been included, which prevents indefinite cycles while ensuring multi-attribute involvement.

The HQR-Scheme tries to leverage all elements of R to compute S. It generates a higher number of exclusive values than the Ext-Scheme. Nevertheless, it produces too many duplicate values compared to the number of tuples in R. Given the variations in the number of attributes involved for each tuple, the starting point of analysis and its direction, the HQR-Scheme experiences a high resilience against the *deletion problem*.

4.2.7. FRBBM-scheme

The FRBBM-Scheme (from Flexible Ratio VPK generation Based on Binary Matching) was proposed in 2024 by Liang et al. [14] to generate one VPK per tuple involving more than one attribute. In their work, the authors refer to the rate at which each attribute is considered as the "attribute selection ratio". The primary objective of this scheme is to increase resilience against the *deletion problem* by adjusting the attribute selection ratio, thereby augmenting the randomness of the process. This approach focuses on controlling the involvement of each attribute separately using binary matching.

The FRBBM-Scheme converts the ratio of each attribute into binary strings, which are stored in a set denoted by $\mathbf{R}$. Each ratio is identified as $\mathbf{R}_i : i \in [0, \nu)$. For each tuple, the value of each attribute is compared with its corresponding binary ratio, and only the values that match are involved in the VPK generation.

The matching principle in [14] can be summarized as follows: each attribute A_i is assigned a binary pattern derived from its selection ratio $\mathbf{R}_i$, and the attribute value is admitted to VPK construction only if its binary suffix matches one of the admissible patterns induced by $\mathbf{R}_i$. Operationally, the scheme checks a small set of suffix conditions (from least significant bits upward) and, upon the first successful match, includes the corresponding value in the per-tuple candidate set; otherwise the attribute is skipped for that tuple.

Same as the HQR-Scheme, the FRBBM-Scheme obtains β by considering the entire binary value of the attribute, excluding only the *lsb*. First, the scheme creates A_{vv} for each value in the tuple and then stores the maximum value of A_{vv} in A_{Mv}. Next, it computes A_{C} for each attribute by cross-splicing the bits from each A_{vv} with A_{Mv}. This operation increases the variability in the values used for matching.

For each tuple, matched A_{C_i} values are collected into $A_{\mathcal{V}}$. The final VPK is then obtained by splicing elements of $A_{\mathcal{V}}$ in ascending order if A_{Mv} is even, and in descending order otherwise.

This scheme enables the computation of fewer duplicate values and more exclusive ones. It does not rely on the order of tuples, but needs to follow the position of the attributes, so the correct binary string of ratios is used to match values on each tuple. The FRBBM-Scheme does not include any criteria based on the average of attribute values, which increases resilience to the *deletion problem*, considering that when an attribute is deleted, it is less probable to delete the maximum value than to affect the average of values contained in the tuple.

With this scheme, it is also possible to balance the ratio of each attribute to obtain VPK sets with different qualities, resulting in varying watermark synchronization degrees. This also results in variations in terms of resilience against the *deletion problem*, considering that when attribute selection ratios vary, different values may be involved in VPK generation.

4.2.8. VT-scheme

The VT-Scheme was proposed in 2024 by Tan et al. [15] to generate one VPK per tuple using multiple attributes. Its name stands for Variational autoencoder (VAE) dimensionality reduction and median Tree encoding, the two steps employed to compute S. Fig. 5 depicts a general view of the architecture of the VT-Scheme.

In the first step, VT uses a trained VAE encoder to map each tuple's attribute vector to a latent representation $\vec{Z} \in \mathbb{R}^h$, thereby distributing identifying information across multiple dimensions and reducing dependence on any single attribute. The model is trained

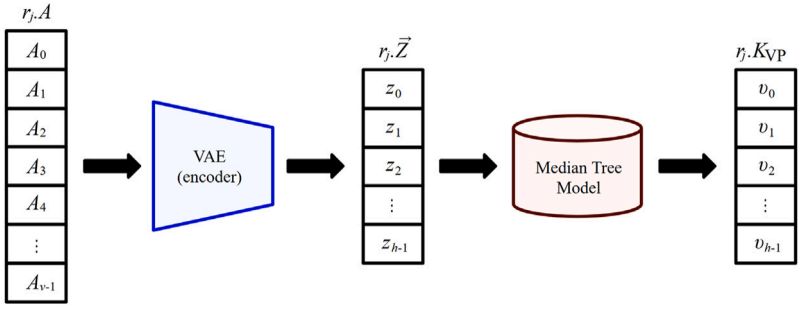


Fig. 5. VT-Scheme workflow: a variational autoencoder maps tuples to a latent vector $\vec{Z}$, which is then encoded by a median-tree model to produce h-bit VPKs with balanced distributions [15].

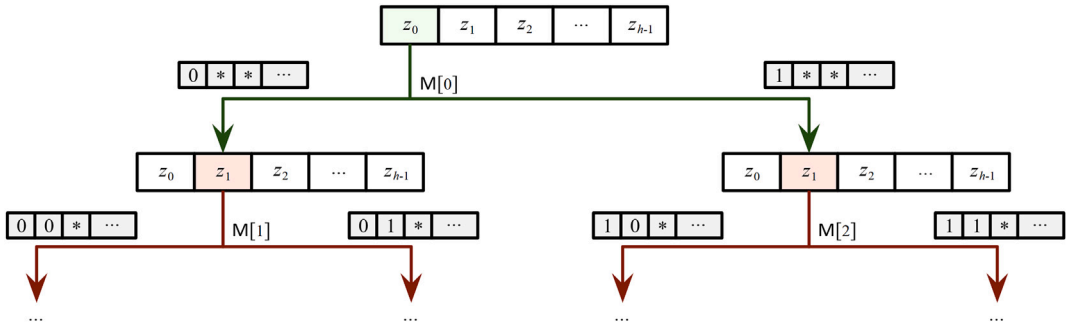


Fig. 6. Median tree encoder used in the VT-Scheme: recursive median splits of latent coordinates produce approximately balanced binary codes for VPK assignment [15].

on a subset of tuples and then applied to the remaining tuples to obtain their latent codes; latent codes from the training set are retained to construct the median-tree thresholds used in the second step [16].

To use the dimensionality reduction properly, it is important to ensure that $\nu > h$, where h is the length of the binary string representing the VPK. The VAE model takes as input β bits of all ν attributes and produces the hidden layer vector $r_j.\vec{Z}$ of dimension h. Each item of the vector is identified as $z_k : k \in [0, h)$. This scheme focuses on satisfying the condition $2^h \geq |W|$, where $|W|$ is the length of the watermark used as a fingerprint. This constraint can be interpreted as a capacity requirement: h binary decisions can generate up to 2^h distinct VPK values, which must be at least $|W|$ in order to address the $|W|$ fingerprint positions (or bins) without exhausting the code space. Equivalently, choosing $h = \lceil \log_2 |W| \rceil$ is the minimum bit-length that guarantees this capacity. This condition is essential for the median tree to uniformly encode $|W|$ fingerprint bits in binary conditions.

For the second step, VT applies median-tree coding to binarize $\vec{Z}$ into VPKs of the form $K_{\mathrm{VP}}(v_0, \ldots, v_{h-1})$ while promoting a balanced code distribution. Instead of using a fixed threshold (e.g., 0) for each dimension, the median tree uses data-dependent median thresholds so that each split is approximately balanced, reducing skew-induced duplication.

The median tree coding model creates a mapping for all tuples r_j from their hidden layer vector $r_j.\vec{Z}$ to the binary string representing the VPK. The model first generates the median tree and then uses it for encoding. The generation is performed based on the set Z 0 from $\vec{Z}$, computed after all training set tuples are processed by the VAE model. The median set M is generated continuously by bipartite for Z 0. Then, for the encoding, the model takes $\vec{Z}$ of all test set tuples and computes the corresponding VPKs.

Considering $h = \lceil \log_2 |W| \rceil$, the median tree recursively partitions the training-set latent values using per-dimension medians, yielding 2^h leaf regions and a median set M with $2^h - 1$ thresholds. For encoding, each test tuple is routed through the tree by comparing its latent coordinates to the corresponding medians, producing the binary code $(v_0, \ldots, v_{h-1})$ (see Fig. 6).

This scheme divides the latent space recursively and evenly at each level, using the median of the data along each dimension. This contributes to dividing the data roughly in half on every iteration, producing the binary code used as VPK that represents a balanced portion of the dataset, and enables the generation of VPK values that are evenly distributed. The VT-Scheme enables the uniform distribution of VPKs, ensuring values are spread out as evenly as possible. This makes it difficult to obtain exclusive values, considering that the cardinality of duplicate value groups will be as even as possible. Thus, making each VPK likely correspond to about the same number of tuples.

4.2.9. HSR-scheme

The HSR-Scheme (from High-Synchronization and Robust scheme) was proposed in 2024 by Yang et al. [17] to involve PKs in watermark synchronization when they are not modified. This approach is more like an architectural framework with a new VPK scheme involved. In this scheme, PKs are not utilized for VPK generation; instead, two embeddings are performed: one is PK-based (defined as a physical embedding) and the other is VPK-based (defined as a logical embedding). Usually, this scheme computes more than one VPK per tuple. The authors state that the logical embedding is performed using the VPKs, considering that, since multiple VPKs are generated per tuple, their use for physical embedding would result in significant data distortion.

For VPK generation, the attributes are first divided into two categories, separating those available for mark embedding from those available for VPK generation, which are stored in a set denoted as A. To reduce duplicate values, the values in A are concatenated with the string corresponding to the attribute's name and stored in a set denoted as Φ_A. Appending the attribute name makes identical raw values from different columns distinguishable (e.g., `23` in `Age` versus `23` in another attribute), reducing the chance that different attributes collapse to the same input string and thus generate the same VPK solely because their stored values coincide. An example of Φ_A obtained for a particular tuple is `{'23◦Age','CS◦Dept'}`. In this case, the tuple stores 23 for the attribute `Age`, and `CS` (standing for *Computer Science* department), for the attribute `Dept`.

After combining values and attributes' names, all possible combinations of elements in Φ_A are given as input to the function $\mathcal{H}(\cdot)$, and the results, which are formally the VPKs, are stored in a dictionary. Here, "all possible combinations" can be understood as hashing every *non-empty subset* of Φ_A (singletons, all pairs, all triples, etc.). For a set of size $|A|$, this yields at most $2^{|A|} - 1$ distinct VPK candidates per tuple when all hashed inputs are unique. Moreover, a fixed element $\kappa \in \Phi_A$ appears in exactly half of these subsets, i.e., $2^{|A|-1}$ of them, which explains why each item participates in $2^{|A|-1}$ VPKs. The total number of VPKs computed for a tuple represented by Φ_A, including duplicate values, is $\sum_{i=1}^{|A|} \left\lfloor \binom{|A|}{i} \cdot r \right\rfloor$, whereas excluding duplicate values, is $2^{|A|} - 1$.

Table 5
VPKs generated for Φ_A = {'23◦Age', 'CS◦Dept'}.

VPK Access Key κ		VPKs
'23◦Age'	→	{$\mathcal{H}$('23◦Age'), $\mathcal{H}$('23◦Age|CS◦Dept')}
'CS◦Dept'	→	{$\mathcal{H}$('CS◦Dept'), $\mathcal{H}$('23◦Age|CS◦Dept')}

This is why this scheme creates more than one VPK per tuple. To access the VPKs, the element involved in the VPK generation (denoted as κ) is used as the key. According to the HSR-Scheme, four VPKs are computed for the previous example, two corresponding to each element κ contained in Φ_A, one of them overlapping (see Table 5). The overlap occurs because the combined value corresponds to hashing a subset that contains both elements, so the same VPK appears under each key κ.

This approach generates a higher number of VPKs than previous schemes. It also assigns a greater role to attribute names, given that the VPK links to each column. The authors state that these features contribute to increasing resilience against the *deletion problem* while increasing watermark capacity.

In terms of watermark synchronization, the physical embedding uses PK to perform actual changes in the least significant bit of selected carriers by storing the mark. On the other hand, logical embedding uses the VPKs and does not act directly on the attribute bits. All embeddings in this scheme are performed using AHK. This means that only tuples satisfying the condition $\mathcal{F}(r_j.K_P) \bmod \gamma = 0$ are selected for logical and physical watermarking. This rule deterministically selects a fraction of tuples (approximately $1/\gamma$ under a uniform-hash assumption) and forces both the physical (PK-based) and logical (VPK-based) embedding to operate on the same tuple subset, keeping the two embedding layers aligned during detection. In the detection process, the watermark extraction is performed first using PK and then VPK.

Another peculiarity of this approach is the generation of a record table $\tilde{P}$ to store auxiliary information for watermark detection. The logical embedding is performed by updating the tuple in $\tilde{P}$ that corresponds to the tuple being watermarked in R. Additionally, despite being classified as physical embedding, the corresponding record in $\tilde{P}$ is updated when a mark is embedded into R.

