**Evaluation Only. Created with Aspose.Words. Copyright 2003-2022 Aspose Pty Ltd.**



See discussions, stats, and author profiles for this publication a[t:http://www.researchgate.net/publication/277774629](http://www.researchgate.net/publication/277774629_Privacy-Preserving_Data_Warehousing?enrichId=rgreq-c55bc3c3-2fc5-4c33-91d7-fefb762d2106&enrichSource=Y292ZXJQYWdlOzI3Nzc3NDYyOTtBUzoyNDM0OTQ0MzAxNzkzMjhAMTQzNTA2NTAwNDQyOQ%3D%3D&el=1_x_2)



[Privacy-Preserving Data Warehousing](http://www.researchgate.net/publication/277774629_Privacy-Preserving_Data_Warehousing?enrichId=rgreq-c55bc3c3-2fc5-4c33-91d7-fefb762d2106&enrichSource=Y292ZXJQYWdlOzI3Nzc3NDYyOTtBUzoyNDM0OTQ0MzAxNzkzMjhAMTQzNTA2NTAwNDQyOQ%3D%3D&el=1_x_3)

**ARTICLE**in INTERNATIONAL JOURNAL OF BUSINESS INTELLIGENCE ANDDATA MINING · JULY 2015



DOWNLOADS VIEWS

65 14

**2 AUTHORS**, INCLUDING:

[Benjamin Fabian](http://www.researchgate.net/profile/Benjamin_Fabian?enrichId=rgreq-c55bc3c3-2fc5-4c33-91d7-fefb762d2106&enrichSource=Y292ZXJQYWdlOzI3Nzc3NDYyOTtBUzoyNDM0OTQ0MzAxNzkzMjhAMTQzNTA2NTAwNDQyOQ%3D%3D&el=1_x_5) [Humboldt-Universität zu Berli](http://www.researchgate.net/institution/Humboldt-Universitaet_zu_Berlin?enrichId=rgreq-c55bc3c3-2fc5-4c33-91d7-fefb762d2106&enrichSource=Y292ZXJQYWdlOzI3Nzc3NDYyOTtBUzoyNDM0OTQ0MzAxNzkzMjhAMTQzNTA2NTAwNDQyOQ%3D%3D&el=1_x_6)n



**69** PUBLICATIONS**344** CITATIONS

[SEE PROFILE](http://www.researchgate.net/profile/Benjamin_Fabian?enrichId=rgreq-c55bc3c3-2fc5-4c33-91d7-fefb762d2106&enrichSource=Y292ZXJQYWdlOzI3Nzc3NDYyOTtBUzoyNDM0OTQ0MzAxNzkzMjhAMTQzNTA2NTAwNDQyOQ%3D%3D&el=1_x_7)



Available from: Benjamin Fabian

Retrieved on: 01 July 2015

Privacy-Preserving Data Warehousing



Benjamin Fabiana,∗, Tom G¨othlinga



aInstitute of Information Systems Humboldt-Universita¨t zu Berlin Spandauer Str. 1, 10178 Berlin, Germany



Abstract

Data warehouses are an important element of business intelligence and decision support in many companies and inter-organizational data infrastructures. However, when personal information of individuals is concerned, it is critical to provide sufficient protection mechanisms in order to preserve privacy. In addition to classical access control, database anonymization is an important element of an encompassing strategy for privacy-preserving data storage. This article gives an overview on selected anonymization concepts and techniques and investigates if they are suitable

for a data warehouse context. Furthermore, a process of privacy-preserving data integration and provisioning is presented and the impact of architecture, privacy criteria, and further parameter choices is discussed. Finally, we experimentally compare the impact of these parameters on data utility after anonymization in several experiments on multiple data sets and derive corresponding recommendations.

