applied sciences

MDPI

Article

Efficiently Supporting Online Privacy-Preserving Data Publishing in a Distributed Computing Environment

Jong Wook Kim

Department of Computer Science, Sangmyung University, Seoul 03016, Korea; jkim@smu.ac.kr

Abstract: There has recently been an increasing need for the collection and sharing of microdata containing information regarding an individual entity. Because microdata typically contain sensitive information on an individual, releasing it directly for public use may violate existing privacy requirements. Thus, extensive studies have been conducted on privacy-preserving data publishing (PPDP), which ensures that any microdata released satisfy the privacy policy requirements. Most existing privacy-preserving data publishing algorithms consider a scenario in which a data publisher, receiving a request for the release of data containing personal information, anonymizes the data prior to publishing—a process that is usually conducted offline. However, with the increasing demand for the sharing of data among various parties, it is more desirable to integrate the data anonymization functionality into existing systems that are capable of supporting online query processing. Thus, we developed a novel scheme that is able to efficiently anonymize the query results on the fly, and thus support efficient online privacy-preserving data publishing. In particular, given a user's query, the proposed approach effectively estimates the generalization level of each quasi-identifier attribute, thereby achieving the *k*-anonymity property in the query result datasets based on the statistical information without applying *k*-anonymity on all actual datasets, which is a costly procedure. The experiment results show that, through the proposed method, significant gains in processing time can be achieved.

Keywords: privacy-preserving data publishing; *k*-anonymity; distributed query processing

Citation: Kim, J.W. Efficiently Supporting Online Privacy-Preserving Data Publishing in a Distributed Computing Environment. *Appl. Sci.* **2021**, *11*, 10740. https://doi.org/10.3390/app112210740

Academic Editor: Byung-Seo Kim

Received: 30 September 2021
Accepted: 8 November 2021
Published: 14 November 2021

1. Introduction

Recently, there has been an increasing need for the collection and sharing of microdata, which contain information on an individual entity. Microdata are a valuable source of information in diverse areas. Many different organizations, including healthcare providers, apply data analysis techniques to large volumes of microdata to extract hidden knowledge with the goal of improving their decision-making capabilities. However, microdata typically contain sensitive information about an individual, and thus directly releasing such data for public use may violate existing privacy requirements. To avoid the privacy problems that occur through the release of microdata for public use, extensive studies have been conducted in the area of privacy-preserving data publishing (PPDP) [1–4]. These methods ensure that the microdata released satisfy the privacy requirements, such as *k*-anonymity [1,2]. Although such methods differ in the way in which the original mircodata are transformed into another format that is releasable for public use, they are all based on the same principle, that is, individuals cannot be uniquely identified in the data released.

Most existing privacy-preserving data publishing algorithms consider a scenario in which data owners receiving a request for the release of data containing personal information anonymize the data before being published, which is conducted offline. However, with the increasing demand for data sharing among various parties, such an offline data publishing scenario is insufficient to support the voluminous request of a data release. Instead, it is more desirable to integrate the data anonymization functionality into existing systems that are capable of supporting online query processing, such as a database management

system or data warehouse. For example, as our motivating application, let us consider the example scenario shown in Figure 1, in which databases are managed by either a database management system or a data warehouse. Here, a data publisher can submit a normal SQL query with anonymization parameters to the system. Then, the system returns the resulting anonymized set to the data publisher who, in turn, releases it for data analytics purposes to data users. However, thus far, the existing privacy-preserving data publishing techniques have overlooked this online privacy-preserving data publishing scenario.

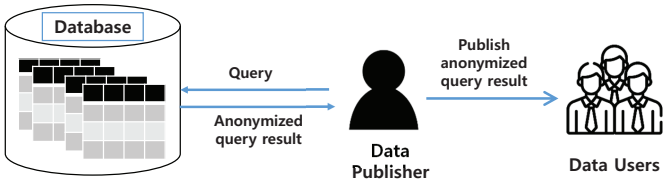


Figure 1. System architecture assumed in this paper.

The major challenge in supporting online privacy-preserving data publishing arises from the efficiency of query processing. Applying data anonymization for query results during the query processing phase clearly adds significant overhead, thereby resulting in a degraded query performance. Thus, to address this efficiency challenge, in this study, we developed a novel scheme that is able to efficiently anonymize the query results on the fly, and therefore eventually support efficient online privacy-preserving data publishing.

In an online query audition, an aggregate query, such as sum, max or min, posed over sensitive data is denied if the query result can reveal sensitive information [5–7]. That is, given a sequence of queries that have already been answered to users, the online query audition denies a new query whenever an answer to the query, along with previous query results, can reveal private data. Furthermore, it is known that anonymization methods are vulnerable to attacks of re-identification caused by the release of multiple anonymous datasets when failing to consider previously released anonymous datasets. Thus, in the literature, extensive studies have been conducted to support continuous data publishing using anonymization methods [8–12]. For example, Wang and Fung [9] studied the problem of sequentially releasing k-anonymous tables with different sets of attributes of the original table. In addition, Fung et al. [10] addressed the problem of continuously publishing k-anonymous tables of an original table into which a new set of records is continuously inserted. Moreover, Xiao and Tao [11] proposed an anonymization algorithm that supports the continuous publication of microdata in the presence of an inserted, deleted, and updated set of records. We note that the method proposed in this paper is a general framework that can be extended along with these existing methods to support a continuous anonymous table release.

The rest of this paper is structured as follows: In the next section, we present the related work. In Section 3, we introduce the background, and then formally define the problem addressed in this paper. In Section 4, we present our algorithm for efficiently supporting online privacy-preserving data publishing. In Section 5, we experimentally evaluate our approach using real datasets. Finally, we provide some concluding remarks in Section 6.

2. Related Work

Extensive studies have been conducted in the area of privacy-preserving data publishing (PPDP). The most popular anonymization algorithm, k-anonymity, was first formulated in [1]. Various algorithms have been proposed to achieve the k-anonymity requirement. LeFevre et al. finds full-domain optimal k-anonymous generalizations with a bottom-up pruning approach [2]. Wang et al. proposed a bottom-up generalization algorithm to find a minimal k-anonymization for classification [13]. Fung et al. presented the top-down specialization scheme in which the specialization process terminates if further specialization on quasi-identifier attribute values violates the k-anonymity requirement [14]. Mondrian [15]

is a multidimensional generalization model that anonymizes data by recursively partitioning the space across the dimension. Clustering-based methods have been proposed to effectively find the k-anonymous table. For example, [16,17] group k similar records into a cluster and generalize each cluster to achieve k-anonymity. Besides k-anonymity, many privacy metrics have been proposed in the literature. Machanavajjhala et al. [3] introduced l-diversity that requires that each equivalence has at least l well represented values of a sensitive attribute. Li et al. proposed t-closeness that requires that the distribution of a sensitive attribute in each equivalence class is similar to the distribution of the entire table [4]. p^+-sensitive k-anonymity was proposed to prevent similarity attacks, and thus to reduce the potential threat for attribute disclosure [18–20]. Kim et al. [21] developed a delay-free anonymization method to publish electronic health data streams. In [22], a utility-preserving anonymization method for PPDP, which preserves the utility of health data by inserting counterfeit records and creating a catalog of the counterfeit records in the process of data anonymization, was proposed. Khan et al. [23] introduced the θ-sensitive k-anonymity privacy model, in which the threshold θ determines the diversity level of an equivalence class, to prevent the sensitive variance attack when publishing electronic health records. A comprehensive survey of privacy-preserving data publishing can be found in [24–29].