Logical embedding is performed using each element κ to access the VPKs in the dictionary. For each VPK, a mark is selected from W and is compared to the one selected with PK. If they match, 0 is added to the string corresponding to κ. Otherwise, 1 is added. Thus, for each tuple and each key κ, the resulting binary string stored in $\tilde{P}$ acts as a compact signature of consistency checks across all VPKs generated from that tuple. After performing this process for all the VPKs of the tuple, the attribute in $\tilde{P}$ created for logical embedding is updated with the array of binary strings. That value corresponds to the logical embedding of the tuple being watermarked in R.

This scheme creates a complex watermark structure linked to the mark selected using PK. Its framework relies on redundancy and external structures, such as $\tilde{P}$, to validate the watermark detection. Combining physical and logical embedding makes it possible to double-check the values and determine with high precision if the watermark is contained in $\hat{R}$. It also enables the restoration of data after watermark extraction. Nevertheless, it depends highly on the accessibility and integrity of $\tilde{P}$. Consistent with the assumptions in Section 2.3, we treat such auxiliary protected structures as available to the legitimate detector but not accessible to the adversary.

4.2.10. Other VPK schemes

In addition to the above schemes, there are other proposals very similar to the S-Scheme and the M-Scheme. In 2014, Chang et al. [18] created a VPK scheme for techniques using textual attributes to synchronize the watermark. Their approach is very similar to the S-Scheme. It works by splitting each textual value into two fragments, identified by A_i^1 and A_i^{-1}. This scheme stores the last word of the targeted attribute in A_i^1, leaving it available for mark embedding. On the other hand, it stores the rest of the text value in A_i^{-1}, which is utilized for VPK generation by computing a hash that represents the attribute.

In 2016, Khanduja et al. [19] proposed a scheme that works similarly to the M-Scheme. Its main difference is that the M-Scheme selects for VPK generation the two attributes with the hash values closest to zero. Nevertheless, the number of attributes can vary according to the data owner's specifications. On the contrary, the proposal by Khanduja et al. considers always only two attributes (identified as A_1 and A_2), which increases the vulnerability of the scheme to the *deletion problem*.

4.3. Non-VPK approaches

Instead of focusing on obtaining VPK sets with higher quality, other approaches aim to improve watermark synchronization when using low-quality VPK sets or to avoid the use of PKs and VPKs altogether. They are formally outside the category of VPK schemes and closer to watermarking techniques. This section discusses two approaches that illustrate how watermarking can be implemented in accordance with these principles.

4.3.1. DF-technique

One example of a technique utilizing low-quality VPK sets for watermark synchronization is the *double fragmentation* approach proposed in [1], formally known as the DF-Technique. This proposal utilizes duplicate values in S to fragment W twice, thereby increasing the number of embedded marks. Fig. 7 presents a general view of the DF-Technique.

The DF-Technique comprises two processes. The first generates elements defined as *first fragments* corresponding to W segments composed of more than one mark. *First fragments* are identified by $M_r : r \in [0, |D|)$. There are as many of them as groups of duplicate

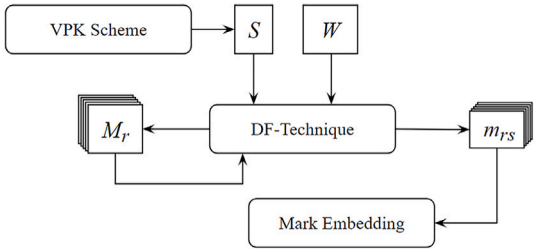


Fig. 7. Architecture of the *double fragmentation* approach: duplicate-group information in S is used to fragment the watermark twice, increasing embedding opportunities under low-quality VPK sets [1].

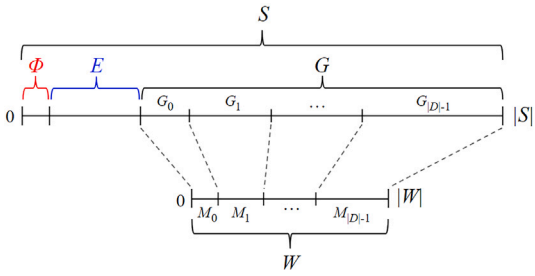


Fig. 8. DF-Technique proportional allocation: fragment lengths $|M_r|$ are assigned in proportion to duplicate-group sizes $|G_r|$, enabling consistent mark selection across groups [1].

values in S. The second process generates the *second fragments*, which are marks contained in the *first fragments*. *Second fragments* are identified by $m_{rs} : s \in [0, |M_r|)$, where M_r is the corresponding *first fragment*.

Contrary to the equality between the number of *first fragments* and the number of groups of duplicate values, the number of *second fragments* in M_r is lower than the cardinality of its corresponding subset G_r (i.e., $|G_r| > |M_r|$). This occurs because the number of tuples in R should be much higher than the watermark length, formally $\eta \gg |W|$. Therefore, equivalence must be defined in that case.

The *first fragments* are obtained using W, $|D|$, and the cardinality of each subset G_r, which is utilized to define a proportionally equivalent size for the different instances of M_r. Fig. 8 shows a graphical view of the differences in proportion between the lengths of *first fragments* and the cardinality of the groups of duplicate values.

Proportions between $|M_r|$ and $|G_r|$ are obtained using (6). Thus, after selecting a duplicate VPK for a given tuple r_j, it is possible to obtain the fragment M_r to embed the mark in r_j.

$$|M_r| = \lfloor \frac{|G_r| \times |W|}{|G|} \rceil \tag{6}$$

A critical step for selecting the mark to embed in r_j (denoted as w_{r_j}) is given by (7). In the equation, the mark is conceptually the second fragment m_{rs}.

The equation takes as input the VPK duplicate value (denoted as $s_j \in S$ when focusing on VPKs per tuple), the fragment $|M_r|$ to perform the pseudo-random selection of the mark, and a seed generated for r_j denoted as $\texttt{seed}_j$. The seed generation is performed for each tuple to increase the entropy of mark selection, thereby increasing the number of different marks selected for the embedding.

Finally, L is a value proportional to the sum of the cardinalities of all duplicate value groups located before the targeted fragment G_r. The term L is required, considering that the position of the selected mark is relative to its group.

$$w_{r_j} = W(\mathcal{H}(s_j, |M_r|, \texttt{seed}_j) + L) : j \in [0, \eta) \tag{7}$$

This approach focuses on duplicate values from S. For exclusive values, the DF-Technique performs watermark embedding by following the process defined in the attached watermarking technique, as if they were PKs.

The success of the DF-Technique depends strongly on the watermark fragmentation and the seed generator. If seeds are obtained using R, given the limited content in the relation, duplicate seeds are to be expected. Nevertheless, the pair VPK + seed should present fewer duplicates than when considering just one element for watermark synchronization, especially if the seed generator involves chaos-based generation.

4.3.2. PK-free and VPK-free watermarking

An example of a watermarking approach that does not use PK or VPK is the work proposed in 2024 by Che et al. [20]. In this case, the authors developed a technique for watermarking numerical tabular datasets used to train machine learning models. This approach embeds a sinusoidal signal as the watermark into a discrete-time signal built from the dataset. The technique does not rely on VPK schemes aiming to prevent the consequences of the deletion problem.

To prevent the degradation of the numerical dataset's usability due to watermark embedding, the authors defined the function $\texttt{MLU}(\cdot)$ to compute the machine learning utility. The function takes the dataset as input and indicates its effectiveness in training good machine learning models. The machine learning utility is measured by the performance of a machine learning model trained on R, assuming that the relation has only numerical attributes.

This technique comprises two steps: (i) mapping data instances in a tabular dataset to a *discrete-time signal*, and (ii) watermark embedding by adding a *sinusoidal signal* to the *discrete-time signal*. For the first step, the authors design two functions, $\phi_x(\cdot)$ and $\phi_y(\cdot)$, to map each tuple to a pair of coordinates, x and y, representing a point in a two-dimensional space. Then, all the points are summarized in groups to form the discrete-time signal.

For mapping, the technique uses a pair of orthogonal vectors, denoted as $\vec{\mathbf{e}}_x, \vec{\mathbf{e}}_y \in \mathbb{R}^v$, with L_2-norm equal to one (formally $\|\vec{\mathbf{e}}_x\|_2 = 1$, and $\|\vec{\mathbf{e}}_y\|_2 = 1$). This is done to prevent collision between watermark embedding and the quality of the dataset. The mapping function $\phi_y(\cdot)$, defined in (8), represents the projection of r_j on $\vec{\mathbf{e}}_y^{\top}$.

$$y_j = \phi_y(r_j) = r_j \, \vec{\mathbf{e}}_y^{\top} \tag{8}$$

The function $\phi_x(\cdot)$ is defined in (9), where the numerator consists of a binning operation that projects r_j into a bin and returns its index, and $b \in \mathbb{R}^+$ is the bin width. By dividing the index by τ, the equation maps the index into an $x-$coordinate, where $1/\tau : \tau \in \mathbb{R}^+$ is the step size between the $x-$coordinates of neighboring bins.

$$x_j = \phi_x(r_j) = \frac{\lfloor \frac{r_j \, \vec{\mathbf{e}}_x^{\top}}{b} \rfloor}{\tau} \tag{9}$$

The coordinates (x_j, y_j) mapped from r_j represent a point in a two-dimensional space. By mapping all the tuples from R is obtained a set of points denoted by $\mathsf{Z} = \{(x_j, y_j) \mid x_j = \phi_x(r_j), y_j = \phi_y(r_j)\}$. These points cannot be used to build a discrete-time signal, considering that some of them may have the same x value but different y values. To convert them into a discrete-time signal, they are processed, by grouping those with the same x. This is represented as $B_{\mathsf{h}} = \{(x_j, y_j) \mid x_j = \mathsf{h}, (x_j, y_j) \in \mathsf{Z}\}$, where h is the value of the x-coordinates of the points in B_{h}.

Then, the points in B_{h} are summarized into a mean point, denoted by $(\bar{x}, \bar{y})$, where $\bar{x} = \mathsf{h}$ and $\bar{y}$ is the mean of the y-coordinates of the points in B_{h}. In this way, the points in Z are converted into a set of mean points with distinct x-coordinates. This set of mean points forms the discrete-time signal, denoted by T. This is how R is mapped to T, which is formally expressed in (10), where $\varphi(\cdot)$ is the mapping operation.

$$T = \varphi(R) \tag{10}$$

The second step (watermark embedding by adding a *sinusoidal signal to* T) is done by performing slight changes in R tuples. The sinusoidal signal is denoted by $y = \lambda \, \texttt{sin}(2\pi\theta x)$, where λ is its amplitude and θ is its frequency. Watermark embedding is done by updating every tuple r_j in R using (11).

$$r'_j = r_j + \lambda \, \texttt{sin}(2\pi\theta x_j) * \vec{\mathbf{e}}_y \tag{11}$$

In their work, the authors proved that, if R' is obtained that way, then a discrete-time signal T' obtained from R' using the same mapping operation $\varphi(\cdot)$, will contain a component of the sinusoidal signal with frequency θ, formally expressed as $T' = \varphi(R')$. Thus, proving the presence of the watermark in R'. This principle is also used to design watermark detection (by detecting the time signal $\hat{T}$ from $\hat{R}$). To analyze whether $\hat{T}$ contains the sinusoidal signal with frequency θ, it is required to check the spectral power $\hat{P}(\theta)$ of $\hat{T}$ at the frequency θ. For computing $\hat{P}(\theta)$, the authors use Lomb-Scargle Periodogram (LSP) according to [21].

5. Qualitative and quantitative evaluation of VPK schemes

This section presents both quantitative and qualitative evaluations of the previously described VPK schemes. The quantitative evaluation focuses on the quality of S, which is primarily influenced by how the VPK schemes address the *duplicate problem*. It also assesses the impact of the *deletion problem* on S, and consequently, on the extracted watermark for each set obtained using the VPK schemes that are sufficiently promising to ensure watermark synchronization. Conversely, the qualitative evaluation highlights the main strengths and limitations of each scheme.

To interpret the results, it is useful to view VPK generation as a mapping from tuples to an effective key space. When the mapping produces identifiers from a small or weakly varying domain (e.g., fixed β values), many tuples collide into the same VPK value, increasing duplication and reducing synchronization entropy. Conversely, schemes that (i) increase the effective key space (e.g., by combining multiple attributes or variable-length representations) and/or (ii) randomize attribute participation tend to reduce collisions and increase the fraction of distinct VPKs. Deletion resilience is governed by a different mechanism: if a scheme depends heavily on a small subset of attributes, deleting one of those attributes alters a large fraction of VPKs, causing widespread re-indexing and desynchronization; schemes that distribute participation more uniformly across attributes (or encode tuples through multi-attribute representations) tend to degrade more gracefully under attribute loss.

5.1. Evaluation metrics

The evolution of VPK schemes has been accompanied by the development of methods for measuring the quality of S. This is a critical aspect in evaluating the performance of VPK schemes when addressing the *duplicate* and *deletion problems*. The proposed

metrics facilitate the design of more effective schemes and provide a framework for their comparison. Some evaluation approaches fail to properly account for the effect of S on watermark synchronization, whereas others treat it as a mandatory consideration, as this is the primary motivation for computing S. In this section, we present the metrics proposed thus far and analyze those that are most suitable for an objective evaluation of the quality of S and of VPK schemes in general.