Keywords: Data Warehouse, Data Integration, Privacy, Anonymity



1. Introduction

Strong growth of the data integration market indicates that companies continuously consolidate their data into data warehouses in order to improve availability and create added value [1]. Because data warehouses usually store sensitive information such as customer or sales insights, they repre- sent a potential target of privacy attacks. The impact of such attacks could be severe. In 2011, a privacy breach of the Sony Play Station network disclosed sensitive user data, including credit card information of 77 million users [2]; in 2013, the market data provider Bloomberg inadvertently allowed reporters to access restricted customer data [3]. Besides external adversaries and inadver- tently published data, employees are also a potential threat to companies’ data. Furthermore, even data that was neither stolen nor published inadvertently can reveal sensitive information. As a re- sult of such privacy breaches, personal data privacy has increasingly caught attention of the public and leads European Data Protection authorities to encourage stricter legislations with regards to personal data processing [4]. Consequently, more and more companies realize the strategic matter

of data privacy [5][6] and data protection has developed into a key issue of an enterprise IT strategy.

Data anonymization seems to be a reasonable approach to mitigate the impact of internal data thefts and to avoid inferences within published datasets. However, research proved that intuitive

∗Corresponding author. Email address: bfabian@wiwi.hu-berlin.de. Phone: +49 30 20935662.



Email addresses: bfabian@wiwi.hu-berlin.de (Benjamin Fabian), tomgoethling@gmx.de (Tom Go¨thling) Preprint submitted to International Journal of Business Intelligence and Data Mining June 23, 2015

methods of data anonymization, such as suppressing directly identifying information, are inappro- priate for guaranteeing anonymity as multiple approaches exist to circumvent such mechanisms. Consequently, over the last decades multiple mechanisms have been developed to create anonymized datasets which guarantee a specific level of data privacy. However, most approaches are based on idealistic assumptions of data management and do not cover the setting of data warehouses. Thus, the application of these mechanisms in data warehouses remains an open research question.



This article will investigate how anonymization approaches can be applied to data warehouse scenarios. The goal is to identify critical process steps in data warehousing with regard to data pri- vacy and provide architectural alternatives to manage them by adopting existing privacy concepts. Since data warehouses are mainly used to perform data analysis, the implications of architectural design decisions on data utility will be evaluated. Possibilities and limitations of the application

of existing approaches in data warehouses are analyzed. Furthermore, this work will focus on data privacy mechanisms based on anonymization that can be applied in addition to access control [7]

and further classical security measures [8].

The article is structured as follows. First, in section 2 work related to the general research area of security and privacy in data warehouses is discussed. Fundamental concepts of database anonymization and related work in this specific field are introduced in section 3. This includes mechanisms to sanitize data, utility metrics, possible attack models on static and dynamic data as

well as privacy criteria in order to preserve data privacy. Section 4 provides insight into different data warehouse architectures. Critical process steps regarding privacy issues will be identified and exemplified with the help of two sample architectures. Afterwards, possible approaches for manag-

ing these process steps will be presented. Section 5 evaluates implications of the different design parameters on data utility based on a prototype which simulates different sample architectures. This is followed by a discussion of potential implementations of the results shown. Implications on data utility are evaluated by applying multiple utility metrics to the results of the simulation.

1. Related Work

Data warehouses are an important foundation for decision support in many commercial and other organizations and corresponding organizational and technical planning processes as well as

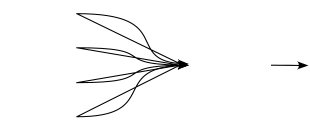
the choice of the right architecture involve many practical challenges [9]. In particular, establishing information security issues for such integrated data stores with confidential content is a challenge

in the face of sophisticated adversaries and heterogeneous attack vectors, moreover, when data is integrated and shared with several organizations such as partner firms or subcontractors [10] [11]. In such settings, classical access control mechanisms [12] need to be adapted to the multidimensional models of data warehousing [13], but they also need to evolve to cope with federated settings [14] and new technologies such as semantic data stores [15] [8], in-memory databases [16][17], and RFID-based information exchange [18].

Beyond access control, further protection mechanisms for confidentiality and privacy are needed

as motivated by the examples of security breaches given in the Introduction. Here, privacy- preserving data mining [19] [20] could improve privacy of individual data subjects if analytical facilities are trustworthy. However, only data sanitization [21] can provide measures to protect pri- vacy even in the face of potentially adversarial data warehouse providers or data consumers. One important area of research in data sanitization is anonymization, on which detailed literature will be provided in section 3. To the best of our knowledge, challenges and trade-offs of anonymization

Individual 1 1 Individual 2



Anonymized Data

Raw Data

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

2

3

4

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

Data User / Adversary

Data Publisher

Individual 3

...