Differential privacy (DP) [30], which is the strongest scheme for protecting individuals' privacy in released data, has been extensively studied in diverse areas, including data mining and medical analysis. DP guarantees that an attacker with arbitrary background knowledge cannot infer with high confidence whether a particular individual is participating in the query result (or the published data). DP can be used in two different settings. The first one is the offline setting where a statistical summary, such as histograms or a set of synthetic data that mimic the original data, is released for public use [31]. The second one is the online setting, where the user issues a statistical query to the original database, and then a perturbed version of the query result is returned to the user [32]. With its strong privacy guarantees, DP has been used in various application areas and many variants of DP have been proposed in the literature, such as local differential privacy [33–36] and geo-indistinguishability [37,38]. DP can be used for publishing location data in a privacy-preserving manner by using a spatial histogram [39,40]. DP complaint spatial histograms are constructed by first partitioning a spatial domain into several cells and then adding carefully calibrated noise to the true count of objects located within the boundaries of each cell. Unlike anonymization methods, DP is mostly used for the release of aggregated results, such as histograms or cross tabulations. However, several recent attempts have been made to apply DP along with an anonymization algorithm to the publishing of microdata. For example, Lee and Chung [41] proposed a method for releasing the ϵ-DP version of an original dataset. The method proposed in [41] uses anonymization methods based on generalization, suppression, and insertion, along with DP to generate an ϵ-DP version of an original dataset. Guo et al. [42] proposed a method based on the combination of k-anonymity and DP for publishing physiological signals collected by wearable devices.

3. Background and Problem Statement

3.1. Background

The most popular anonymization algorithm, k-anonymity, was formulated in [1]. The k-anonymity algorithm guarantees that, for each record, there are at least $k-1$ other records included in the released data that have the same values for a set of quasi-identifier attributes (which are defined as special attributes that can be linked with external data to uniquely identify individual records in the released data), thereby ensuring that every record in the released data is indistinguishable from at least $(k-1)$ others, despite a linkage attack [1,2]. Each record in a dataset is generalized into an indistinguishable group, called the equivalent class, by replacing the specific values of the quasi-identifier attributes with more general values. For instance, let us consider the example table in Figure 2a, in which the attributes, *Age* and *Zip*, are quasi-identifier attributes, and the attribute, *Disease*, is a

sensitive attribute. Let us further assume that the domain generalization hierarchies of *Age* and *Zip* are defined as in Figure 3. Then, the *k*-anonymous table in Figure 2b is obtained by replacing the values of the quasi-identifier attributes, *Age* and *Zip*, of each record with more general values defined in the domain generalization hierarchies. For example, the first record, $\langle 14, 3068, Pneumonia \rangle$, in Figure 2a is generalized as $\langle$10–20, 3060–3070, $Pneumonia\rangle$ in Figure 2b, and thus is indistinguishable from the next three records (i.e., RID = 2, 3, 4).

RID	Age	Zip	Disease
1	14	3068	Pneumonia
2	15	3061	Diabetes
3	16	3069	Anemia
4	19	3069	Pneumonia
5	23	3074	Anemia
6	28	3079	Diabetes
7	23	3071	Pneumonia

(**a**)

RID	Age	Zip	Disease
1	10-20	3060-3070	Pneumonia
2	10-20	3060-3070	Diabetes
3	10-20	3060-3070	Anemia
4	10-20	3060-3070	Pneumonia
5	21-30	3071-3080	Anemia
6	21-30	3071-3080	Diabetes
7	21-30	3071-3080	Pneumonia

(**b**)

Figure 2. (**a**) Original table and (**b**) *k*-anonymized table ($k = 3$).

Many *k*-anonymity algorithms employ the concept of a generalization lattice to compute an anonymous table. A generalization lattice over attribute domain generalization hierarchies is constructed using a set of all possible combinations of the generalization levels of each attribute (Figure 3). Then, an optimal *k*-anonymized table is computed by traversing the generalization lattice in a bottom-up manner until the *k*-anonymity property is satisfied. See [2] for a more detailed description of the *k*-anonymity algorithm.

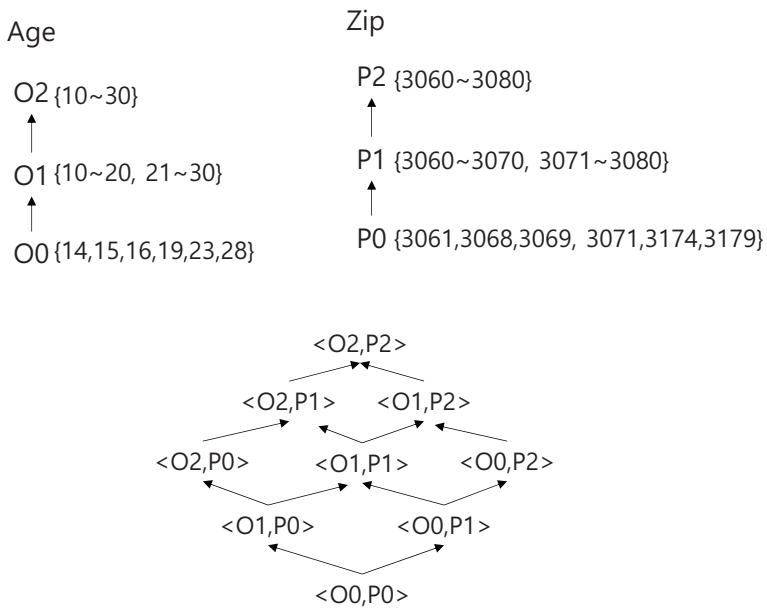


Figure 3. Example domain generalization hierarchies for *Age* and *Zipcode*, and the corresponding generalization lattice.

3.2. Problem Statement

Let us assume the relation $R(A_1, A_2, \cdots, A_m)$. Let $A = \{A_1, A_2, \cdots, A_m\}$ be a set of attributes in R. Let us further assume that $Q \in A$ is a set of quasi-identifier attributes and $S \in A$ is a set of sensitive attributes. In this paper, we then focus on a selection query on a single relation, which can be written as follows:

```
SELECT proj_list
FROM   R
WHERE  pred_1 AND pred_2 AND ··· AND pred_l.
```

Here, we further assume the following:

- *proj_list* consists of attributes that are in either Q or S,
- the condition in the `WHERE` clause consists of l conjunctive selection conditions, $pred_1$, $pred_2, \cdots, pred_l$, and
- each predicate $pred_i$ can be either an equality condition or a range condition on an attribute not in *proj_list*.