We use two complementary families of indicators. First, we report VPK-set quality metrics that quantify collision and uniqueness properties of S (duplicate-group behavior and the exclusiveness index ρ), which directly reflect how well tuples can be re-identified without the original PK. Second, to capture the end-to-end impact of attribute deletion on watermark synchronization, we report recovery-oriented measures (CF and SSIM) computed from the synchronized watermark images, which quantify detection effectiveness and structural similarity, respectively.

5.1.1. Duplicate index

In 2003, Li et al. [12] proposed the first metric for quantifying the effects of the *duplicate problem* in S. They defined (12) to compute the *duplicate index*, denoted as $\delta \in [0, \infty]$, where $\omega_{\max}$ and $\omega_{\min}$ represent the maximum and minimum number of times the same mark is embedded, respectively.

$$\delta = (\omega_{\max} - \omega_{\min})/\omega_{\min} \tag{12}$$

According to (12), the highest quality is achieved when all marks are embedded the same number of times. This condition is expressed as $\omega_{\max} = \omega_{\min}$, which yields $\delta = 0$. On the other hand, when any mark is excluded, $\omega_{\min} = 0$, resulting in $\delta = \infty$, which represents the worst-case scenario regardless of the number of marks excluded from watermark synchronization.

This represents a fundamental limitation of this metric. It does not adequately capture the differences in watermark synchronization that arise when varying numbers of marks are ignored, even though such differences affect watermark quality to different extents. Some watermarking techniques disregard certain marks during embedding, yet still achieve high synchronization. Consequently, ignoring a few marks from W should not lead to the same conclusion as when half of the watermark is excluded.

Another major limitation of the *duplicate index* is that it can only be computed after watermark embedding. Consequently, if S is of low quality, time may be wasted performing watermark synchronization before realizing the drawbacks of using such a VPK set. This metric does not allow for assessing the quality of S independently of the watermarking technique. Moreover, it conceptually overlooks the VPK scheme itself, thereby disregarding the potential impact of the *deletion problem* on S and on watermark synchronization.

5.1.2. Exclusiveness index

The *exclusiveness index* was proposed in 2019 by Pérez Gort et al. [8] to measure the quality of S and to evaluate its preservation after attribute deletions in R. This metric is based on the structure of S described in Section 4.1. It does not require watermark synchronization, thereby assigning greater relevance to the VPK scheme and allowing for time savings by enabling the selection of a more suitable VPK set in advance.

The motivation for introducing the *exclusiveness index* arises from the fact that the cardinality of each subset of S is insufficient to capture the differences between VPK sets. There may be cases in which the cardinalities of the S subsets are identical, yet S itself differs.

Examples of this situation are given by records 4 and 5 in Table 6. In addition to the VPK sets and the cardinalities of E, G, and D, the table also reports the value of the *exclusiveness index* computed for each case, denoted by ρ.

The examples in Table 6 illustrate that, even when subsets have equal cardinalities, it is possible to distinguish different qualities of S by computing ρ. In the table, exclusive keys are shown in black and are ordered according to their VPK values, whereas duplicate values are grouped by VPK and displayed in different colors, one for each group.

The *exclusiveness index* ρ captures the distribution of duplicate value groups stored in G in a way that reflects the total number of distinct values (not only the exclusive ones) contained in S. This concept is based on the idea that the number of distinct values in S is expected to exceed the number of marks in W, thereby enabling watermark synchronization even in the presence of duplicate values in S. Accordingly, the *exclusiveness index* identifies as optimal the VPK set containing the largest number of exclusive values, followed by the one with the greatest number of duplicate groups.

The *exclusiveness index* is computed according to (13), where $\rho \in [1, 100]$. It quantifies the distinct keys in S as a percentage relative to its cardinality. The only case in which $\rho = 100$ occurs is when S consists entirely of exclusive values. The summation in the equation accumulates the inverse of the cardinality of each group of duplicate values, implying that larger groups do not contribute to the quality of S, as they reduce the diversity of values by limiting the space for other duplicate groups or exclusive values. The *exclusiveness index* treats each exclusive value as a group of one element with a weight of 1. These are excluded from the summation

Table 6
Value of ρ for the examples of S.

S	$\lvert E\rvert$	$\lvert G\rvert$	$\lvert D\rvert$	ρ
{12, 15, 23, 35, 57, 58, 61, 76, 80, 97}	10	0	0	100
{12, 15, 23, 35, 57, 58, 44, 44, 75, 75}	6	4	2	70.0
{12, 15, 75, 75, 75, 75, 75, 75, 75, 75}	2	8	1	21.3
{15, 15, 15, 34, 34, 34, 75, 75, 75, 75}	0	10	3	9.17
{15, 15, 34, 34, 75, 75, 75, 75, 75, 75}	0	10	3	11.7
{75, 75, 75, 75, 75, 75, 75, 75, 75, 75}	0	10	1	1.00

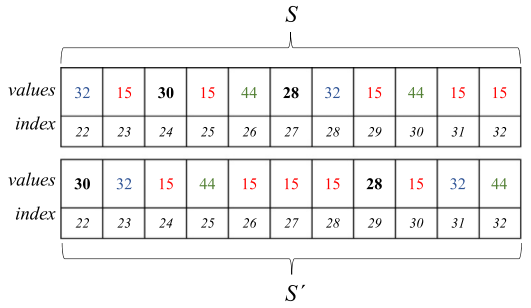


Fig. 9. Example of index–value mismatch between S and S': identical VPK values are associated with different tuple indexes, motivating calibrated metrics that account for index consistency [8].

and are instead accounted for in the cardinality of the set E.

$$\rho = \frac{\left(sum_{r=1}^{|D|} u_r / |G_r| + |E|\right) \times 100}{|S|} \tag{13}$$

With regard to the *deletion problem*, it is almost certain that after deleting attributes from R, the original VPK set will differ from the one obtained from $\hat{R}$. By measuring the differences between S and S', it is possible to estimate how watermark synchronization may be affected. In (13), the term u_r represents the payload for each group of duplicate values, formally defined as $u_r \in [0, 1]$ and computed using (14). In this equation, $a_r \in [0, |G_r|]$ represents the number of VPKs in the group that change values due to attribute deletions. In this way, the *exclusiveness index* can also be used to assess the effects of the *deletion problem* on S.

$$u_r = (|G_r| - a_r)/|G_r| \tag{14}$$

If all VPKs in a group preserve their values, then $a_r = 0$, resulting in $u_r = 1$. On the other hand, if all VPK values change, then $a_r = |G_r|$, resulting in $u_r = 0$. For exclusive values, when one of them is affected by the *deletion problem*, one unit is subtracted from $|E|$.

To use (13) solely for evaluating the quality of S, it is assumed that $u_r = 1$, since no payload changes are considered. Conversely, to assess the effects of the *deletion problem*, both VPK sets, S and S', are required in order to compute u_r for each group.

5.1.3. Calibrated exclusiveness index

The *calibrated exclusiveness index* was introduced in 2019 by Pérez Gort et al. [8] under the name *exclusiveness index precision*. We adopt the term *calibrated exclusiveness index*, as it more accurately reflects the concept underlying its purpose. This metric is based on the fact that, after an attribute is deleted, a VPK may take on the value of another VPK initially stored in S. Thus, some keys in S' could identify the wrong tuples in R. To prevent this from happening, each VPK is linked to an index, thereby avoiding reliance solely on the VPK value to identify tuples. Fig. 9 illustrates the problem of VPK replacement by value, showing a scenario in which two sets store identical VPKs but point to different tuples based on the index associated with their keys.

The problem of computing VPKs different from their respective match in S, but with values equal to other authentic keys in the VPK set, makes it hard for us to rely on the ρ for S' after some attributes are deleted. This is why the *exclusiveness index* calibration is required. This concept can be understood as the introduction of constraints in the computation of ρ, ensuring that the resulting value accurately reflects the quality of S' . Following this idea, (15) is defined to obtain the degree of calibration of the *exclusiveness index*, denoted as $\varphi_c \in [0, 1]$, and computed by considering only the number of VPKs $\mathfrak{m} \in [0, |S|]$ that match both in index and value with the total number of VPKs in S.

$$\varphi_c = \mathfrak{m}/|S| \tag{15}$$

The following example illustrates how φ_c affects the calculation of ρ. Both sets, S and S', are presented along with their respective ρ values, obtained without considering the key indices. For this example, the index of each key is given by its position within its set. Therefore, only the keys in S' that match in value and position with the keys in S are considered authentic (see underlined keys in S').

$$S = \{12, 34, \mathbf{21}, \mathbf{45}, 12, 34, \mathbf{15}, \mathbf{48}, \mathbf{22}, \mathbf{82}\}, \rho = 70.0$$

$$S' = \{\underline{12}, 12, \mathbf{23}, \underline{\mathbf{45}}, \underline{12}, \mathbf{17}, \underline{\mathbf{15}}, \mathbf{44}, \mathbf{67}, \mathbf{81}\}, \rho = 73.0$$

At first glance, S' appears to have higher quality, as it shows a greater value of ρ, which is expected given that it contains a larger number of exclusive values. However, when considering just the underlying keys, it becomes evident that the quality of S' should be lower, considering that only four keys preserve their indices. Therefore, $\rho = 73.0$ for a given $\varphi_c = 0.4$. Nevertheless, to evaluate ρ with high precision (i.e., $\varphi_c = 1.0$), only the VPKs that preserve their indices should be considered, while the remaining ones are ignored. Under these conditions, $\rho = 25.0$, providing a more accurate representation of the quality of S' with respect to S.

Table 7
Summary of the quantitative evaluation setup (dataset, watermark sources, and parameter settings).

Category	Setting used for the experiments
Dataset	Forest Cover Type (UCI repository); first 30,000 tuples (of 581,012) and first 10 numerical attributes (of 54), as commonly used in prior VPK-scheme validations (Table 8).
Watermarking technique	Sardroudi & Ibrahim [11] (meaningful watermarking using a binary image as source).
Watermark sources	{UTM (82×80), WWF (40×45), Dào (20×21), E (10×10)} (Table 9).
Global bit parameters	$\beta = 3$ for fixed-β bit-extraction schemes (S-, E-, M-Schemes); $\xi \in \{0, 1\}$ with $\xi = 0$ (no embedding space reserved) and $\xi = 1$ (reserve one *lsb* for embedding).
Tuple selection	AHK-style selection with tuple fraction $\gamma = 1$ in embedding/extraction runs (controls expected selected tuple fraction $1/\gamma$).
Scheme-specific settings (summary)	S-Scheme: evaluated per attribute (10 runs). E-Scheme: all tuple–attribute values; fixed β. M-Scheme: combines multiple attribute-derived hashes (as defined in [12]). Ext-/HQR-Schemes: scheme-defined adaptive β (via β_f and selection rules as in the original proposals). FRBBM-Scheme: per-attribute selection ratio set to 0.2 (uniform across attributes; ≈ 2 attributes selected per tuple on average). VT-Scheme: h selected to satisfy $2^h \geq \lvert W \rvert$; configurations reported for the tested watermarks (e.g., $h \in \{7, 9, 11, 13\}$). HSR-Scheme: uses auxiliary table $\bar{P}$ and multi-VPK generation as defined in [17].
Deletion experiments	One-attribute deletion simulated by nulling the deleted column; 10 runs per scheme (one per attribute) for the selected robust schemes (Section 5.2.3).
Implementation environment	Java 1.8 (client), Oracle 19c (server), Python 3.13 + NumPy + PyTorch for VT; Intel i7-7700K, 32GB RAM, Windows 10 Pro.

Denoting ρ_C as the value of ρ when $\varphi_c = 1.0$ (i.e., the *calibrated exclusiveness index*), and ρ_O as the value of ρ computed over all elements of S' (not only the matching ones), the noise introduced in S' with respect to S is quantified as $\partial = |\rho_O - \rho_C|$. This metric allows us to measure the difference between the two VPK sets. In the previous example, the noise added to S is $\partial = |73 - 25| = 48$.

5.2. Quantitative comparison

The quantitative evaluation of VPK schemes makes it possible to compare them in an objective way. We proceed using the following standardized, unbiased experimental setup.

5.2.1. Experimental setup and parameter selection

To improve reproducibility and to clarify the evaluation context, Table 7 summarizes the dataset, watermark sources, and the key parameters used throughout Section 5.2. We follow the common practice in the surveyed literature of evaluating VPK quality under a fixed-β setting when applicable, and we additionally report results under two embedding-capacity conditions: (i) $\xi = 0$ to isolate the intrinsic VPK-set properties when no space is reserved for embedding, and (ii) $\xi = 1$ to reflect the practical constraint of reserving at least one least significant bit for watermark insertion. For fixed-β schemes (S- and E-Schemes), we use $\beta = 3$ as a representative low-bit extraction setting that is widely used to expose collision/duplication behavior; for adaptive schemes, β (or its proxy) is determined by the scheme definition. The tuple fraction γ is set to $\gamma = 1$ in all embedding/extraction runs to control the expected number of selected tuples while ensuring sufficient carriers for the watermark sizes considered. For the VT-Scheme, the VPK length h is selected to satisfy the scheme's capacity condition $2^h \geq |W|$. We report the configurations evaluated in Tables 11 and 14.