Collection Anonymization Publishing



Figure 1: Process of privacy preservation in databases

in data warehousing have so far not received sufficient attention in the literature. The current article is motivated by this research gap.

1. Anonymization Process and Measures
2. Fundamentals and Process of Anonymization

As shown in Figure 1, a typical scenario of privacy preservation in databases consists of three phases – collection, anonymization, and publishing. The aim of the collection phase is to gather data (1) and to transform them into a usable format for data analysis (2). Usually, the collected

raw data contains sensitive information. Hence, data publishers have to implement measures to protect this information against privacy disclosure. This is usually enforced by anonymization (3).

In accordance with [22, 23, 24], anonymization can be defined as a process of converting data in order to hide the identity of each record owner, assuming that sensitive attribute values must be retained. In the next step, the anonymized dataset is published to data users (4). Before release, the data publisher has to decide what kind of access should be granted to data users. Principally,

it can be distinguished between full access releases, where the whole dataset is available for data recipients, and query-based approaches, which only allow restricted queries [25].

In an untrusted publisher model, data publishers may attempt to extract sensitive information from the dataset [26]. As a consequence, sensitive information has to be protected before making the data available to the publisher because he represents a potential attacker. In contrast, the trusted publisher model assumes a trustworthy data publisher in a secure environment. This enables new possibilities to ensure data privacy because data publishers can operate on the raw data. In case of data integration scenarios, both models are realistic. For example, an integration project which operates within a single corporation will rather assume a trusted data publisher, whereas cross-company integration projects probably tend to use the untrusted model.

Moreover, data publishers have the choice of releasing aggregated or non-aggregated data [25]. Non-aggregated data provide information on the detail level of individuals [27] (for example, per- sons, products, or departments) and are also called microdata. Each row of a microdata table conventionally represents one individual unit of a given population. Each column contains values

of an attribute of the individuals. According to [23], microdata tables contain the following types

of attributes: Identifier attributes contain information that explicitly identify record owners, for example, social security number or name. Quasi Identifier attributes potentially identify record owners when combined, for example, gender, age, and ZIP code. QI-groups in a dataset are the subsets of tuples that share the same values for the quasi identifiers. Sensitive attributes represent sensitive individual-specific information that has to be protected against privacy disclosure, for ex- ample, disease or salary. Non-sensitive attributes contain neither identifying information nor hold other information which deserve protection, for example, metadata.



Depending on the intended use of data, data publishers either provide the entire dataset or only allow queries on the data without revealing it. In the former case, the data publisher has to anonymize data in a way that satisfies privacy goals. Unfortunately, it is not sufficient to only suppress identifying attributes because attackers may use quasi-identifier attributes to identify records representing an individual. Thus, data publishers have to apply advanced techniques that are presented in section 3.4. Query-based approaches, for example, SQL queries or statistical queries [28], require restrictions in order to ensure privacy of individuals [29]. Aggregated data are commonly provided in contingency tables, which contain count information about the frequency

of attribute combinations and are used by statistical bureaus for releasing statistics on microdata [30]. In case of data integration systems, data publishers usually use microdata tables for flexible data handling and avoiding data loss caused by aggregation [31].

1. Mechanisms and Criteria for Anonymization

Data publishers usually have to manage two fundamental contrary goals. On the one hand, data users are interested in receiving as much detailed data as possible to achieve good utility.

On the other hand, the privacy of individuals has to be preserved, which is usually related to a certain loss of information. To assess and manage this trade-off between privacy and utility, data publishers can utilize sanitization mechanisms, privacy criteria, and utility metrics. A sanitization mechanism reduces the precision of a dataset [25]. It is applied to a raw dataset and converts it

into anonymized data. Sanitization mechanisms include suppression, generalization, and swapping of attribute values as well as more advanced techniques such as anatomization [32], the addition of noise or synthetic data, or multi-view releases.