In this paper, we consider a scenario in which the results of a query are anonymized using k-anonymity with generalization, which is the most popular type of scheme.

We assume a distributed system running on a shared nothing architecture in which the data are horizontally split and stored in multiple nodes. In a shared nothing architecture, each node has its own private resources, such as memory or disk, and thus does not share resources with other nodes. In a distributed environment, given a selection query on a single relation, every slave node executes the query against its own data and sends the query result to the master node. The master node then aggregates all query results from all nodes and sends them to the user. In this distributed query processing scenario, none of the slave nodes can locally anonymize their own query results because the generalization level of each quasi-identifier attribute used to satisfy the k-anonymity requirement cannot be determined without combining all of the results from each of the slave nodes. Hence, the straightforward solution is to first aggregate the results from every slave node and then globally anonymize the aggregated results at the master node (Figure 4a). However, considering the large volume of data stored in the system, this global approach is highly inefficient because the data anonymization is performed solely by the master node. Furthermore, with a global approach, the resources of slave nodes are not utilized during the anonymization phase.

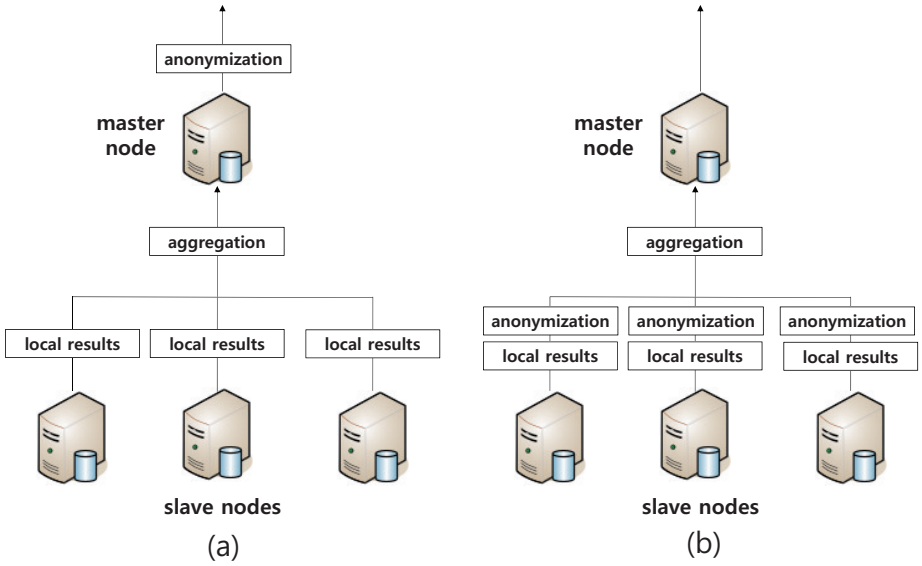


Figure 4. (**a**) Global anonymization vs. (**b**) local anonymization in a distributed environment.

A more promising solution is for each node to locally anonymize its own results and send the anonymized results to the master node, which combines the anonymized results from every slave node (Figure 4b). Thus, in this study, we develop a method that enables the query result at each slave node to be locally anonymized as much as possible, thereby fully utilizing the resources of the slave nodes during the anonymization phase.

4. Efficient Support of Online Data Publishing

In this section, we describe the proposed algorithm for efficiently anonymizing the query results on the fly in a distributed environment to support online privacy-preserving data publishing. The proposed approach in this paper is summarized as follows (Figure 5):

1. First, given a query, the master node estimates the generalization level of each quasi-identifier attribute to satisfy the k-anonymity property over the query result datasets, and then send it to each slave node along with the user query (Section 4.1);
2. Each slave node then executes the user query, anonymizes its own query results based on the generalization information received from the master node, and sends the anonymized query results to the master node (Section 4.2);
3. Finally, the master node aggregates the anonymized query results from every slave node and returns the aggregated results to the user (Section 4.3).

It is well known that k-anonymity algorithms are generally computationally expensive and complex, making them difficult to perform well with large amounts of data [43]. Thus, several approximation methods requiring a trade-off between data utility and computing time have been proposed [44–47]. We also note that the approach proposed in this study is an approximation-based algorithm in that it trades off between data utility and computing time. We will now describe each of the above three steps in detail.

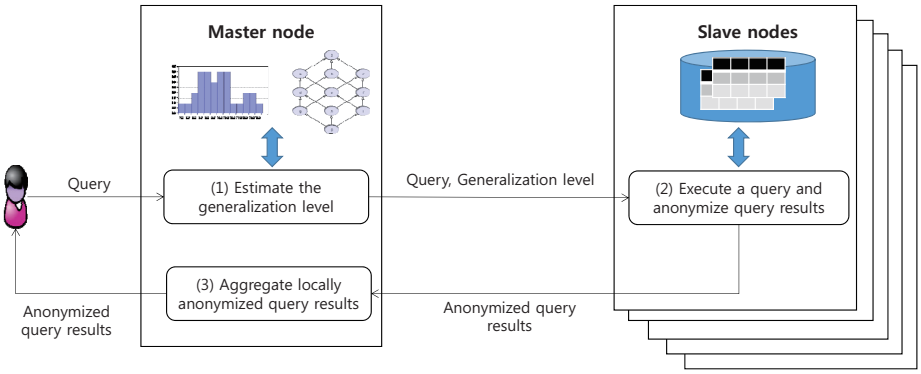


Figure 5. Overview of the proposed approach.

4.1. Phase I: Estimating the Generalization Level

In this paper, we estimate the generalization level of each quasi-identifier attribute to achieve k-anonymity over the query results by leveraging the statistical information, such as the histograms that are maintained for query optimization purposes in most commercial database management systems. In general, a histogram on attribute A_i is constructed by dividing the entire value range of A_i into w disjointed subranges, $H(A_i) = \{x_1, x_2, \cdots, x_w\}$. Each subrange, x_j, usually stores x_j^s, x_j^e, f_j, and dv_j. Here, x_j^s and x_j^e represent the start point and the end point of the subrange x_j, respectively. Furthermore, f_j corresponds to the number of tuples whose A_i values lie between x_j^s and x_j^e, and dv_j represents the number of distinct values in x_j.

Given a query, we first estimate the size of the query result. Estimating the query result size, which is known as a cardinality estimation, has been extensively studied over the past several decades [48–53]. Although there are many complex algorithms that can provide a very high level of accuracy in a cardinality estimation, our approach uses a solution based on the assumption of attribute value independence. The cardinality estimation in many database management systems indeed relies on a method based on an attribute value independence assumption owing to its simplicity and reasonably good accuracy. For example, PostgreSQL [54], which is a well-known open-source DBMS, assumes that all attributes are mutually independent and maintains one-dimensional histograms [53].