For the dataset, we utilized *Forest Cover Type*.[2] We selected the first 30,000 tuples (out of 581012) and the first 10 attributes (out of 54), as this is the subset that is most commonly used for validating the schemes proposed so far. Table 8 contains the average of each attribute's binary length in column **BL(Avg)**,[3] their maximum and minimum values in columns **Max** and **Min**, respectively, and the mean and standard deviation of the numerical distribution of their values in columns **Mean** and **StdDev**, respectively. Knowing this information in advance allows us to better judge the results obtained for each attribute, considering the coverage they provide for VPK generation.

Although the *Forest Cover Type* dataset is a controlled benchmark commonly used in prior validations of VPK schemes, it captures several characteristics that are relevant to PK-free synchronization in practical relational settings. In particular, it provides structured numerical attributes with heterogeneous value distributions and moderate dimensionality, which contribute to exposing core VPK behaviors such as duplication trends, attribute involvement, and sensitivity to attribute deletion under a consistent setup.

For cases where the *msb* remains the same during the entire process, we use $\beta = 3$. Additionally, focusing on the performance of VPK generation and management, we set $\xi = 1$ to compare the schemes using the highest coverage possible. Considering that we target the study of how VPK schemes perform against the *deletion problem*, we consider attribute deletion as the only operation that may threaten watermark synchronization. Additional malicious operations may be used to analyze the resilience of watermarking techniques. We select the metric ρ to evaluate the quality of S.

To test the quality of the watermark synchronized with the different VPK sets, we selected the watermarking technique proposed by Sardroudi & Ibrahim in [11]. This technique is a meaningful approach that uses a binary image as an external source to generate the watermark, which is then embedded into R. We selected sources of different topologies and lengths to capture their role in

[2] The *Forest Cover Type* dataset is available online in the University of California Irvine (UCI) machine learning repository at http://kdd.ics.uci.edu/databases/covertype/covertype.html [22].

[3] The binary length **BL** denotes the number of bits used in the binary representation of the values stored in that attribute.

Table 8
Features of the *Forest Cover Type* numerical dataset.

No.	Attribute Name	Attribute Description	BL(Avg)	Max	Min	Mean	StdDev
1	ELEVATION	Elevation in meters	11.99	3849	1863	2780.08	322.30
2	ASPECT	Aspect in degrees azimuth	7.11	360	0	144.60	108.17
3	SLOPE	Slope in degrees	4.13	61	0	14.19	8.13
4	HOR_DIST_TO_HYDROLOGY	Horz Dist to nearest surface water features	7.13	1343	0	207.04	183.73
5	VERT_DIST_TO_HYDROLOGY	Vert Dist to nearest surface water features	4.33	554	−146	38.63	50.41
6	HOR_DIST_TO_ROADWAYS	Horz Dist to nearest roadway	11.40	7117	0	2643.32	1895.24
7	HILLSHADE_9AM	Hillshade index at 9am, summer solstice	7.99	254	0	215.53	27.04
8	HILLSHADE_NOON	Hillshade index at noon, summer solstice	8.00	254	99	221.77	19.61
9	HILLSHADE_3PM	Hillshade index at 3pm, summer solstice	7.57	248	0	136.57	38.05
10	HOR_DIST_TO_FIRE_POINTS	Horz Dist to nearest wildfire ignition points	11.67	7173	0	3210.02	2157.09

Table 9
Watermark sources used in the experiments.

W	**UTM**	**WWF**	**Dào**	**E**
Image Name	Universiti Teknologi Malaysia	World Wildlife Fund.	Chinese character Dào	Latin character E
Image				
length (pixels)	82 × 80	40 × 45	20 × 21	10 × 10

synchronization depending on S (see Table 9). We use red as a placeholder color to highlight the absence of a pixel that is missing during synchronization.

To evaluate the quality of the image built from the synchronized watermark we used the Correction Factor (CF ∈ [0, 100]) and the Structural Similarity Index (for our purposes, SSIM ∈ [0, 1]). CF is obtained with (16) by comparing the pixels of the image I_{emb} built from the embedded watermark, with the ones of the image I_{ext} generated from the extracted watermark. In the equation, h^I and w^I represent the height and width of both images. CF = 100 describes that both watermarks are identical, and CF = 0 indicates that the watermarks are entirely different.

$$\text{CF} = \frac{\sum_{i=1}^{h^I} \sum_{j=1}^{w^I} (I_{\text{emb}}(i,j) \oplus \overline{I_{\text{ext}}(i,j)})}{h^I \times w^I} \times 100 \tag{16}$$

We obtain SSIM according to (17), using multiple windows defined by x and y with common sizes $N \times N$. In the equation, μ_x and μ_y represent the averages of x and y, respectively, σ_x^2 and σ_y^2 their variances, and σ_{xy} their covariance. The symbols C_1 and C_2 are stabilization constants. When SSIM = 1, there is a perfect structural similarity between the two images. On the other hand, SSIM = 0 indicates the lack of similarity.

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{17}$$

We implemented the VPK schemes in a client–server architecture, distributing the logic between tiers according to each scheme's design and complexity. In some cases, only server-side programming was required, while in others, additional processes were implemented on the client side.

The client layer was developed using Java 1.8 and Eclipse Integrated Development Environment (IDE) 4.21. The database server engine was Oracle Database 19c, managed through Oracle SQL Developer 21.4. We also used Python 3.13 to implement the VT-Scheme, employing the python-oracledb library for database connectivity, NumPy for numerical processing, and PyTorch for the training and inference of the Variational Autoencoder (VAE).

The runtime environment was a PC equipped with an Intel i7-7700K processor (4.20 GHz), 32 GB of RAM, and Windows 10 Pro OS. To complement the qualitative discussion of computational overhead in Section 5.3, we also provide a compact summary of the dominant cost drivers and approximate complexity classes of the major VPK scheme families (see Table 20). This summary is intended as an implementation-agnostic indication of scaling behavior with respect to the number of tuples and attributes, while the empirical results reported in Section 5.2 reflect the execution environment described above.

5.2.2. Quality of VPK sets

The first set of experiments aims to evaluate the quality of the VPK set S produced by each scheme. To this end, we begin by generating S with every VPK scheme under two conditions. In the first condition, no cover is reserved for watermark synchronization, thereby maximizing the attribute content available for VPK construction. In the second, more realistic condition, VPKs are generated

Table 10
S-Scheme: features and quality of S per attribute ($\beta = 3$, $\xi = 0$).

Attribute	$\lvert\Phi\rvert$(%)	$\lvert E\rvert$(%)	Duplicate Values					ρ
			# Groups	Group's Size Range		Mean	StdDev	
				Max.(%)	Min.(%)			
ELEVATION	0	0	4	59.34	1.23	7500	7356.34	1.04×10^{-5}
ASPECT	1.96	0	4	32.56	15.97	7353	2776.68	2.03×10^{-6}
SLOPE	4.33	0	4	31.05	18.60	7175.50	1598.16	1.92×10^{-6}
HOR_DIST_TO_HYDROLOGY	8.46	0	4	27.24	14.69	6865.75	1691.35	2.06×10^{-6}
VERT_DIST_TO_HYDROLOGY	24.04	0	4	23.77	13.53	5697	1334.64	2.45×10^{-6}
HOR_DIST_TO_ROADWAYS	0.01	0	4	30.31	16.66	7499.25	1936.35	1.89×10^{-6}
HILLSHADE_9AM	0.0033	0	4	47.87	3.86	7499.75	6220.33	4.38×10^{-6}
HILLSHADE_NOON	0	0	4	54.44	0.95	7500	7605.03	1.37×10^{-5}
HILLSHADE_3PM	0.34	0	4	43.45	15.82	7474.25	3844.32	2.09×10^{-6}
HOR_DIST_TO_FIRE_POINTS	0.0066	0	4	32.72	13.18	7499.50	2497.85	2.00×10^{-6}
AVERAGE	**3.91**	**0**	**4**	**38.27**	**11.45**	**7206.40**	**3686.10**	$\mathbf{4.29 \times 10^{-6}}$

Table 11
VT-Scheme: features and quality of S for different values of h ($\xi = 0$).

VPK length	$\lvert\Phi\rvert$(%)	$\lvert E\rvert$(%)	Duplicate Values					ρ
			# Groups	Group's Size Range		Mean	StdDev	
				Max.(%)	Min.(%)			
E → $h = 7$	0	0	128	0.7833	0.78	234.375	0.4860	1.82×10^{-3}
Dào → $h = 9$	0	0	512	0.1967	0.1933	58.59	0.4916	2.91×10^{-2}
WWF → $h = 11$	0	0	2048	0.05	0.0467	14.6484	0.4776	4.67×10^{-1}
UTM → $h = 13$	0	0	8192	0.0133	0.0067	3.6621	0.4733	7.60

while ensuring that the least significant bit remains available for mark embedding. This allows us to assess the robustness of each scheme under practical watermarking constraints.

The results are presented in the same order in which the schemes were introduced. We begin by generating S with the S-Scheme. Although this scheme selects only one attribute for VPK construction, a more comprehensive assessment of the quality of the resulting sets requires evaluating each attribute independently. Therefore, we conducted ten experiments, each generating one VPK set using a different attribute of R. For the experiments in which the entire attribute value is available (i.e., without reserving space for watermark synchronization), the only requirement is that each value considered must be sufficiently long to extract the number of bits specified by *msb*. Formally, this condition is expressed as $|[A_j]_2| \geq \beta$, for all $j \in [0, \nu)$. The results obtained under this setting are reported in Table 10.

In the table, columns $|\Phi|$(%) and $|E|$(%) contain the cardinalities of the sets Φ and E, respectively, expressed as percentages relative to the cardinality of S. Column **# Groups** indicates the number of duplicate groups contained in each set. The maximum and minimum cardinalities of the duplicate groups, expressed as percentages with respect to $|S|$, are shown in columns **Max.**(%) and **Min.**(%), respectively. The mean and standard deviation of the number of VPKs per duplicate group are presented in columns **Mean** and **StdDev**. Together, these metrics characterize the numerical distribution and dispersion of the groups in G. The final column contains the value of the *exclusiveness index* ρ. The last row of the table provides the average values across the different VPK sets, offering an overall indication of the quality obtained with S-Scheme.

Due to the simplicity of S-Scheme, the number of null values is very small, no exclusive VPK values are produced, and a small number of duplicate groups is generated. Overall, the quality of the VPK sets obtained with the S-Scheme is too low to ensure reliable watermark synchronization.

After completing the experiments with the S-Scheme, we proceeded to generate S with the remaining VPK schemes, still without reserving cover for watermark synchronization. For the VT-Scheme, we conducted a series of experiments to analyze how the VPK set varies with different values of h. Recall that h denotes the length of the binary string representing each VPK and must satisfy the condition $2^h \geq |W|$. Considering the watermarking technique selected to evaluate the synchronization quality associated with the VPK sets, we used the watermark sources listed in Table 9 to determine appropriate values of h. These values appear in the first column of Table 11, which reports the results obtained for the VT-Scheme.

An additional requirement for correctly applying the dimensionality reduction step in VT-Scheme is that $\nu > h$. This condition is not always satisfied for the watermark sources under consideration, given their number of pixels. Since R contains ten attributes, only the configurations $h = 7$ and $h = 9$ satisfy this criterion.

For the FRBBM-Scheme, we assigned a ratio of 0.2 to each attribute of R, resulting in the selection of approximately two attributes per tuple. Using the same ratio for all attributes ensures that each contributes equally to VPK generation and is therefore similarly exposed to the *deletion problem*. If greater relevance must be given to a particular attribute, its selection ratio can simply be increased, as specified by the scheme.

Table 12 presents the features and metrics of the sets obtained with each scheme under the condition $|[A_j]_2| \geq \beta$, for all $j \in [0, \nu)$. For broader comparison, the table also includes the results of the set computed with the T-Scheme (which relies on the primary key),

Table 12
Features and quality of S for all schemes ($\beta = 3$, $\xi = 0$).

VPK Scheme	$\lvert\Phi\rvert$(%)	$\lvert E\rvert$(%)	Duplicate Values					ρ
			# Groups	Group's Size Range		Mean	StdDev	
				Max.(%)	Min.(%)			
T-Scheme†	0	100	0	–	–	–	–	100
S-Scheme	3.91	0	4	38.27	11.45	7206.40	3686.10	4.29×10^{-6}
E-Scheme	3.91	0	4	26.78	22.56	72064	5754.87	1.86×10^{-8}
M-Scheme	0	0	7	71.99	0.11	4285.71	7906.79	1.45×10^{-4}
Ext-Scheme	20.70	1.21	343	6.95	0.01	68.30	239.46	1.49
HQR-Scheme	0.75	58.46	2983	0.6333	0.0067	0.0137	0.0217	62.23
FRBBM-Scheme	13.82	66.62	1469	0.73	0.0067	3.9952	9.0562	68.5232
VT-Scheme $_{(h=13)}$	0	0	8192	0.0133	0.0067	3.6621	0.4733	7.60
HSR-Scheme	0	76.05	42056	0.18	2.08×10^{-4}	5.47	24.22	77.70

as well as the average values of the VPK sets produced by the S-Scheme (previously shown in the last row of Table 10). For the VT-Scheme, we report the results corresponding to the configuration with $h = 13$, which represents the best-performing setting in terms of quality.