Privacy criteria define properties that a sanitized dataset has to satisfy in order to guarantee

a specific level of data privacy. A sanitized dataset which satisfies a specific privacy criterion is called a release candidate. The definition of a privacy criterion is mostly related to an attack model because an attacker’s background knowledge strongly affects requirements on data sanitization. Privacy criteria can only guarantee a specific level of privacy. For example, if a privacy criterion requires that at least three tuples have equal quasi-identifier attribute values and different sensitive values, it prevents the identification of the record for a particular individual even if adversaries

have knowledge about quasi-identifier values. However, such adversaries would be able to constrain

the disease of an individual to three potential values. In order to quantify utility of anonymized datasets, utility metrics measure the loss of information caused by sanitization. If multiple release candidates satisfy a privacy criterion, utility metrics will be an appropriate instrument to select the most useful candidate.

1. Privacy Preservation for Dynamic Datasets

Most attack models discussed in the literature assume static datasets. For data warehouses, this assumption is not reasonable because datasets can change over time, sources can become un- available, or additional sources deliver new information. Those changes on the dataset extend the capability of adversaries to infer sensitive information because information can be linked across different releases. The existing literature distinguishes between different application scenarios of dynamic datasets that have different properties with respect to operations allowed and adversaries’ background knowledge. Multiple-Release Publishing concerns scenarios where data publishers pro- vide different sanitized projections of a given dataset. Privacy is threatened because adversaries



may be able to link information between different releases. However, the tuples of the underlying microdata do not change between different releases. Sequential-Release Publishing handles scenar- ios where insert and/or delete operations are performed on the microdata. Hence, underlying data differ from release to release. Collaborative Data Publishing describes the situation where multiple data publishers provide different pieces of data to integrate them into a single dataset. The goal is that no data publisher should be able to infer more information than is available from his own

and the final integrated dataset [24]. The original datasets remain stable. Multiple-release and collaborative data publishing do not cover situations where microdata is changed. Thus, if changes on data are possible or necessary, data publishers have to apply sequential-release publishing tech- niques to preserve privacy. For this reason, we will concentrate in the following on sequential-release publishing.

In addition to all attack models that can be applied to static datasets, dynamic datasets can also be the target of sequential attacks. An intuitive method to attack dynamic data is to compare attribute values of individuals through a sequence of releases (Generalization-Linking Attack ). If records are generalized differently, for each attribute and each tuple, an attacker can select the less generalized value. As a result, the attacker can link information from several releases and consequently get more detailed information about an individual’s quasi-identifier values. In a so- called Exclusion Attack, adversaries have background knowledge about which release contains which tuples and will be able to exclude possible sensitive attribute values by comparing different releases.

By intersecting sensitive attribute values of each QI-group of a tuple across a sequence of releases, attackers can infer the sensitive attribute value of it. In case of recently added tuples, attackers use previous tuples to exclude sensitive attribute values. Set-Theoretical Attacks extend methods from the Exclusion Attack model and show that, even if adversaries only have knowledge about

the values of quasi-identifier attributes of one individual as well as the release table at which the tuple was inserted, simple set-theoretical methods will enable them to reveal sensitive information. Critical Absence Attacks require background knowledge about the lifespan of at least one tuple which is a realistic assumption. It is easy to apply because adversaries only have to track a record

over its lifespan. If delete operations are applied to a dataset, this attack represents an appropriate technique to extract sensitive information.

The previously considered attacks have the aim to associate a set of sensitive attribute values to an individual. Unlike these approaches, with Value-Equivalence Attacks, adversaries try to determine if two multisets of tuples are associated to the same multiset of sensitive attribute values [33]. Such attacks add a new perspective to the set of attack models because adversaries use dependencies between individuals to determine sensitive information. If data publishers provide samples of a dataset or parts of the dataset are available in other sources, this attack will become a notable threat to data privacy. Possible World Exclusion Attacks represent another possibility to attack privacy in scenarios where changes of some values are possible but specific sensitive values of tuples remain unchanged, for example, incurable diseases or criminal records. Such attacks are conducted by making assumptions on the linkage between an individual and his or her sensitive value. If the linkage is possible through the sequence of sanitized tables, adversaries can infer additional information [34]. Possible World Exclusion Attacks follow the principle of trial and error. Therefore, this attack is accompanied by high computational costs compared to other attack models. Furthermore, it requires that some sensitive values remain linked to their individuals on a sequence of releases. Finally, Background Knowledge Attacks are possible if sensitive values are correlated to each other and can change. Adversaries then have the possibility to use general background knowledge about the sensitive values as well as sequential background knowledge resulting from changes of sensitive values for breaking privacy [35].