Given a histogram $H(A_i) = \{x_1, x_2, \cdots, x_w\}$ and predicate $pred_u$ involving attribute A_i, let s_u be the selectivity ratio associated with $pred_u$. The overall distribution of entire

values of the attribute A_i can be captured by using the histogram $H(A_i)$. Furthermore, we assume that attribute values in each subrange of $H(A_i)$ are uniformly distributed, which is a common assumption in modern database systems [53]. Then, the selectivity ratio s_u is obtained as follows: For the equality predicate (i.e., $\sigma_{A_i=val}(R)$, where val denotes any integer value located within the range between x_j^s and x_j^e), s_u is defined as:

$$s_u = \frac{f_j}{|R|} \times \frac{1}{dv_j}.$$

Here, $|R|$ is the number of tuples in the relation R. For the range predicate (i.e., $\sigma_{val_1 \le A_i \le val_2}(R)$, where val_1 and val_2 are any integer values located within the range of x_j^s and x_j^e), s_u is defined as:

$$s_u = \frac{f_j}{|R|} \times \frac{val_2 - val_1}{x_j^e - x_j^s}.$$

Note that the above equation considers the case in which val_1 and val_2 are located within the same subrange x_j. Let assume the case where val_1 and val_2 are located in different subranges, x_j and x_{j+k}, respectively. In this case, subranges, $x_{j+1}, x_{j+2}, \cdots, x_{j+k-1}$, are fully covered by the range predicate, while subranges, x_j and x_{j+k}, are partially covered. Thus, in this case, the selectivity ratio s_u is computed as:

$$s_u = \left(\frac{f_j}{|R|} \times \frac{x_j^e - val_1}{x_j^e - x_j^s}\right) + \left(\sum_{c=j+1}^{j+k-1} \frac{f_c}{|R|}\right) + \left(\frac{f_{j+k}}{|R|} \times \frac{val_2 - x_{j+k}^s}{x_{j+k}^e - x_{j+k}^s}\right).$$

Given a query having l predicates, $pred_1, pred_2, \cdots, pred_l$, let R_{res} be the corresponding result relation. Then, given selectivity ratios, $s_1, s_2, \cdots, s_l$, computed as explained previously, the query result size is estimated as:

$$|R_{res}| = |R| \times s_1 \times s_2 \times \cdots \times s_l.$$

That is, based on the assumption of attribute value independence, the query result size is computed by the product of all selectivity ratios, $s_1, s_2, \cdots, s_l$.

Once the number of query results is computed, we next estimate the generalization level to achieve the k-anonymity property over the query results. Given a projection list, $proj\_list$, of a query, let $Q_{proj} = \{A'_1, A'_2, \cdots, A'_y\}$ be the set of quasi-identifier attributes in $proj\_list$ (where $Q_{proj} \subset Q$). Let us further assume that $L(N, E)$ be a generalization lattice constructed with the attributes in Q_{proj}, where N and E are the set of nodes and edges, respectively. The set of possible values for the quasi-identifier attributes in Q_{proj} at the specific node $n_i \in N$ is then defined as follows:

$$EQ_{n_i} = \{(v_1, v_2, \cdots, v_y) \mid v_1 \in V_{A'_1},\ v_2 \in V_{A'_2}, \ldots, v_y \in V_{A'_y}\}.$$

Here, $V_{A'_t}$ $(1 \le t \le y)$ is the set of possible values for the quasi-identifier attribute, A'_t, at node n_i. Note that each possible value combination in EQ_{n_i} indeed corresponds to an equivalence class in the k-anonymity algorithm.

Example 1. *Let us consider the example shown in Figure 3, where $Q_{proj} = \{Age, Zip\}$. Let n_5 be a node associated with $\langle O_1, P_1 \rangle$. The set of possible values for the quasi-identifier attributes, Age and Zip, at n_5 is as follows:*

$$V_{Age} = \{10\text{–}20,\ 21\text{–}30\}, V_{Zip} = \{3060\text{–}3070,\ 3071\text{–}3080\}.$$

Then, EQ_{n_5} is computed as follows:

$$EQ_{n_5} = \{(10\text{–}20, 3060\text{–}3070),\ (10\text{–}20, 3071\text{–}3080),\ (21\text{–}30, 3060\text{–}3070),\ (21\text{–}30, 3071\text{–}3087)\}.$$

Each element in EQ_{n_5} corresponds to an equivalence class at the node n_5 in the generalization lattice.

Given a result relation, R_{res}, and a node, n_i, in a generalization lattice, let $R_{res}[v_1, v_2, \cdots, v_y]$ be an equivalence class whose values correspond to $(v_1, v_2, \cdots, v_y) \in EQ_{n_i}$ at node n_i. The size of an equivalence class is then estimated as:

$$|R_{res}[v_1, v_2, \cdots, v_y]| = |R_{res}| \times s_{\langle A'_1 = v_1 \rangle} \times s_{\langle A'_2 = v_2 \rangle} \times \cdots \times s_{\langle A'_y = v_y \rangle}.$$

Here, $s_{\langle A'_t = v_t \rangle}$ $(1 \leq t \leq y)$ is the selectivity ratio associated with the quasi-identifier attributes A'_t and the value v_t. Note that $s_{\langle A'_t = v_t \rangle}$ is estimated by leveraging a histogram, $H(A'_t)$.

Example 2. *Let us continue with the example in Figure 3. $|R_{res}[(10–20, 3071–3080)]|$ is computed as:*

$$|R_{res}[10–20, 3071–3080]| = |R_{res}| \times s_{\langle Age = 10–20 \rangle} \times s_{\langle Zip = 3071–3080 \rangle}.$$

Here, $s_{\langle Age = 10–20 \rangle}$ corresponds to the selectivity ratio associated with $\sigma_{10 \leq Age \leq 20}(R)$, which is computed using a histogram, $H(Age)$, as described earlier. Similarly, $s_{\langle Zip = 3071–3080 \rangle}$ can be computed using a histogram $H(Zip)$.

K-anonymity is achieved if each equivalence class contains at least *k*-tuples. Thus, given node n_i in a generalization lattice, our approach checks whether the equivalence class having the minimum size satisfies the *k*-anonymity property as follows:

$$\min_{(v_1, v_2, \cdots, v_y) \in EQ_{n_i}} |R_{res}[v_1, v_2, \cdots, v_y]| \geq k.$$

Thus, our approach traverses each node of a generalization lattice in a bottom-up manner, such as in [2], until a node that satisfies the above equation is found. Our approach is similar to the algorithm in [2] in that the generalization lattice is traversed in a bottom-up manner. However, it should be noted that, unlike the method in [2], our approach estimates the generalization level of each quasi-identifier attribute for *k*-anonymity based on the estimation method presented in this subsection, instead of performing *k*-anonymity on actual datasets.

4.2. Phase II: Executing a Query and Anonymizing Local Query Results

Upon receiving the user query from the master node, each slave node executes the received query over its local data collections and applies anonymization to the query results according to the generalization information received from the master node. It then returns the anonymized results to the master node. We note that this phase is executed in parallel by the salve nodes, which leads to the resources of slave nodes to be fully utilizied during the anonymization phase.