The results in Table 12 reflect the expected collision behavior of each construction. Fixed-β bit-extraction schemes (S- and E-Schemes) map tuples into a very small set of possible identifiers when β is small (here $\beta = 3$), so collisions are inevitable and the resulting ρ becomes extremely low. The M-Scheme increases variability by combining multiple attributes, but still inherits collision-prone behavior when its underlying bit-extraction parameter remains static. In contrast, the Ext- and HQR-Schemes increase the effective identifier variability by adapting β and varying attribute participation, which increases the number of distinct VPKs and improves ρ. The FRBBM-Scheme further reduces collisions by explicitly controlling attribute participation through ratio-based matching, while the VT-Scheme prioritizes balanced bucketization of tuples (uniform VPK distribution) rather than maximizing exclusiveness, which explains its moderate ρ despite strong robustness. Finally, the HSR-Scheme achieves the largest fraction of exclusive values by generating multiple content-derived identifiers per tuple (via combinations that include attribute labels), at the cost of auxiliary structures and higher complexity.

The results shown in Table 12 correspond to implementations of the schemes that strictly enforce the restriction that only values satisfying $|[A_j]_2| \geq \beta$, for all $j \in [0, \nu)$, are involved in VPK generation. From these results, it is evident that the VPK schemes exhibit a temporal evolution, reflected in the general increase of the ρ values across the rows. Although some features, such as the proportion of null values, do not always change in the most favorable direction, others, including the number of exclusive values and the number of duplicate groups, tend to improve in the more recent proposals. This overall trend explains the higher values observed for the *exclusiveness index*.

Nevertheless, some of the results can be attributed to intrinsic differences among the schemes. For example, the best values in the table correspond to the HSR-Scheme, which produces a substantially larger number of VPKs than the other schemes. This behavior arises from its construction strategy, in which VPKs are generated through combinations of attribute values with attribute names, as well as with all the rest of the values within the same tuple. As a consequence, the HSR-Scheme yields larger sets of both exclusive and duplicate values, ultimately leading to a higher quality outcome as reflected by ρ. It is worth noting, however, that this scheme requires significantly greater computational and time resources due to its inherent complexity.

In contrast, the VT-Scheme constitutes a particular case that differs from the more recent proposals. Rather than focusing primarily on maximizing ρ, the VT-Scheme aims to involve all attributes as uniformly as possible in the generation of S, ensuring that none becomes a weak point with respect to the *deletion problem*. This design objective explains why its performance profile does not always follow the same trends observed in the other schemes.

After verifying the quality of S without reserving space for watermark synchronization, we proceeded to generate the sets again, this time ensuring that each value provides sufficient space for mark embedding. To avoid any overlap between the β *msb* used for VPK generation and the first *lsb* reserved for ξ, we excluded the least significant bit from the VPK construction. Under this setting, a value is considered valid for VPK generation only if it satisfies the condition $|[A_j]_2| \geq (\beta + \xi)$ for all $j \in [0, \nu)$; in practice, this reduces to $|[A_j]_2| \geq (\beta + 1)$. The experiments were executed using the schemes in the same order as before. The results obtained for the S-Scheme under this configuration are presented in Table 13.

The overall quality of the sets obtained with the S-Scheme, now ensuring coverage for watermark synchronization, is not substantially different from the quality achieved when the full attribute values were used for VPK generation. This reinforces the observation that the performance of this scheme is both poor and largely invariant.

We then repeated the experiments for the VT-Scheme, again enforcing the requirements for both VPK generation and mark embedding. In this case, the results remained very similar to those obtained when no space was reserved for watermark synchronization (see Table 14).

This behavior arises from the fact that the primary factor influencing the performance of the VT-Scheme is the parameter h, and that the outcome of training the underlying models does not change substantially when the least significant bit of the data is excluded.

Table 15 presents the features and quality metrics of the VPK sets obtained with each scheme when coverage for watermark synchronization is also taken into account. Overall, the variations observed across schemes are minimal and not sufficient to conclude that their performance differs significantly under this condition. The schemes exhibiting the largest variations are the HQR-Scheme

Table 13
S-Scheme: features and quality of S per attribute ($\beta = 3$, $\xi = 1$).

Attribute	$\|\Phi\|$(%)	$\|E\|$(%)	Duplicate Values					ρ
			# Groups	Group's Size Range		Mean	StdDev	
				Max.(%)	Min.(%)			
ELEVATION	0	0	4	59.34	1.23	7500	7356.34	1.04×10^{-5}
ASPECT	3.59	0	4	32.14	15.54	7230.50	2783.57	2.08×10^{-6}
SLOPE	22.26	0	4	27.66	13.34	5830.50	1841.87	2.46×10^{-6}
HOR_DIST_TO_HYDROLOGY	8.46	0	4	27.24	14.69	6865.75	1691.35	2.06×10^{-6}
VERT_DIST_TO_HYDROLOGY	30.48	0	4	22.17	11.51	5213.75	1412.09	2.72×10^{-6}
HOR_DIST_TO_ROADWAYS	0.01	0	4	30.31	16.66	7499.25	1936.35	1.89×10^{-6}
HILLSHADE_9AM	0.0033	0	4	47.87	3.86	7499.75	6220.33	4.38×10^{-6}
HILLSHADE_NOON	0	0	4	54.44	0.95	7500	7605.03	1.37×10^{-5}
HILLSHADE_3PM	0.37	0	4	43.44	15.81	7472.50	3844.48	2.09×10^{-6}
HOR_DIST_TO_FIRE_POINTS	0.0066	0	4	32.72	13.18	7499.50	2497.85	2.00×10^{-6}
AVERAGE	**6.52**	**0**	**4**	**37.73**	**10.68**	**7011.15**	**3718.93**	$\mathbf{4.38 \times 10^{-6}}$

Table 14
VT-Scheme: features and quality of S for different values of h ($\xi = 1$).

VPK length	$\|\Phi\|$(%)	$\|E\|$(%)	Duplicate Values					ρ
			# Groups	Group's Size Range		Mean	StdDev	
				Max.(%)	Min.(%)			
E $\rightarrow h = 7$	0	0	128	0.7833	0.78	234.375	0.4860	1.82×10^{-3}
Dào $\rightarrow h = 9$	0	0	512	0.1967	0.1933	58.59	0.4916	2.91×10^{-2}
WWF $\rightarrow h = 11$	0	0	2048	0.0533	0.0467	14.6484	0.4786	4.67×10^{-1}
UTM $\rightarrow h = 13$	0	0	8192	0.0133	0.01	3.6621	0.4730	7.60

Table 15
Features and quality of S for all schemes ($\beta = 3$, $\xi = 1$).

VPK Scheme	$\|\Phi\|$(%)	$\|E\|$(%)	Duplicate Values					ρ
			# Groups	Group's Size Range		Mean	StdDev	
				Max.(%)	Min.(%)			
T-Scheme[†]	0	100	0	–	–	–	–	100
S-Scheme	6.52	0	4	37.73	10.68	7011.15	3718.93	4.38×10^{-6}
E-Scheme	6.52	0	4	26.19	21.79	70111.5	5954.09	1.91×10^{-8}
M-Scheme	0	0	8	69.30	0.0066	3750	7175.71	1.80×10^{-3}
Ext-Scheme	20.70	1.18	344	6.95	0.0067	68.13	239.12	1.46
HQR-Scheme	0.73	43.38	3491	0.95	0.0067	4.80	9.0497	47.65
FRBBM-Scheme	13.48	62.81	1767	0.48	0.0067	4.0249	6.4263	65.0669
VT-Scheme $_{(h=13)}$	0	0	8192	0.0133	0.01	3.6621	0.4730	7.60
HSR-Scheme	0	76.05	42056	0.18	2.08×10^{-4}	5.47	24.22	77.70

and the FRBBM-Scheme, primarily in terms of the value of ρ. Nevertheless, their relative ranking among the VPK schemes remains unchanged. It is also worth noting that, in the case of the HSR-Scheme, the reported values remain constant. This is expected, since watermark embedding in that scheme relies on an external structure rather than reserving cover within the values of R.

After evaluating the quality of S generated with each scheme, we observe that their evolution over time is coherent, even in those cases where the value of ρ does not increase. These cases correspond to schemes whose design objectives emphasize resilience to the *deletion problem*, rather than maximizing ρ. In the following section, we assess how the schemes perform when watermark synchronization is challenged by the deletion of arbitrary attributes from R.

5.2.3. Resilience to the deletion problem

To properly assess how each VPK scheme is affected by the *deletion problem*, we first determine which schemes are capable of producing VPK sets of sufficiently high quality to enable reliable watermark synchronization. Using the watermark sources, watermarking technique, and evaluation metrics introduced in the experimental setup, we carry out this preliminary step before analyzing the impact of attribute deletions. More specifically, for each VPK scheme, we generate the corresponding set S, embed a watermark constructed from the sources listed in Table 9 using the technique proposed by Sardroudi & Ibrahim [11], and subsequently perform watermark extraction. Finally, by comparing the embedded and extracted watermarks, we compute the CF and SSIM values, which quantify the quality of the synchronized watermark associated with each VPK set.

The first results correspond to the sets generated with the S-Scheme (see Table 16). To maintain consistency with the earlier evaluation of this scheme, we use each of the ten VPK sets obtained from the individual attributes of R. For each set, the table reports its corresponding value of ρ, along with the CF and SSIM values computed for the synchronized watermarks. These two metrics are

Table 16
Quality of the synchronized watermark with S generated using the S-Scheme ($\beta = 3$ and $\xi = 1$).

Attribute	ρ	UTM		WWF		Dào		E	
		CF	SSIM	CF	SSIM	CF	SSIM	CF	SSIM
ELEVATION	1.04×10^{-5}	0.06	0.06	0.22	0.02	0.95	0	4	0.1
ASPECT	2.08×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
SLOPE	2.46×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
HOR_DIST_TO_HYDROLOGY	2.06×10^{-6}	0.06	0.06	0.22	0.02	0.95	0	4	0.1
VERT_DIST_TO_HYDROLOGY	2.72×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
HOR_DIST_TO_ROADWAYS	1.89×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
HILLSHADE_9AM	4.38×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
HILLSHADE_NOON	1.37×10^{-5}	0.06	0.06	0.22	0.02	0.95	0	4	0.1
HILLSHADE_3PM	2.09×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
HOR_DIST_TO_FIRE_POINTS	2.00×10^{-6}	0.07	0.05	0.27	0.02	1.19	0	4	0.1
AVERAGE	$\mathbf{4.38 \times 10^{-6}}$	**0.067**	**0.053**	**0.255**	**0.02**	**1.118**	**0**	**4**	**0.1**

Table 17
Quality of the synchronized watermark with S generated using VT-Scheme considering different VPK lengths according to h ($\xi = 1$).

VPK Scheme	ρ	UTM			WWF			Dào			E		
		CF	SSIM	W	CF	SSIM	W	CF	SSIM	W	CF	SSIM	W
E $\rightarrow h = 7$	1.82×10^{-3}	1.95	0.04		6.94	0.08		26.42	0.26		73.00	0.69	
Dào $\rightarrow h = 9$	2.91×10^{-2}	7.50	0.08		24.50	0.26		70.47	0.70		100	1	
WWF $\rightarrow h = 11$	4.67×10^{-1}	26.58	0.25		67.44	0.63		98.80	0.99		100	1	
UTM $\rightarrow h = 13$	7.60	70.62	0.57		98.5	0.94		100	1		100	1	

shown grouped under the columns associated with the watermark sources UTM, WWF, Dào, and E. As expected from the low values of ρ, the quality of the synchronized watermarks is generally poor across all cases.

After evaluating the quality of the watermarks synchronized using the sets obtained with the S-Scheme, we proceed to assess watermark synchronization using the sets produced by the VT-Scheme. The corresponding results are presented in Table 17. Since the VPK sets generated by this scheme exhibit higher quality, we additionally include an image of the synchronized watermark for each group column, organized by watermark source. As a reminder, red pixels in these images indicate cases where pixels were ignored or lost during the synchronization process.

From these results, we observe that when using the VPK sets generated with the VT-Scheme, it is possible to synchronize a recognizable watermark for configurations with $h = 11$ or $h = 13$. Moreover, watermark sources composed of fewer pixels tend to be more advantageous, as they allow successful synchronization even when the corresponding VPK sets exhibit relatively low quality. This behavior is particularly evident in the case of the watermark synchronized using source E with $h = 9$.

We present all the results of the synchronized watermarks obtained with the sets generated using each VPK scheme in Table 18. For the S-Scheme, we report only the best-performing result, which corresponds to the set generated using the attribute HILLSHADE_NOON. For the VT-Scheme, we likewise include the best configuration, namely the set produced with $h = 13$. In addition, the table incorporates the column **max(h)**, which indicates the maximum length of the binary string representing the VPK for each scheme. This metric plays a key role in the quality of the synchronized watermark.