1. Privacy Criteria for Sequential Releases

Some intuitive methods exist to protect data on sequential release publishing. One possible idea is to anonymize and publish new records separately [36]. However, the major disadvantage

of this approach is that releases become overgeneralized. Tuples from different updates cannot

be combined to form a more specific QI-group. Thus, the approach suffers from bad data utility. Another approach is to apply privacy criteria for static datasets but, as shown in section 3.3,

this approach is not appropriate to preserve individuals’ privacy. An intuitive method to manage scenarios where delete operations are allowed is to ignore these operations [37]. However, this approach is not suitable for some application scenarios, for example, if deletions must be performed

by legal regulations. Furthermore, data could suggest wrong implications. For example, some diseases such as flu epidemics occur in limited periods. If records are not deleted, data could suggest that patients suffer from flu even if this is not the case. Considering the disadvantages of

these approaches, it becomes obvious that these mechanisms either are not appropriate to preserve privacy or suffer from bad data utility. In order to present a solution to this challenge, several major concepts have been introduced in the literature.

The first concept, k-Anonymity Against Updates, was proposed by [38] as a privacy criterion

to maintain k-Anonymity [39, 40] with incremental updates which protects against Generalization- Linking attacks. k-Anonymity is defined as follows: Given a dataset T with quasi-identifier at- tributes q1, ..., qn; T is called k-anonymous, if every combination of values of q1, ..., qn occurs at least k times. The idea of k-Anonymity against updates is to avoid different generalizations

on a sequence of releases in order to prevent adversaries from inferring QI-attributes of a record

by comparing different releases. Although this approach guarantees k-Anonymity on a sequence

of releases, it has various weaknesses with respect to privacy preservation. First, k-Anonymity is vulnerable to attacks against sensitive attribute values, for example, Homogeneity Attacks [41]. Second, the approach assumes a weak attack model with regard to background knowledge. For ex- ample, it is assumed that the attacker has no knowledge about which release contains which records. Furthermore, the criterion is restricted to insert-only scenarios. On the other hand, computational costs are low compared to other privacy criteria of sequential release publishing scenarios.

The second concept uses l-Diversity [41] as a baseline. Unlike k-Anonymity, l-Diversity takes sensitive attribute values of the QI-groups into account. While k-Anonymity can be attacked by homogeneity or background knowledge attacks, l-Diversity extends k-Anonymity by enforcing dis-

tinct sensitive attribute values within the QI-groups [41]. Formally, a QI-group q is called l-diverse

if it contains at least l well represented values for the sensitive attribute S. A dataset T complies to l-Diversity if every QI-group is l-diverse. This definition is still flexible with respect to the exact definition of ”well-represented”. One straightforward choice is that each QI-group must contain

at least l distinct sensitive attribute values. Incremental l-Diversity [36] is an update mechanism to prevent Exclusion as well as Set-Theoretical Attacks on update-only scenarios. It is assumed

that adversaries have knowledge about quasi-identifier attributes and know which individuals are included in which release tables. The basic idea is that new tuples will be inserted only if they do

not compromise privacy of existing tuples. Thus, waiting lists temporarily store tuples which so



far could not be inserted [36]. Major advantages of this approach are low computational costs and



an improved privacy level compared to k-Anonymity against updates. However, it does not cover scenarios where delete or update operations are allowed.