4.3. Phase III: Aggregating (and Further Anonymizing) Locally Anonymized Results

In the final phase, the master node aggregates the anonymized results from every slave node. Because the method proposed in this paper estimates the generalization level of each quasi-identifier attribute based on the histograms, the aggregated results may not satisfy the *k*-anonymity requirement when either an under- or overestimation occurs. An underestimation corresponds to a case in which a node in the generalization lattice, which is estimated by the algorithm described in Section 4.1, is located before the set of nodes in the traversing order of the lattice node, where *k*-anonymity with minimal generalizations is achieved. Similarly, an overestimation is defined as a case in which an estimated node in

a generalization lattice is located after the set of nodes in the traversing order of the lattice node, where *k*-anonymity with minimal generalizations is satisfied.

For example, in Figure 6a, let us assume that *k*-anonymity with minimal generalizations is achieved with the node $\langle O_0, P_2 \rangle$, which is highlighted with the red oval. Furthermore, let assume that the generalization lattice is traversed in a bottom-up manner and nodes in the same label are traversed from left to right. In this example, the nodes $\langle O_2, P_0 \rangle$, which are estimated using the algorithm described in Section 4.1, correspond to an underestimation case. On the other hand, the estimated node $\langle O_2, P_1 \rangle$ is an overestimation case.

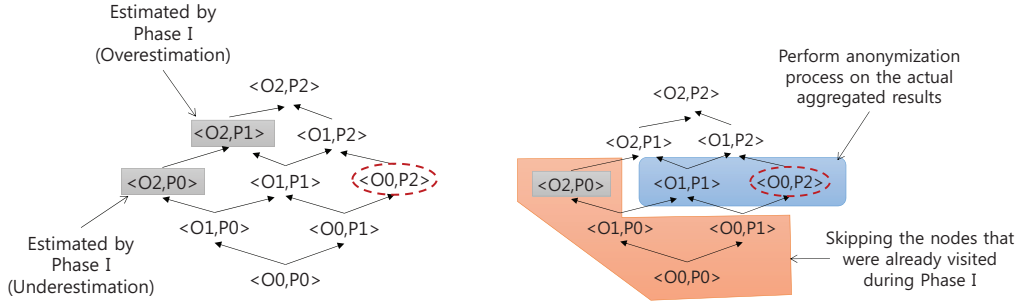


Figure 6. (**a**) Example of an underestimation and an overestimation of Phase I, in which we assume that *k*-anonymity with minimal generalizations is achieved with the node $\langle O_0, P_2 \rangle$, and (**b**) for the case of an underestimation, the master node may apply the anonymization process on the actual aggregated results (blue area), skipping the nodes that were already visited during Phase I (red area).

Hence, after aggregating the anonymized results from every slave node, the master node needs to check whether *k*-anonymity is satisfied over the aggregated results. If so, the aggregated results are returned to the user. However, if *k*-anonymity is not satisfied owing to an underestimation of the generalization level of each quasi-identifier attribute, the master node needs to conduct further anonymization of the aggregated results until *k*-anonymity is satisfied. It should be noted that, even in such a situation, the proposed method is more efficient than the baseline approach (i.e., the global approach shown in Figure 4a), because the nodes in the generalization lattice that were already visited during the generalization estimation phase in Section 4.1 can be skipped during the anonymization process of the master node. For example, consider the underestimation example in Figure 6b, in which the algorithm described in Section 4.1 estimates that the *k*-anonymity requirement is satisfied with the node $\langle O_2, P_0 \rangle$, even though in reality it is not. In this case, the master node conducts the anonymization process on the actual aggregated results, starting from the node $\langle O_1, P_1 \rangle$, and thus skips the nodes that were already visited during Phase I. This anonymization process continues until *k*-anonymity is satisfied. In the example in Figure 6b, the anonymization process stops at the nodes $\langle O_0, P_2 \rangle$, where *k*-anonymity is satisfied.

By contrast, an overestimation causes a loss of information of the released microdata because the quasi-identifier attributes are more generalized than necessary. With the anonymized results received from the slave nodes, the master node cannot detect whether an overestimation actually occurs, which results in returning more coarse-grained *k*-anonymity results to the user. This, in turn, leads to a loss in the data utility of the released microdata. That is, the algorithm proposed in this paper achieves a high level of efficiency in terms of applying *k*-anonymity by trading information loss with efficiency. However, as described in the experiment section, the proposed approach does not cause a significant reduction in the information on the released microdata, despite the occurrence of an overestimation, while achieving a high level of efficiency.

5. Experiment Evaluations

In this section, we describe the experimental evaluation of the performance of the proposed approach. First, we describe the experimental setup and then discuss the results.

5.1. Experiment Setup

To evaluate the proposed approach, we used the NPS dataset from the Health Insurance Review and Assessment (HIRA) service in Korea [55]. The National Patients Sample (NPS) dataset consists of electronic health records of 3% of the Korean people sampled in 2011. We randomly selected 5 M records with seven attributes (*Age*, *Sex*, *Length* of stay in the hospital, *Location*, *Surgery* status, *Disease*, and *Height* of patience) from the NPS dataset. We consider the first five attributes (*Age*, *Sex*, *Length*, *Location*, *Surgery*) to belong to QA, and the disease attribute belonging to S. In the experiments, we focused on the following range query:

```
SELECT Age, Sex, Length, Location, Surgery, Disease
FROM  R
WHERE min_height ≤ Height AND Height ≤ max_height.
```

Here, the values of min_{height} and max_{height} were varied during the experiments. In addition to reporting the experimental results for the method proposed in this paper (which is based on local anonymization in a distributed environment), we also report the results for the k-anonymity algorithm that is based on global anonymization in a distributed environment.

One way to evaluate the performance of the proposed approach is to implement the proposed scheme on commercial or open source distributed DBMSs and conduct comprehensive experiments in real application environments. This, however, is out of scope at this stage of the research. Thus, in this paper, we simulated a distributed query processing environment as following: we used a cluster with one master node and five slave nodes for the experiments. Each node has a 3.30 GHz of CPU. 1 Gbps LAN is used for node communication. The data used in the experiment is horizontally partitioned into five fragments which are distributedly stored in the five slave nodes. That is, each slave node has a relation with the same set of attribute (i.e., *Age*, *Sex*, *Length*, *Location*, *Surgery*, *Disease*, and *Height*) and records are randomly and evenly distributed among five slave nodes. Each slave node has its local (standalone) DBMS, MySQL [56], managing local data. The communication between a master and a slave node is implemented using standard TCP/IP. Upon receiving a user query, the master node sends it to slave nodes which run in parallel. Then, each slave node runs the user query against the local data, and returns query results to the master node. We ran each query five times and the averaged values are presented in the paper.

5.2. Results and Discussion

Figure 7 shows the execution times for varying the number of query results. During the experiments, various numbers of query results were obtained by controlling the values of min_{height} and max_{height}. The key observations, based on Figure 7, can be summarized as follows: the proposed method (*Est_kAnonymity* that is based on local anonymization in a distributed environment) significantly outperforms the original k-anonymity algorithm (*kAnonymity* that is based on global anonymization in a distributed environment) in terms of the execution time. As the number of results increases, the performance gap between *Est_kAnonymity* and *kAnonymity* increases. Figure 7 also shows whether the generalization level of each quasi-identifier attribute for k-anonymity is correctly estimated (marked with 'C' in the figure), or under- or overestimated (marked with a 'U' or 'O', respectively, in the figure) by the estimation method described in Section 4.1. As can be seen in Figure 7, the underestimation causes a slight increase in the execution time because the anonymization is applied on the actual query result dataset by the master node.