An interesting case in Table 18 is the HSR-Scheme. Notice that the synchronization of meaningful watermarks using the watermark sources and the previously mentioned watermarking technique is an adaptation of this scheme for comparing the results obtained with it against the other VPK schemes. This is not how this scheme is meant to be used for watermark synchronization, as explained in Section 4.2.9.

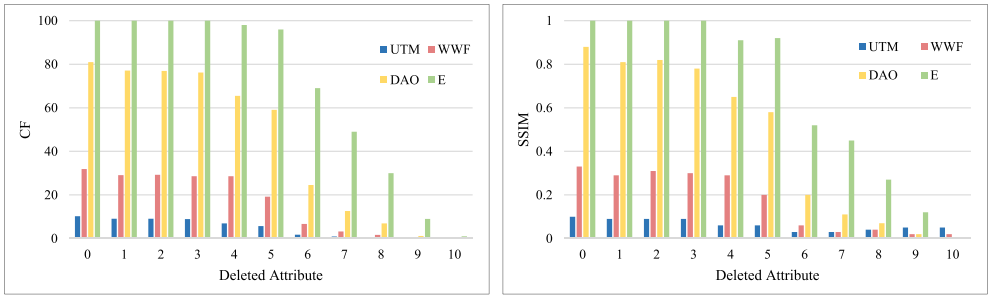
Although the HSR-Scheme yields the largest VPK length, it also generates a substantially greater number of VPKs. As a result, the abundance of keys is sufficient to ensure a high-quality set S, which in turn enables successful synchronization even for the largest watermark sources. Notably, for the HSR-Scheme, the watermark UTM is fully synchronized, a result that is not achievable with the T-Scheme, owing to its considerably smaller number of keys and the effects of the *pseudo-random* pixel selection mechanism. Consequently, when considering the HSR-Scheme, it is important to account for its performance requirements, as the scheme demands significantly more computational resources and processing time.

After evaluating the quality of the synchronized watermarks obtained with the VPK sets generated by each scheme, we proceed to assess their performance under the *deletion problem*. We do not include schemes that fail to guarantee watermark synchronization, since the watermark is already compromised in those cases. Therefore, only the Ext-Scheme, HQR-Scheme, FRBBM-Scheme, and VT-Scheme are selected for this analysis.

The HSR-Scheme is also excluded, not only because it consistently achieves high synchronization quality, but also due to its particular characteristics: it relies on both physical and logical embedding, which prevents the watermark from being compromised if any single attribute is deleted.

Table 18
Quality of the synchronized watermark ($\xi = 1$).

VPK Scheme	max(h)	ρ	UTM			WWF			Dào			E		
			CF	SSIM	W	CF	SSIM	W	CF	SSIM	W	CF	SSIM	W
T-Scheme	10	100	98.38	0.89		100	1		100	1		100	1	
S-Scheme	10	1.37×10^{-5}	0.06	0.06		0.22	0.02		0.95	0		4	0.1	
E-Scheme	10	1.91×10^{-8}	0.03	0.06		0.11	0.02		0.47	0.01		2	0.1	
M-Scheme	19	1.80×10^{-3}	0.12	0.06		0.44	0.02		1.9	0.03		8	0.12	
Ext-Scheme	11	1.47	10.18	0.1		31.88	0.33		80.95	0.88		100	1	
HQR-Scheme	20	47.65	91.57	0.73		100	1		100	1		100	1	
FRBBM-Scheme	128	65.07	94.86	0.76		100	1		100	1		100	1	
VT-Scheme	13	7.60	70.62	0.57		98.5	0.94		100	1		100	1	
HSR-Scheme	256	77.70	100	1		100	1		100	1		100	1	



(a) (b)

Fig. 10. Quality of the synchronized watermark obtained with the VPK set S generated by the Ext-Scheme, after simulating the *deletion problem* through the removal of each attribute of R (one at a time). (a) CF values. (b) SSIM values.

These benefits, however, stem from additional features that may be viewed as limitations, such as the need for auxiliary structures and the substantial computational resources required. Consequently, users of the HSR-Scheme must carefully consider its objectives, operational requirements, and overall functioning before deployment.

To evaluate the resilience of the selected VPK schemes to the *deletion problem*, we embed the watermark using the VPK set generated by each scheme and then simulate the deletion of one of the ten attributes of R. After deleting an attribute, watermark detection is performed to compare the quality of the extracted signal with that of the embedded watermark. The deletion is simulated by assigning null values to the column corresponding to the removed attribute, forcing the VPK scheme to disregard it during the synchronization.[4] For each scheme, we conduct ten experiments, each simulating the deletion of a different attribute of R.

The first round of experiments corresponds to the Ext-Scheme. Fig. 10 presents the results obtained for each attribute deletion. The numbers identifying the deleted attributes match those used in Table 8. The first group of bars represents the quality of the synchronized watermark when no attribute is deleted, which is why it is grouped with the value 0 for the deleted attribute axis. This behavior in Fig. 10 is consistent with the Ext-Scheme's selection mechanism: because attribute participation depends on value-dependent most significant bit criteria, some attributes become disproportionately influential in VPK formation; deleting a highly influential attribute causes many tuples to be re-identified, producing abrupt synchronization loss, whereas deleting a weakly involved attribute has a limited effect.

As expected, the most resilient signal corresponds to the watermark generated from the image E. However, due to the unequal involvement of the attributes in the VPK construction, deleting attributes 6 to 10 leads to a more pronounced degradation in watermark recognition. This effect is particularly evident for attributes 9 and 10, which clearly represent critical cases. Overall, this behavior highlights the highly inconsistent resilience of the Ext-Scheme to the *deletion problem*: while it performs well in some scenarios, it severely compromises watermark synchronization in others.

For the second round of experiments, we tested the performance of the HQR-Scheme. Fig. 11 presents the results obtained for this scheme. In this case, higher resilience is observed more consistently across the ten scenarios simulating attribute deletion. The

[4] In practice, data owners may also consider alternative strategies to mitigate the impact of attribute deletion. For example, the deleted column could be filled with synthetic values that approximate the statistical distribution of the original attribute, potentially improving the likelihood of successful watermark detection.

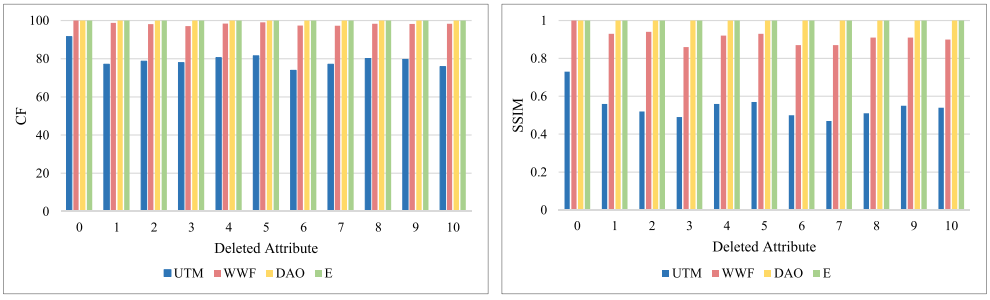


Fig. 11. Quality of the synchronized watermark obtained with the VPK set S generated by the HQR-Scheme, after simulating the *deletion problem* through the removal of each attribute of R (one at a time). (a) CF values. (b) SSIM values.

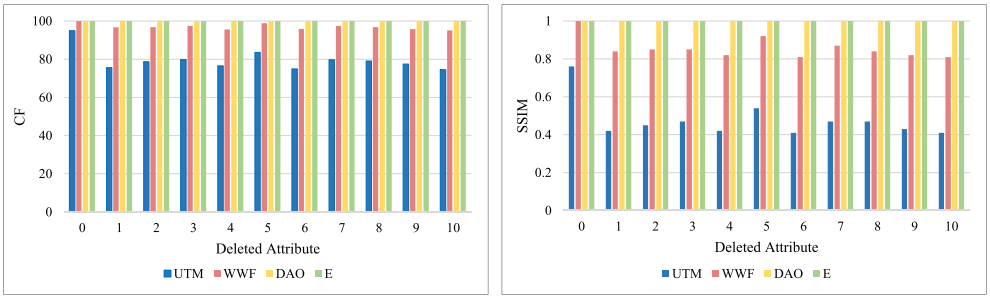


Fig. 12. Quality of the synchronized watermark obtained with the VPK set S generated by the FRBBM-Scheme, after simulating the *deletion problem* through the removal of each attribute of R (one at a time). (a) CF values. (b) SSIM values.

influence of the watermark source length is also more evident here. Notice that for this scheme, its resilience depends heavily on how the computation of the average values of A_{VV} is performed. Relying on such operational values may weaken the scheme. Therefore, whenever a column is identified as missing, it must be excluded from these computations. The improved stability follows from the HQR-Scheme's cyclic traversal and randomized initialization, which distribute VPK formation across multiple attributes and reduce reliance on any single column. However, components that aggregate across attributes (e.g., averages) can still act as global dependencies and should be computed excluding missing columns to avoid unnecessary sensitivity.

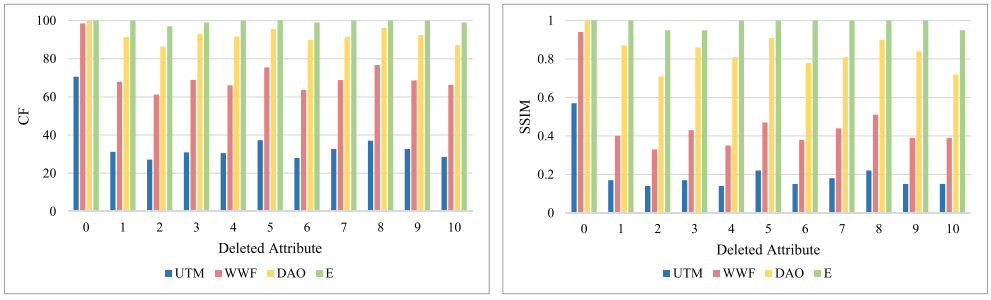
The third round of experiments evaluating resilience to the *deletion problem* was performed using the FRBBM-Scheme. This experiment was carried out using the same configuration as before, assigning each attribute a selection ratio of 0.2. Under this setting, a uniform behavior is observed when each attribute of R is deleted (see Fig. 12). When alternative configurations with non-uniform selection ratios are used, results similar to those obtained with the Ext-Scheme are expected. A notable advantage of the FRBBM-Scheme is that it allows the data owner to tailor the computation of S to the needs of the application, taking into account the strengths and potential drawbacks associated with the chosen configuration. The FRBBM-Scheme's robustness stems from ratio-controlled participation: each attribute contributes according to a configured probability pattern, so deleting one attribute perturbs only the subset of tuples in which that attribute would have matched, yielding more predictable and uniform degradation.

The final experiment is conducted using the VT-Scheme. Given the comparatively lower quality of the VPK sets produced by this scheme relative to the more recent proposals, a lower degree of watermark synchronization is expected. Nonetheless, because its design objective emphasizes the uniform involvement of all attributes in the construction of VPKs, the VT-Scheme is anticipated to exhibit strong resilience to the *deletion problem*. This behavior is precisely reflected in Fig. 13. Although the CF and SSIM values are generally lower than those obtained for most of the previously evaluated schemes (with the notable exception of the Ext-Scheme), the synchronization quality remains remarkably stable regardless of which attribute is deleted. The VT-Scheme's consistent performance is explained by its representation-learning pipeline: the latent encoding and median-tree binarization aim to produce balanced codes and distribute information across dimensions, so the impact of removing a single attribute is spread rather than concentrated, improving consistency even when absolute synchronization quality is not maximal.

Fig. 14 summarizes the performance of each scheme when synchronizing the highest-capacity watermark, UTM. This visualization clearly highlights the differences among the synchronized signals produced by the various VPK sets.

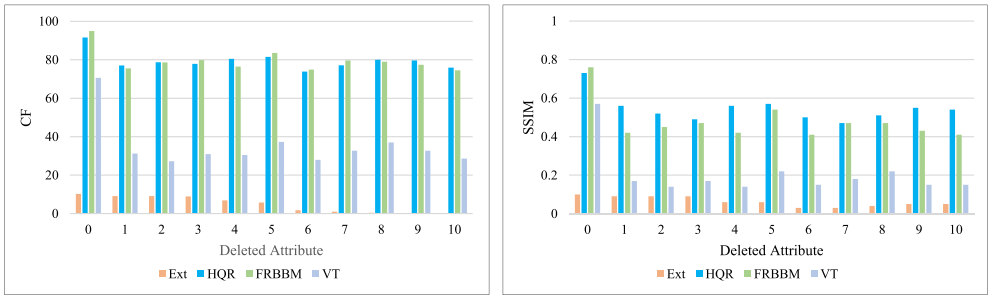
As previously discussed, the Ext-Scheme exhibits the weakest performance: both CF and SSIM remain consistently low, and the scheme fails to preserve synchronization for several attribute deletions, reflecting its limited robustness.

In contrast, the HQR- and FRBBM-Schemes achieve substantially higher synchronization quality across all deletion scenarios, with CF values typically above 70 and SSIM values remaining comparatively stable. Notably, the FRBBM-Scheme offers the additional advantage of allowing the data owner to adjust its resilience through the choice of selection ratios.