The third concept, m-Invariance [37], guarantees privacy in scenarios that allow insert as well

as delete operations. Adversaries are assumed to possess knowledge about the quasi-identifier attribute values of each tuple as well as knowledge about which release contains which tuples. Missing sensitive values allow adversaries to apply critical absence attacks [37]. Correspondingly,

the basic idea of m-Invariance is to avoid such attacks by holding the signature of a tuple’s QI- groups constant over its lifespan where the signature of a QI-group q is the set of distinct sensitive values. In order to document the lifespan of a tuple, each tuple t in a sequence of sanitized tables

T 0,..., T n is augmented with a timestamp attribute. The generalized historical union Un contains

all timestamped tuples for all publishing times. Furthermore, m-Uniqueness is defined in [37] which is similar to l-Diversity: A sanitized table T is m-unique if each QI-group in T contains at least

m tuples and all tuples in the group have different sensitive values. Then, m-Invariance is defined as follows [37]: A sequence of sanitized tables T 0,..., T n with n ≥ 1 is m-invariant if both of the

following conditions hold: (1) T j is m-unique for all j ∈[1,n], (2) for any tuple t ∈Un with lifespan [x,y], qx(t),qx+1 (t),...,qy(t) have the same signature where qj(t) is the QI-group of t at time j.

One can show that if a sequence of sanitized tables T 0, ..., T is m-invariant, then T 0, ...,T n−1,

T n will be also m-invariant if T is m-unique and for any tuplne−t1∈ T ∩T , t’s QI-groups have

the same signature [37]. This prnoperty enables data publishers to creat n−e 1a ne nw release table T n+1 , by only consulting Tn,Tn+1, and T n, which significantly simplifies the process of anonymization.

There are situations where maintaining m-Invariance is not possible [37]; this is mitigated by using synthetic counterfeits tuples which are augmented to existing QI-groups in order to preserve their signatures. Publishing the number of counterfeits tuples of each QI-group has no influence on data privacy. M-Invariance disables generalization-based attacks such as Exclusion Attacks and Generalization Linking Attacks as well as attacks that compare sensitive values such as Critical Absence Attacks or Set-Theoretical Attacks. However, [33] shows that m-Invariance is not able

to protect data against Value-Equivalence Attacks. Moreover, it does not provide appropriate protection in scenarios which allow updates of values of existing tuples.

A further concept applies Graph-based Anonymization. In [33], the authors also proposed a graph-based approach to modify and extend m-Invariance so that Value-Equivalence Attacks be- come impossible. Since m-invariant tables already imply the existence of m-Value-Equivalence attacks, this approach has the goal to prevent e-Value-Equivalence Attacks with e < m. To prevent such attacks, tuples cannot always be published in QI-groups with each sensitive attribute value occurring only once because these QI-groups often expose correspondence structures. Instead, [33] propose to merge QI-groups sharing the same signature. In order to preserve data utility, they use anatomization to publish original quasi identifier attribute values. However, this modification is still not sufficient to protect data against e-Value-Equivalence Attacks.

Another concept is called HD-Composition. [34] proposes a generalization technique which applies l-Diversity to dynamic datasets. They assume a scenario where quasi-identifier attribute values as well as most sensitive attribute values of an individual can change over time. They further assume that individuals can be linked to more than one sensitive value. Sensitive values that once have been linked to an individual and can never be unlinked are called permanent sensitive values. In contrast, changeable sensitive attributes are called transient sensitive values. They further assume

that adversaries have knowledge about quasi-identifier attribute values of each individual as well as the lifespan of a fraction of individuals. Note, that for now, we have assumed that attackers have knowledge about the lifespan of all tuples. The basic idea of HD-Composition is to focus privacy-preserving measures on tuples with permanent sensitive values, called s-holder, because these tuples create inferences across a sequence of releases. The approach of [34] uses tuples with transient sensitive values, called decoys, to cloak s-holders. Each decoy has the goal to cloak a specific permanent sensitive value. The key idea is that each QI-group must contain a specific number of decoys for each s-holder. In [34] it is shown that HD-Composition provides effective protection against Possible World Exclusion Attacks. Furthermore, it is an appropriate approach for scenarios where values of existing tuples can change. However, to apply this methodology sensitive values must be approximately uniformly distributed [35]. Another disadvantage results from the fact that transient sensitive values are not protected against sequential knowledge. Even



if attackers have no guarantee that transient sensitive values of existing tuples remain unchanged, they may assume these properties about transient values which rarely change. In this case, privacy of transient values is not preserved. Thus, privacy protection only covers a specific setting.