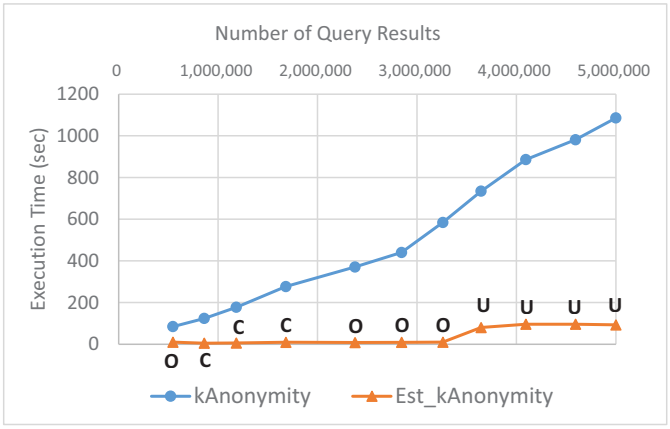
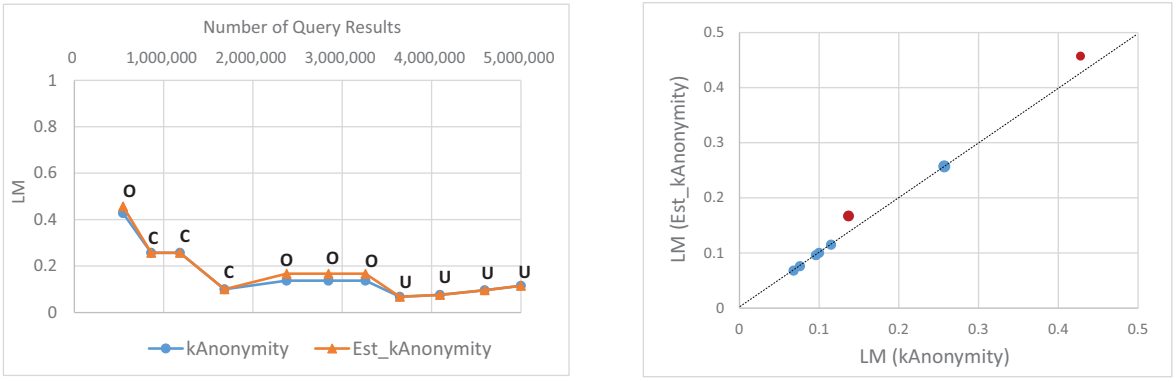


Figure 7. The execution times when varying the number of query results.

Figure 8 plots the loss metric (LM) [57] for varying the number of query results. Note that LM measures the amount of information that is lost due to a generalization of the quasi-identifier attributes, ranging from zero to one (a lower value is better). The proposed method, *Est_kAnonymity*, shows a very similar pattern with *kAnonymity* in terms of the LM. As can be seen in Figure 8a, the LM slightly increases with the proposed approach. In particular, the increases are observed when the generalization level is overestimated. This is because the overestimation causes the values of the quasi-identifier attributes to be more generalized than needed, which results in the increased LM. However, the underestimation does not lead to an increase in the LM, the reason for which is that, in the case of an underestimation, the values of the quasi-identifier attributes are less generalized, and thus the master node applies further anonymization on the actual aggregated results until *k*-anonymity is satisfied, which corresponds to Phase III in Section 4.3.

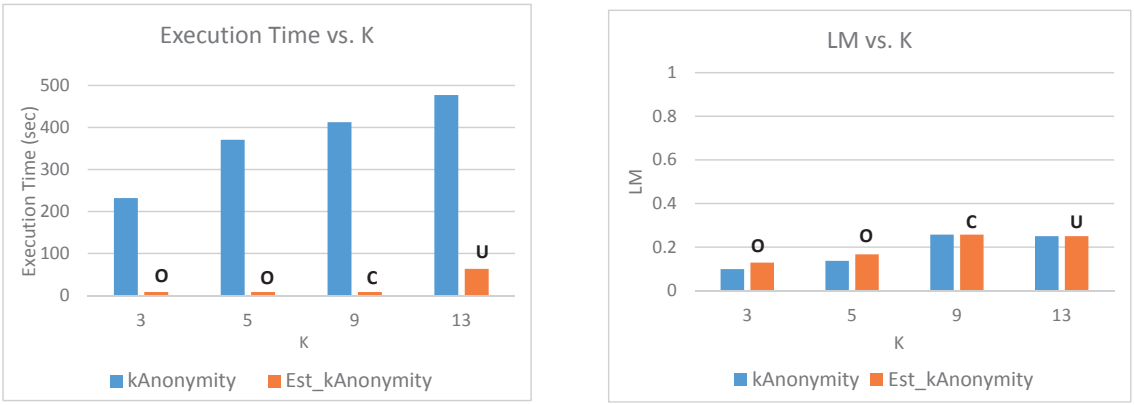


(**a**) (**b**)

Figure 8. (**a**) LM when varying the number of query results, and (**b**) comparison of LM between *kAnonymity* and *Est_kAnonymity*.

To further compare the LM between *kAnonymity* and *Est_kAnonymity*, we plot the LM results in Figure 8b, where the x-axis represents the LM quantity for *kAnonymity* and the y-axis represents the LM quantity for *Est_kAnonymity*. Here, the red circles correspond to the overestimation cases, whereas the blue circles represent either the underestimation or corrected estimation cases. As can be seen in the figure, in most cases, the circles are located on the dotted diagonal line, which indicates that there is no loss in data utility with the proposed approach. Even under the occurrence of an overestimation, it is observed that the red circles are closely located on the diagonal line, indicating that the proposed approach does not cause a significant reduction in the information of the released microdata.

Figure 9 shows (a) the execution times and (b) loss metric (LM) for varying values of k. During the experiments, the values of k varied among 3, 5, 9, and 13, and the values of min_{height} and max_{height} were set such that the number of query results was about 2.3 M. Key observations based on Figure 9 can be summarized as follows: as expected, the proposed method ($Est\_kAnonymity$) significantly outperforms the original k-anonymity algorithm ($kAnonymity$) in terms of the execution time. The performance gaps between $Est\_kAnonymity$ and $kAnonymity$ increase as the value of k increases. The figure also shows whether the generalization level for the k-anonymity is correctly estimated (i.e., 'C', 'U', and 'O'). Once again, as shown in Figure 9b, the LM is slightly increased with the proposed approach, particularly when the generalization level is overestimated (i.e., k = 3, 5).



(**a**) (**b**)

Figure 9. (**a**) The execution times and (**b**) loss metric (LM) when varying k.