(a) (b)

Fig. 13. Quality of the synchronized watermark obtained with the VPK set S generated by the VT-Scheme, after simulating the *deletion problem* through the removal of each attribute of R (one at a time). (a) CF values. (b) SSIM values.



(a) (b)

Fig. 14. Comparison under the *deletion problem* (UTM watermark): synchronization quality after single-attribute deletion for the evaluated schemes. (a) CF values. (b) SSIM values.

The HQR-Scheme, however, shows a mild dependence on the underlying distribution of attribute values, as evidenced by the slight oscillations across deleted attributes.

Finally, the VT-Scheme presents a distinctive pattern: although its absolute CF and SSIM values are generally lower than those obtained by the HQR- and FRBBM-Schemes, the behavior remains remarkably stable across the ten deletion cases. This confirms that the uniform involvement of attributes in the generation of VPKs yields strong consistency, even when the overall quality of the synchronized watermark is lower.

To summarize deletion-resilience beyond individual curves, we interpret each scheme's performance across all attribute-deletion cases using two complementary statistics: (i) the average value of the metric (e.g., CF/SSIM) across deleted attributes, which captures expected synchronization quality under attribute loss, and (ii) the variability across deleted attributes (e.g., standard deviation or range), which captures sensitivity to which attribute is removed. In this view, schemes with similar average performance can differ substantially in robustness if their variability is high, indicating that synchronization depends strongly on a small subset of attributes; conversely, low variability indicates a more uniform attribute involvement and a more predictable degradation pattern.

The experiments conducted reveal substantial differences in how VPK schemes withstand the *deletion problem*. While some approaches, such as the Ext-Scheme, display inconsistent behavior and fail to preserve watermark synchronization under certain deletions, others (most notably the HQR- and FRBBM-Schemes) provide robust protection, maintaining stable synchronization quality across all scenarios.

The VT-Scheme, although yielding lower absolute synchronization quality, stands out for its high consistency, reflecting the advantages of uniformly distributing attribute contributions during VPK construction. These findings indicate that resilience must be assessed not only through numerical performance but also in relation to each scheme's underlying design choices.

Motivated by these observations, the next section offers a qualitative comparison that examines the broader strengths, limitations, and trade-offs associated with each VPK proposal.

5.3. Qualitative comparison

In addition to the quantitative analysis presented in the previous subsections, a qualitative examination of the main VPK schemes provides a broader perspective on their conceptual design, operational characteristics, and practical suitability for watermark synchronization. While numerical metrics quantify specific aspects of performance, they do not capture fundamental distinctions in scheme architecture, adaptability, reliance on auxiliary mechanisms, or the trade-offs introduced by different design philosophies. To complement the quantitative findings, Table 19 summarizes these qualitative properties and highlights the principal trends shaping

Table 19
Summary of qualitative comparison of VPK Schemes.

VPK Scheme	Generation Principle	PK Dependence	Attributes Used	Duplicate Keys	Deletion Resilience	Auxiliary Structures/ Mechanisms	Computational Complexity	Main Advantages	Main Limitations
T-Scheme (Tuple-based)	Based on AHK algorithm using the tuple's PK	High – relies on PK	PK attributes	None (inherits PK uniqueness)	Low – fails if PK deleted/replaced.	None	Low	Simple, foundational baseline	Not PK-free, fails if PK missing
S-Scheme (Simple)	Uses most significant bits (β) of attributes. It replaces PK in AHK.	None	1 (numeric attribute)	Very High – constant β causes many duplicates	Low – highly affected by attribute deletion.	None	Low	PK-free, simple	Massive duplication, poor resilience
E-Scheme (Element-based)	Computes one VPK per attribute. Uses two hash functions for selection and embedding.	None	All ($\eta \times \nu$ values)	Very High – β constant across all tuples.	Very Low – vulnerable to deletion of any attribute.	None	Medium	Comprehensive use of relation data; generalized S-Scheme.	Overly redundant; many duplicates; fragile to deletions.
M-Scheme (Multiple-elements)	Combines hashes of two or more attribute values per tuple (lowest hash values).	None	≥ 2 attributes	High – β constant.	Medium – improved by multi-attribute usage.	None	Medium	Simple multi-attribute scheme; improved over S/E.	Still high duplication; static β limits adaptability.
Ext-Scheme (Extended)	Adaptive β selection based on *msb* analysis. Dynamic attribute participation using XOR rule.	None	Variable number of attributes	Lower than M-Scheme.	Moderate–Varies; experiences resilience for some attributes, but others may compromise synchronization.	None	Medium	Produces more exclusive VPKs; more resilient to deletion.	May waste tuples; mark embedding density reduced.
HQR-Scheme (High Quality & Resilient)	Cyclic attribute order. Dynamic start, direction, and number of attributes. Uses AMv, AAv.	None	Variable (up to ℓ per tuple).	Moderate–High – still duplicates but balanced.	High – resistant to deletion and reordering.	None	Medium – High.	High resilience, high-quality VPKs, cyclic attribute mapping.	Generates many duplicates; increased complexity.
FRBBM-Scheme (Flexible Ratio Binary Matching)	Attribute selection ratio encoded in binary. Attributes included by bitwise matching.	None	Multi-attribute, ratio-controlled.	Low – flexible matching reduces duplicates.	High – resilient by dynamic attribute ratio.	None	High – bitwise matching and ratio management.	Flexible attribute participation; tunable quality vs resilience.	Complex tuning; sensitive to parameter setup.
VT-Scheme (VAE + Median Tree)	Dimensionality reduction via VAE $\rightarrow$ median tree binary encoding.	None	Multi-attribute (compressed latent space).	Moderate – uniform distribution limits duplicates but reduces exclusivity.	Very High – resilient to deletions, robust VPK distribution.	Neural model (VAE) + median tree encoder.	Very High – training and encoding phases.	Balanced, uniform VPKs; strong resilience and generalization.	Requires training data; computationally expensive.
HSR-Scheme (High-Synchronization & Robust)	Hybrid PK + VPK embedding. Uses attribute–name concatenation.	Partial – PK used for physical embedding.	Multi-attribute; VPKs per attribute combination.	Low – many VPKs per tuple, high uniqueness.	High – dual embedding improves resilience.	Auxiliary record table ($\tilde{P}$) for logical embedding.	High	Combines PK/VPK embedding; redundancy and robustness.	Depends on $\tilde{P}$ table integrity; complex implementation.

Table 20
Computational overhead summary (dominant cost drivers and approximate complexity).

Family/Scheme	Dominant operations	Approx. complexity (VPK generation)
Early bit-extraction (S-)	Extract β *msbs* from one attribute per tuple	$O(\eta)$
Element-level (E-)	Per cell: *msb* extraction + hashing/selection	$O(\eta\nu)$
Multi-attribute hash (M-)	Hash multiple attribute-derived values per tuple; select few	$O(\eta\nu)$ (selection over ν)
Adaptive (Ext-, HQR-)	Per tuple: scan attributes; adaptive β; hashing; cyclic traversal (HQR)	$O(\eta\nu)$ (plus scheme-specific constant factors, higher for HQR)
Ratio-controlled (FRBBM)	Per tuple: compute per-attribute scores; bit matching + splicing	$O(\eta\nu)$
Learning-based (VT)	Train VAE + encode tuples; median-tree routing	Training: $\approx O(\text{epochs} \cdot \eta \cdot \text{cost(VAE)})$; Generation: $\approx O(\eta \cdot \text{cost(VAE)} + \eta h)$
Hybrid/auxiliary (HSR)	Multi-VPK generation from attribute subsets + auxiliary table updates	Worst-case: $O\left(\eta \cdot 2^{\lvert A\rvert}\right)$; practical: $O\left(\eta \cdot \sum_{i=1}^{k} \binom{\lvert A\rvert}{i}\right)$ with bounded k

Table 21
Decision-support summary mapping VPK schemes to operational contexts.

Operational context	Recommended scheme families (examples)	Rationale and practical considerations
Low attribute-deletion risk, limited computational overhead	Early bit-extraction schemes (S-, E-, M-Schemes)	Lightweight and simple to deploy, but prone to high duplication and low deletion resilience. Suitable only when attribute modification or removal is unlikely and computational simplicity is a priority.
Moderate deletion risk, no auxiliary protected structures available	Adaptive multi-attribute schemes (Ext-, HQR-Schemes)	Improve VPK quality and deletion resilience compared to early schemes without relying on external structures. Trade-offs include increased computational overhead and, in some cases, uneven attribute involvement.
High deletion risk, attribute participation can be tuned	Ratio-controlled multi-attribute schemes (FRBBM-Scheme)	Provide stronger deletion resilience and reduced duplication by adjusting attribute selection ratios. Requires careful parameter tuning to balance robustness and synchronization quality.
High deletion risk, maximum robustness desired, higher computational overhead acceptable	Learning-based schemes (VT-Scheme)	Exploit latent representations and median-tree encoding to distribute identifying information across attributes, yielding balanced VPKs and strong robustness to attribute deletion. Incur higher computational cost due to model training and inference.
Availability of protected auxiliary structures and maximum synchronization robustness required	Hybrid and auxiliary-structure schemes (HSR-Scheme)	Leverage redundancy and external record tables to achieve high robustness and synchronization accuracy. Depend on the integrity and availability of auxiliary structures and introduce additional system complexity.
Mixed environments where PK may or may not be preserved	Hybrid approaches (HSR-Scheme) or robust PK-free schemes (HQR-, FRBBM-Schemes)	HSR can exploit PKs when available while maintaining VPK-based synchronization; otherwise, robust PK-free schemes offer practical alternatives in PK-unavailable settings.

the evolution of VPK approaches. Building on this synthesis, Table 21 provides a compact decision-support view that maps scheme families to common operational constraints.

The qualitative comparison highlights several important trends in the evolution of VPK generation. Early schemes such as the T-, S-, E-, and M-Schemes introduced foundational ideas for PK-free synchronization but suffered from significant drawbacks, most notably the high incidence of duplicate VPK values and limited resilience against attribute deletions. As Table 19 indicates, the S- and E-Schemes, which rely on fixed most-significant-bit extraction, generate substantial redundancy due to the use of constant β-values, while the M-Scheme, despite incorporating multiple attributes, still inherits this limitation. These early approaches therefore provide low-quality synchronization in scenarios involving benign updates or intentional deletion attacks, underscoring the need for more adaptive VPK formation mechanisms.

While Table 19 qualitatively ranks computational complexity, practitioners often need an approximate sense of scaling. We therefore summarize the dominant cost drivers and asymptotic complexity for each family (see Table 20). For most classical VPK constructions, the cost is linear in the number of processed tuples and attributes, with additional constant-factor overhead from hashing and bit parsing. Learning-based VT introduces a distinct two-phase cost profile: offline model training followed by linear-time encoding at detection. HSR-style hybrid approaches may incur combinatorial costs if all attribute-subset combinations are generated, so practical deployments typically restrict the candidate subset size.

Subsequent developments, including the Ext-Scheme and HQR-Scheme, introduced attribute-adaptive processes specifically designed to reduce VPK duplication and enhance robustness. As summarized in Table 19, the Ext-Scheme's dynamic selection of β and XOR-based attribute participation increases the exclusivity of VPKs, providing stronger resistance to deletion and tuple-reordering attacks. The HQR-Scheme further advances this direction by employing cyclic attribute traversal with randomized initialization

parameters, thereby increasing the entropy and variability of VPK assignments across tuples. Although both schemes substantially improve quality, their increased computational cost and the persistent, though reduced, presence of duplicates illustrate the trade-off between flexibility and complexity inherent in adaptive VPK construction.

Table 21 complements Table 19 by translating the comparative properties into deployment-oriented recommendations for common PK-unavailable data-handling scenarios.

More recent proposals, such as the FRBBM-Scheme, VT-Scheme, and HSR-Scheme, reflect a shift toward hybrid and learning-based VPK generation strategies.

As shown in Table 19, the FRBBM-Scheme achieves strong deletion resilience and low duplication through ratio-controlled binary matching, but introduces a more complex parameter space requiring careful tuning.

The VT-Scheme represents a more radical departure from traditional bit-level operations, employing a variational autoencoder and a median-tree encoder to produce well-balanced VPK distributions with uniform attribute involvement. This methodological shift results in particularly high deletion resilience, as confirmed by the quantitative evaluation, although at the cost of considerable computational overhead and reliance on model training.

Similarly, the HSR-Scheme combines PK-based physical embedding and VPK-based logical embedding within a hybrid synchronization framework, resulting in high robustness through redundant embedding while requiring auxiliary structures and more elaborate implementation workflows.

Viewed collectively through the qualitative lens provided in Table 19, the evolution of VPK schemes reveals a clear progression from simple bit-extraction methods toward increasingly adaptive, multi-attribute, and learning-driven approaches. Each innovation addresses specific weaknesses of its predecessors, particularly regarding duplication, deletion resilience, and synchronization reliability. At the same time, this progression underscores persistent trade-offs: enhancements in robustness and uniqueness typically entail higher computational complexity, reliance on auxiliary structures, or greater sensitivity to parameter configuration. The qualitative evaluation thus highlights the importance of balancing resilience, efficiency, and deployability when selecting VPK mechanisms for practical watermarking scenarios.