The final major concept is JS-Reduce introduced by [35]. This is a privacy mechanism for scenarios where attackers have knowledge of quasi identifier attribute values of individuals, about probabilities of correlations between individuals and sensitive values as well as probabilities of se- quences of sensitive values. Furthermore, they assume changing sensitive values. The JS-Reduce criterion is used to protect data against Sensitive Value Background Knowledge Attacks. The basic idea is to form QI-groups whose tuples have similar distributions with regard to sensitive values background knowledge. In [35] similarity of probability distributions is measured by ap- plying the Jenson-Shannon Divergence [42]. To protect data, data publishers use sensitive value background knowledge and sequential background knowledge to compute posterior background knowledge. They further calculate revised background knowledge which intuitively calculates de- cision trees for each combination of individuals and sensitive values. These computations are used to create similar QI-groups with regard to revised background knowledge.

JS-Reduce can be combined with further privacy criteria such as m-Invariance or l-Diversity. Thus, it can provide a privacy guarantee even under strong assumptions with regard to adversary background knowledge. Unlike HD-Composition, JS-Reduce is also applicable if sensitive values are not uniformly distributed. However, compared to other privacy criteria, computational costs are very high, which leads to limited practicality especially for large datasets. Furthermore, Sen- sitive Value Background Knowledge Attacks require specific assumptions on correlations within

the dataset. These assumptions do not hold in most application scenarios. In consequence, data publishers have to evaluate if protection against this attack is necessary.

1. Further Related Research

The literature also provides some other approaches to protect data in multiple release publishing scenarios. [43] showed that releasing additional count tables of combinations of values, for example, the number of tuples with gender attribute value ”male” and ”age < 30” could increase utility. Unfortunately, they also lead to threats in privacy. Therefore, [43] proposed a criterion to check whether such a marginal leads to privacy risks or not. If previously published releases of a dataset already exist, [44] introduced the concept of lossy joins to create additional releases. The basic idea is that joining different views of a dataset must create additional tuples which do not represent

real world objects. Generalization has to guarantee that join attributes between views match more

than one record owner. The goal is that even if adversaries have knowledge about all releases, they should not be able to disclose privacy by joining them.



[45] suggests to base generalization on the previous releases instead of the microdata. This leads

to the property that new releases are not more specific than previous releases. Unfortunately, if values become more generalized in later releases, then data utility usually decreases as well. As mentioned in section 3.3, collaborative data publishing describes scenarios where different data owners merge their data with the goal that no data owner extracts additional knowledge than provided in the merged dataset. [46] propose an approach of a two party integration where both parties initially generalize their dataset to the most general values. Then, values are iteratively refined by both parties to find the best global specialization. The party which owns an attribute to be specialized provides the IDs of the QI-groups that were generated to the second party which also creates these QI-groups in its dataset. [47] and [48] introduced a solution for the same problem.

In their approach, cryptographic techniques are used to exchange information. Both parties create k-anonymous tables. Then, intersections between QI-groups are evaluated. If an intersection of two QI-groups contains at least k tuples, joining both QI-groups will represent a QI-group in the global dataset. To prevent information disclosure by exchanging tuple IDs of QI-groups, cryptographic methods are applied to compute intersections of QI-groups.

1. Comparison of Criteria

Table 1 shows which of the earlier discussed major privacy criteria are suitable to preserve privacy against what type of attack. Since assumptions about background knowledge and data ma- nipulation operations can differ between privacy criteria and attack models, attack model scenarios are assumed on assessing compatibilities of privacy criteria. Table 1 also presents the allowed data manipulation operations for each privacy criterion.

Table 1: Comparison of privacy criteria of sequential release publishing scenarios.

(Notation: GLA = Generalization-Linking Attack, EA = Exclusion, STA = Set-Theoretical, CAA = Critical Absence, VEA = Value-Equivalence, PWE = Possible World Exclusion, SVBK = Sensitive Value Background Knowledge; i = insert, d = delete, u = update.)