Finally, Figure 10 shows the way that the execution time (shown in Figure 9a) is split among three phases: (1) estimating the generalization level (Phase I); (2) executing a query and anonymizing query results (Phase II); and (3) aggregating (and further anonymizing) locally anonymized query results (Phase III). As can be seen in the figure, Phase III, which corresponds to aggregating (and further anonymizing) locally anonymized query results, has been identified as a major contributor to the execution time for all the cases. Especially, a significant increase in the execution of the Phase III is observed, when the generalization level is underestimated (i.e., k = 13). This is because, in the case of an underestimation, the master node should perform further anonymization on the aggregated results until the k-anonymity is satisfied, which causes a significant increase in the execution time of the Phase III. Note that Phases I and III are applied by the master node, whereas Phase II is executed by each slave node. Thus, the distributed nature of the presented algorithm affects the execution time of Phase II. That is, if more slave nodes are used, the execution time of Phase 2 will be reduced.

The experimental results in this section verify that, with the proposed method, significant processing time gains can be achieved without a significant reduction in the information on the released mircrodata.

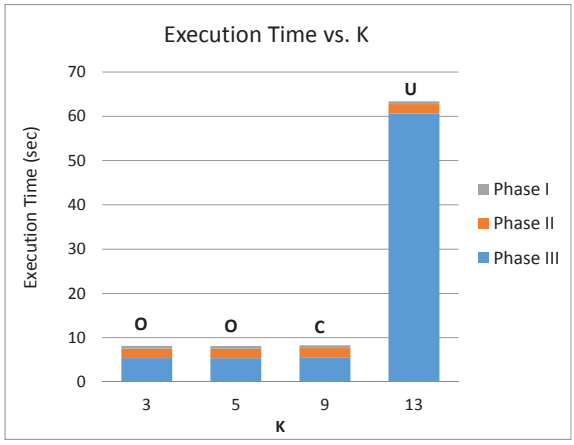


Figure 10. The way that the execution time is split among three phases when varying *k*.

6. Conclusions

Most existing privacy-preserving data publishing algorithms consider an offline data publishing scenario in which the data publisher first anonymizes the data in an offline manner, and then releases the anonymized data for public use. However, with the increasing demand for the sharing of microdata among various parties, an offline privacy-preserving data publishing scenario is insufficient to support the voluminous request for a release of data. Instead, it is more desirable to integrate the data anonymization functionality into existing systems that are capable of supporting online query processing. In this paper, with the aim of supporting efficient online privacy-preserving data publishing, we presented a novel scheme that is able to efficiently anonymize the query results on the fly. In particular, given a user's query, the proposed approach effectively estimates the generalization level of each attribute for achieving the *k*-anonymity property in the query result datasets based on the statistical information. The proposed algorithm achieves a high level of efficiency in applying *k*-anonymity by effectively sacrificing the information loss of the released microdata. The experimental results when applying a real dataset show that significant processing time gains can be achieved with the proposed method, while avoiding a significant reduction of information on the released mircrodata. Future work will include an investigation into the various types of queries containing complex operations, such as a join or aggregation.

Funding: This research was funded by a 2021 research Grant from Sangmyung University.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Sweeney, L. *k*-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **2002**, *10*, 557–570. [CrossRef]
2. LeFevre, K.; DeWitt, D.J.; Ramakrishnan, R. Incognito: Efficient full domain *k*-anonymity. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, MD, USA, 14–16 June 2005.
3. Machanavajjhala, A.; Kifer, D.; Gehrke, J.; Venkitasubramaniam, M. *l*-diversity: Privacy beyond *k*-anonymity. *ACM Trans. Knowl. Discov. Data* **2007**, *1*, 3-es. [CrossRef]
4. Li, N.; Li, T.; Venkatasubramanian, S. *t*-closeness: Privacy beyond kanonymity and *l*-diversity. In Proceedings of the International Conference on Data Engineering, Istanbul, Turkey, 15–20 April 2007.
5. Biskup, J.; Bonatti, P.A. Controlled query evaluation for known policies by combining lying and refusal. In Proceedings of the International Symposium on Foundations of Information and Knowledge Systems, Salzau Castle, Germany, 20–23 February 2002.
6. Kenthapadi, K.; Mishra, N.; Nissim, K. Simulatable auditing. In Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Baltimore, MD, USA, 14–16 June 2005.
7. Nabar, S.U.; Marthi, B.; Kenthapadi, K.; Mishra, N.; Motwani, R. Towards robustness in query auditing. In Proceedings of the International Conference on Very Large Data Bases, Seoul, Korea, 12–15 September 2006.
8. Katsomallos, M.; Tzompanaki, K.; Kotzinos, D. Privacy, space and time: A survey on privacy-preserving continuous data publishing. *J. Spat. Inf. Sci.* **2019**, *19*, 57–103. [CrossRef]