6. Opportunities

The comparative analysis of existing VPK schemes presented in this survey identifies several opportunities to advance the design, implementation, and practical deployment of VPK-based synchronization techniques for relational data watermarking. Although existing proposals address many challenges (e.g., PK independence, multi-attribute selection, adaptive construction, and robustness to common modifications), important gaps remain, thereby opening several promising research directions.

A first opportunity concerns lightweight VPK generation strategies that reduce duplication without relying on computationally intensive learning models. As shown in the qualitative comparison, fixed bit-extraction schemes (e.g., S-, E-, and M-Schemes) exhibit a high collision rate of duplicate VPK values, whereas learning-based approaches, such as the VT-Scheme, improve deletion resilience and distributional balance at the cost of additional computational overhead for neural-model training and inference. Bridging this gap, achieving low-collision VPKs with modest computational demands, remains an open challenge.

A second opportunity lies in data-adaptive VPK generation. Most existing schemes employ static attribute sets or predetermined selection mechanisms that do not react to evolving database characteristics or schema changes. Future schemes may benefit from automatically adjusting attribute participation, bit-depth selection, or hashing combinations based on data entropy, attribute type, or value distributions. Such adaptivity would improve synchronization reliability in dynamic data environments.

A third direction concerns incorporating attribute semantics into the VPK construction. Current schemes treat attributes primarily as undifferentiated numeric values or latent features, overlooking semantic context such as functional dependencies, field volatility, and categorical structure. Incorporating semantic information may yield VPKs that are more stable, more exclusive, and more resilient to realistic update patterns, thereby reducing the sensitivity to benign modifications.

Another opportunity involves hybrid PK–VPK watermarking without reliance on external metadata structures. While approaches such as the HSR-Scheme demonstrate the robustness benefits of hybrid synchronization, their dependence on auxiliary record tables reduces deployability. Future work may explore hybrid designs in which synchronization anchors are derived directly from intrinsic tuple properties or the relational structure, eliminating the need for auxiliary data storage.

There is also substantial room for advancing VPK schemes capable of withstanding realistic database attack models. Existing evaluations typically focus on tuple deletion, reordering, or simple updates. However, in practice, released relational datasets often undergo batch deletions, schema evolution, type transformations, and block-level operations. VPK schemes explicitly designed to withstand such conditions would significantly broaden the applicability of relational watermarking in modern data management scenarios.

A further opportunity concerns the integration of reversible or low-distortion watermarking principles into VPK generation. Although most VPK schemes tolerate minor modifications to attribute values, several application domains, such as medical, financial, or regulatory data, require exact restoration of the original content after watermark extraction. Therefore, techniques inspired by reversible watermarking represent a promising direction for designing VPK mechanisms that support distortion-free embedding while maintaining reliable synchronization capabilities.

Finally, the increasing complexity of modern schemes suggests a need for more transparent and verifiable VPK generation processes. Learning-based or highly adaptive methods may obscure the rationale behind individual VPK assignments, complicating forensic analysis or systematic verification processes. Future designs may therefore incorporate explainable mechanisms, deterministic derivation trails, or cryptographic commitments that enhance trust and reproducibility. A broader opportunity lies in developing

a unified, parameterized VPK framework that encapsulates attribute selection, adaptive behavior, collision-avoidance strategies, and resilience mechanisms within a single configurable structure. Such a meta-scheme would unify classical and contemporary approaches while enabling practitioners to tailor VPK behavior to specific applications and data characteristics.

Overall, these opportunities highlight the considerable potential that remains for advancing VPK-based synchronization. By addressing persistent weaknesses, including duplication, limited adaptivity, lack of semantic awareness, practical constraints, and insufficient verification mechanisms, future schemes may achieve the combined goals of robustness and practical deployability that current methods only partially satisfy.

7. Conclusions

This survey examined VPK schemes for relational data watermarking and synthesized more than two decades of work into a unified comparative perspective. By analyzing conceptual foundations, algorithmic designs, and operational behavior, we clarified the strengths, limitations, and trajectories that have shaped PK-free watermark synchronization. The qualitative and quantitative evaluations presented in this work reveal a consistent pattern: early schemes laid the foundation for VPK-based watermarking but suffered from substantial duplication and limited robustness, whereas more recent proposals increasingly incorporate attribute adaptivity, multi-attribute strategies, and learning-based representations to improve synchronization reliability.

The comparative evaluation yields three practical takeaways. (i) VPK-set quality is mainly determined by the duplication rate (how many distinct VPKs are produced) and recomputability under attribute loss/modification (deletion resilience). (ii) Schemes based on fixed, limited bit/attribute subsets are lightweight but more vulnerable to duplication and synchronization failure under attribute deletion. (iii) Adaptive and multi-attribute designs reduce duplication and improve robustness, but often introduce higher computational costs and may require auxiliary structures or model training.

Although these advances mark significant progress, several challenges remain. High duplication rates, sensitivity to attribute deletions, constrained attribute-selection strategies, and a lack of semantic or distributional awareness continue to affect many approaches. Recent methods, such as the FRBBM-, VT-, and HSR-Schemes, introduce mechanisms that enhance VPK distinctiveness and deletion resilience. Yet, these improvements often require higher computational complexity, reliance on auxiliary structures, or careful parameter configuration. At present, no single scheme offers a uniformly strong balance among robustness, efficiency, interpretability, and practical deployability.

A practical implication of our findings is that scheme selection should reflect expected data-release and preprocessing conditions. It is reasonable to use lightweight schemes only when the schema is stable and attribute deletion is unlikely. On the other hand, adaptive and multi-attribute schemes are preferred when PK removal and attribute loss/modification are expected. Hybrid designs are advised when maximum VPK quality is required, provided any auxiliary structures remain protected and available to the legitimate detector.

Future progress will likely depend on (i) lightweight VPK designs that reduce duplication without sacrificing recomputability, (ii) adaptive and semantics-aware attribute involvement, and (iii) hybrid synchronization strategies that avoid reliance on external metadata while remaining interpretable. Equally important are clearer threat models that separate benign preprocessing from adversarial manipulation, and evaluation under realistic data-management conditions (schema evolution, ETL transformations, partial releases). Establishing reproducible benchmarks and shared baselines would enable fair cross-scheme comparison and accelerate the development of VPK mechanisms that are robust, efficient, and practical.

VPK schemes remain a central component of relational watermarking, particularly in scenarios where PK-free synchronization and resilience against data modification are essential. By consolidating the existing landscape and outlining clear opportunities for future development, this survey aims to support the design of the next generation of VPK approaches, mechanisms that are theoretically grounded, operationally efficient, and well suited to the requirements of modern data management environments.

CRediT authorship contribution statement

Maikel Lázaro Pérez Gort: Writing – original draft, Validation, Software, Investigation. **Agostino Cortesi:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially funded by the European Union – NextGenerationEU, in the framework of the iNEST – Interconnected Nord-Est Innovation Ecosystem (iNEST ECS_00000043 – CUP H43C22000540006), SERICS (PE00000014 – CUP H73C2200089001), and PADS4Health (PRIN PNRR 2022 P2022MSMAW – CUP N. H53D23010880001). The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union, nor can the European Union be held responsible for them.

Data availability

Data will be made available on request.

References

[1] M.L. Pérez Gort, C. Feregrino-Uribe, A. Cortesi, F. Fernández-Peña, A double fragmentation approach for improving virtual primary key-based watermark synchronization, IEEE Access 8 (2020) 61504–61516, https://doi.org/10.1109/ACCESS.2020.2979659

[2] S.A. Nawaz, J. Li, U.A. Bhatti, M.U. Shoukat, D. Li, M.A. Raza, Hybrid watermarking algorithm for medical images based on digital transformation and mobilenetv2, Inf. Sci. 653 (2024) 119810, https://doi.org/10.1016/j.ins.2023.119810

[3] J.-Z. Zou, M.-X. Chen, L.-H. Gong, Invisible and robust watermarking model based on hierarchical residual fusion multi-scale convolution, Neurocomputing 614 (2025) 128834, https://doi.org/10.1016/j.neucom.2024.128834

[4] Z. Liu, Y. Guo, L. Wei, Robust watermarking for diffusion model generated images, Inf. Sci. (2025) 122686, https://doi.org/10.1016/j.ins.2025.122686

[5] H. Guo, Y. Li, A. Liu, S. Jajodia, A fragile watermarking scheme for detecting malicious modifications of database relations, Inf. Sci. 176 (10) (2006) 1350–1378, https://doi.org/10.1016/j.ins.2005.06.003

[6] A. Khan, A. Siddiqa, S. Munib, S.A. Malik, A recent survey of reversible watermarking techniques, Inf. Sci. 279 (2014) 251–272, https://doi.org/10.1016/j.ins.2014.03.118

[7] C.J. Date, An Introduction to Database Systems, eighth ed, Pearson Education India, Delhi, India, 2006, international edition.

[8] M.L. Pérez Gort, C. Feregrino-Uribe, A. Cortesi, F. Fernández-Peña, HQR-Scheme: a high quality and resilient virtual primary key generation approach for watermarking relational data, Expert Syst. Appl. 138 (2019) 112770, https://doi.org/10.1016/j.eswa.2019.06.058

[9] R. Agrawal, J. Kiernan, Watermarking relational databases, in: VLDB'02: Proceedings of the 28th International Conference on Very Large Databases, Elsevier, 2002, pp. 155–166, https://doi.org/10.1016/B978-155860869-6/50022-6

[10] R. Halder, S. Pal, A. Cortesi, Watermarking techniques for relational databases: survey, classification and comparison, J. Univ. Comput. Sci. 16 (21) (2010) 3164–3190, https://doi.org/10.3217/jucs-016-21-3164

[11] H.M. Sardroudi, S. Ibrahim, A new approach for relational database watermarking using image, in: 5th International Conference on Computer Sciences and Convergence Information Technology, IEEE, 2010, pp. 606–610, https://doi.org/10.1109/ICCIT.2010.5711126

[12] Y. Li, V. Swarup, S. Jajodia, Constructing a virtual primary key for fingerprinting relational data, in: Proceedings of the 3rd ACM Workshop on Digital Rights Management, 2003, pp. 133–141, https://doi.org/10.1145/947380.947398

[13] M.L. Pérez Gort, E.A. Díaz, C.F. Uribe, A highly-reliable virtual primary key scheme for relational database watermarking techniques, in: 2017 International Conference on Computational Science and Computational Intelligence (CSCI), IEEE, 2017, pp. 55–60, https://doi.org/10.1109/CSCI.2017.10

[14] T. Liang, Y. Zhao, H. Wang, Z. Cai, Z. Wang, W. Wang, C. Liu, FRBBM-Scheme: a flexible ratio virtual primary key generation approach based on binary matching, in: International Conference on Intelligent Information Processing, Springer, 2024, pp. 438–452, https://doi.org/10.1007/978-3-031-57808-3_32

[15] W. Tan, B. Hu, Y. Zhou, S. He, Y. Liu, T. Liu, X. Guan, A virtual primary key generation scheme based on dimensionality reduction and encoding for digital fingerprinting, in: 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE), IEEE, 2024, pp. 252–257, https://doi.org/10.1109/CASE59546.2024.10711292

[16] L. Pinheiro Cinelli, M. Araújo Marins, E.A. Barros da Silva, S. Lima Netto, Variational autoencoder, in: Variational Methods for Machine Learning with Applications to deep networks, Springer, 2021, pp. 111–149, https://doi.org/10.1007/978-3-030-70679-1_5

[17] K. Yang, S. Yuan, J. Yu, Y. Wang, T. Yang, C. Chen, Don't abandon the primary key: a high-synchronization and robust virtual primary key scheme for watermarking relational databases, in: International Conference on Information and Communications Security, Springer, 2024, pp. 289–309, https://doi.org/10.1007/978-981-97-8801-9_15

[18] C.-C. Chang, T.-S. Nguyen, C.-C. Lin, A blind robust reversible watermark scheme for textual relational databases with virtual primary key, in: International Workshop on Digital Watermarking, Springer, 2014, pp. 75–89, https://doi.org/10.1007/978-3-319-19321-2_6

[19] V. Khanduja, S. Chakraverty, O.P. Verma, Enabling information recovery with ownership using robust multiple watermarks, J. Inf. Secur. Appl. 29 (2016) 80–92, https://doi.org/10.1016/j.jisa.2016.03.005

[20] X. Che, M. Akbari, S. Li, D. Yue, Y. Zhang, L. Chu, Primary key free watermarking for numerical tabular datasets in machine learning, in: International Conference on Pattern Recognition, Springer, 2025, pp. 254–270, https://doi.org/10.1007/978-3-031-78119-3_18

[21] N.R. Lomb, Least-squares frequency analysis of unequally spaced data, Astrophys. Space Sci. 39 (1976) 447–462, https://doi.org/10.1007/BF00648343

[22] Colorado-State-University, Forest CoverType, The UCI KDD Archive, 1999. http://kdd.ics.uci.edu/databases/covertype/covertype.html.