Operations GLA EA STA CAA VEA PWE SVBK Allowed



K-Anonymity ag. Updates i √√ √× √× × × × × Incremental l-Diversity i √ √ √ √× × × ×

M-Invariance i/d √ √ √ × × × Graph-based Anonymization i/d √ √ √ √ √√ × ×

HD-Composition i/d/u √ √ √ √ √ √ ×

√ √ √

JS-Reduce i/d/u



By only considering Table 1 alone, JS-Reduce appears to be the best privacy criterion for sequential release publishing because it provides a broad protection against multiple privacy at- tacks. However, data publishers additionally have to take two essential aspects into account. First, computational costs differ considerably between privacy criteria. For example, JS-Reduce requires computations of several probabilities for each tuple over all releases, whereas Incremental l-Diversity only needs to store previous generalizations. Data publishers have to evaluate which privacy cri- teria can be applied based on the available resources. Second, most of the attack models are only applicable to specific settings of data structures, background knowledge, and data manipulation

operations. For example, Critical Absence Attacks are only possible if delete operations are per- formed on the data. Data publishers need to analyze whether attack models constitute a real threat or may not be applicable to the given setting.



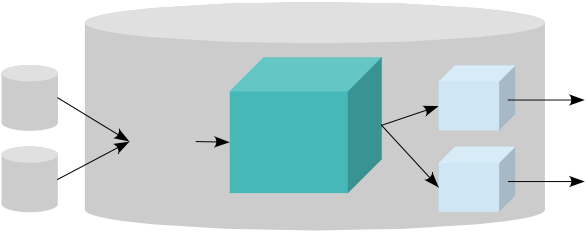
Privacy Criteria that work in Multiple Release Publishing scenarios can be applied to achieve data privacy if the structure of the dataset changes across the published releases, for example, by addition of a new quasi-identifier attribute. However, they do not cover data manipulation of the underlying dataset. Privacy criteria that manage data privacy in Sequential Release Publishing scenarios cover data manipulation across multiple releases but do not allow for changes in data structure. Hence, privacy preservation in scenarios where data as well as its structure can change is still an open research question.

1. Data Warehouse Architectures and Privacy Preservation
2. Data Warehouse Architectures

The basic principle of data warehouse architectures is the replication of data from heterogeneous sources into a single database. Queries are executed directly on the database without involving any of the sources at the time of query processing. This approach offers a considerable advantage

in response time compared to virtual integration. Storing the data within the data warehouse furthermore enables aggregation, manipulation, and cleansing of data. Data warehouses are also called Online Analytical Processing (OLAP) systems. Typical characteristics are high storage capacity, few but complex analytic queries with high data volume, no manipulation of existing records, periodical bulk inserts, a multidimensional data model, and business focus [49].

Metadata Repository



Data Mart 1



Source 1 Source 2

Staging Area

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

Data Cube

Data Mart 2

Figure 2: Data warehouse architecture

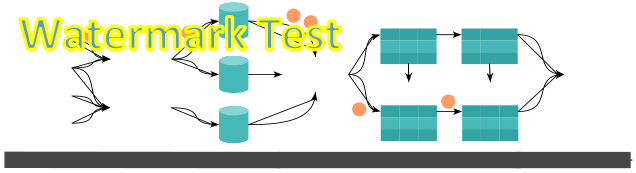
Figure 2 illustrates an exemplary architecture of data warehouses. The integration process of

data warehouses is known as Extract, Transform, Load (ETL). The extraction phase includes all steps for extracting data from sources to the staging area, for example, executing export scripts, file parsing, or duplicate elimination [49]. In the transformation phase, data is transformed into

the data structure of the data warehouse and stored in the staging area, which works as a buffer between sources and data cube, allowing for data cleansing and more efficient transformations.

The multidimensional data cube represents the central storage component of a data warehouse

**1c** Raw Data Anonymized



Source **5** 3 Data

**1a 1b**

Individual 1 2 4

Data Collector

Individual 2 Data User/

Source Data Publisher Adversary Individual 3 Raw Data 2 Anonymized

Data 2

**7**

Data Collector **6**

**This document was truncated here because it was created in the Evaluation Mode.**