9. Wang, K.; Fung, B.C.M. Anonymizing sequential releases. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20–23 August 2006.
10. Fung, B.C.M.; Wang, K.; Fu, A.; Pei, J. Anonymity for continuous data publishing. In Proceedings of the International Conference on Extending Database Technology, Nantes, France, 25–29 March 2008.
11. Xiao, X.; Tao, Y. M-invariance: Towards privacy preserving re-publication of dynamic data sets. In Proceedings of the ACM SIGMOD international conference on Management of Data, Beijing, China, 12–14 June 2007.
12. He, Y.; Barman, S.; Naughtoni, J.F. Preventing equivalence attacks in updated, anonymized data. In Proceedings of the IEEE International Conference on Data Engineering, Hannover, Germany, 11–16 April 2011.
13. Wang, K.; Yu, P.S.; Chakraborty, S. Bottom-up generalization: A data mining solution to privacy protection. In Proceedings of the IEEE International Conference on Data Mining, Brighton, UK, 1–4 November 2004.
14. Fung, B.C.M.; Wang, K.; Yu, P.S. Top-down specialization for information and privacy preservation. In Proceedings of the IEEE International Conference on Data Engineering, Tokyo, Japan, 5–8 April 2005.
15. LeFevre, K.; DeWitt, D.J.; Ramakrishnan, R. Mondrian multidimensional *k*-anonymity. In Proceedings of the IEEE International Conference on Data Engineering, Atlanta, GA, USA, 3–7 April 2006.
16. Byun, J.W.; Kamra, A.; Bertino, E.; Li, N. Efficient *k*-anonymization using clustering technique. In *Advances in Databases: Concepts, Systems and Applications*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 188–200.
17. Aggarwal, G.; Panigrahy, R.; Feder, T.; Thomas, D.; Kenthapadi, K.; Khuller, S.; Zhu, A. Achieving anonymity via clustering. *ACM Trans. Algorithms* **2010**, *6*, 1–19. [CrossRef]
18. Sun, X.; Sun, L.; Wang, H. Extended *k*-anonymity models against sensitive attribute disclosure. *Comput. Commun.* **2011**, *34*, 526–535. [CrossRef]
19. Anjum, A.; Malik, S.U.R.; Choo, K.-K.R.; Khan, A.; Haroon, A.; Khan, S.; Khan, S.U.; Ahmad, N.; Raza, B. An efficient privacy mechanism for electronic health records. *Comput. Secur.* **2018**, *72*, 196–211. [CrossRef]
20. Kanwal, T.; Anjum, A.; Malik, S.U.R.; Sajjad, H.; Khan, A.; Manzoor, U.; Asheralieva, A. A robust privacy preserving approach for electronic health records using multiple dataset with multiple sensitive attributes. *Comput. Secur.* **2021**, *105*, 102224. [CrossRef]
21. Kim, S.; Sung, M.K.; Chung, Y.D. A framework to preserve the privacy of electronic health data streams. *J. Biomed. Inform.* **2014**, *50*, 95–106. [CrossRef] [PubMed]
22. Lee, H.; Kim, S.; Kim, J.W.; Chung, Y.D. Utility-preserving anonymization for health data publishing. *BMC Med. Inform. Decis. Mak.* **2017**, *17*, 104. [CrossRef]
23. Khan, R.; Tao, X.; Anjum, A.; Kanwal, T.; Malik, S.R.; Khan, A.; Rehman, W.; Maple, C. θ-Sensitive *k*-Anonymity: An anonymization model for IoT based electronic health records. *Electronics* **2020**, *9*, 716. [CrossRef]
24. Fung, B.C.M.; Wang, K.; Chen, R.; Yu, P.S. Privacy-preserving data publishing: A survey of recent developments. *ACM Comput. Surv.* **2010**, *42*, 1–53. [CrossRef]
25. Mohammed, N.; Fung, B.C.M.; Hung, P.C.K.; Lee, C.K. Centralized and distributed anonymization for high-dimensional healthcare data. *ACM Trans. Knowl. Discov. Data* **2010**, *4*, 1–33. [CrossRef]
26. Gkoulalas-Divanis, A.; Loukides, G.; Sun, J. Publishing data from electronic health records while preserving privacy: A survey of algorithms. *J. Biomed. Inform.* **2014**, *50*, 4–19. [CrossRef]
27. Abdelhameed, S.A.; Moussa, S.M.; Khalifa, M.E. Privacy-preserving tabular data publishing: A comprehensive evaluation from web to cloud. *Comput. Secur.* **2018**, *72*, 74–95. [CrossRef]
28. Majeed, A.; Lee, S. Anonymization techniques for privacy preserving data publishing: A comprehensive survey. *IEEE Access* **2020**, *9*, 8512–8545. [CrossRef]
29. Zigomitros, A.; Casino, F.; Solanas, A.; Patsakis, C. A survey on privacy properties for data publishing of relational data. *IEEE Access* **2020**, *9*, 51071–51099. [CrossRef]
30. Dwork, C. Differential privacy. In Proceedings of the International Conference on Automata, Languages and Programming, Venice, Italy, 10–14 July 2006.
31. Li, H.; Xiong, L.; Zhang, L.; Jiang, X. DPSynthesizer: Differentially private data synthesizer for privacy preserving data sharing. In Proceedings of the VLDB Endowment, Hangzhou, China, 1–5 September 2014.
32. Xiao, X.; Bender, G.; Hay, M.; Gehrke, J. iReduct: Differential privacy with reduced relative errors. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Athens, Greece, 12–16 June 2014.
33. Erlingsson, U.; Pihur, V.; Korolova, A. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014; pp. 1054–1067.
34. Bassily, R.; Smith, A. Local, private, efficient protocols for succinct histograms. In Proceedings of the ACM Symposium on Theory of Computing, Portland, OR, USA, 13–15 June 2015; pp. 127–135.
35. Wang, N.; Xiao, X.; Yang, Y.; Hoang, T.D.; Shin, H.; Shin, J.; Yu, G. Privtrie: Effective frequent term discovery under local differential privacy. In Proceedings of the IEEE International Conference on Data Engineering, Paris, France, 16–19 April 2018; pp. 821–832.
36. Wang, T.; Li, N.; Jha, S. Locally differentially private heavy hitter identification. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 982–993. [CrossRef]

37. Andres, M.E.; Bordenabe, N.E.; Chatzikokolakis, K.; Palamidessi, C. Geo-indistinguishability: Differential privacy for location-based systems. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013; pp. 901–914.
38. Ahuja, R.; Ghinita, G.; Shahabi, C. A utility-preserving and scalable technique for protecting location data with geo-indistinguishability. In Proceedings of the International Conference on Extending Database Technology, Lisbon, Portuga, 26–29 March 2019; pp. 210–231.
39. Zhang, J.; Xiao, X.; Xie, X. Privtree: A differentially private algorithm for hierarchical decompositions. In Proceedings of the International Conference on Management of Data, San Francisco, CA, USA, 14–19 June 2016; pp. 155–170.
40. Kim, J.S.; Chung, Y.D.; Kim, J.W. Differentially private and skew-aware spatial decompositions for mobile crowdsensing. *Sensors* **2018**, *18*, 3696. [CrossRef] [PubMed]
41. Lee, H.; Chung, Y.D. Differentially private release of medical microdata: An efficient and practical approach for preserving informative attribute values. *BMC Med. Inform. Decis. Mak.* **2020**, *20*, 155. [CrossRef] [PubMed]
42. Guo, J.; Yang, M.; Wan, B. A practical privacy-preserving publishing mechanism based on personalized *k*-anonymity and temporal differential privacy for wearable IoT applications. *Symmetry* **2021**, *13*, 1043. [CrossRef]
43. Meyerson, A.; Williams, R. On the complexity of optimal *k*-anonymity. In Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Paris, France, 14–16 June 2004.
44. Park, H.; Shim, K. Approximate algorithms for *k*-anonymity. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, 12–14 June 2007.
45. Sun, X.; Li, M.; Wang, H.; Plank, A. An efficient hash-based algorithm for minimal *k*-anonymity. In Proceedings of the Australasian Conference on Computer Science, Wollongong, Australia, 22–25 January 2008.
46. Babu, K.S.; Reddy, N.; Kumar, N.; Elliot, M.; Jena, S.K. Achieving *k*-anonymity using improved greedy heuristics for very large relational databases. *Trans. Data Priv.* **2013**, *6*, 1–17.
47. Hernandez-Baigorri, D.R.C.; Forne, J.; Soriano, M. Incremental *k*-anonymous microaggregation in large-scale electronic surveys with optimized scheduling. *IEEE Access* **2018**, *6*, 60016–60044.
48. Chaudhuri, S.; Motwani, R.; Narasayya, V. On Random Sampling over Joins. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, USA, 31 May–3 June 1999.
49. Chakrabarti, K.; Garofalakis, M.; Rastogi, R.; Shim, K. Approximate query processing using wavelets. In Proceedings of the International Conference on Very Large Data Bases, Cairo, Egypt, 10–14 September 2000.
50. Babcock, B.; Chaudhuri, S. Towards a robust query optimizer: A principled and practical approach. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, MD, USA, 14–16 June 2005.
51. Spiegel, J.; Polyzotis, N. Graph-based synopses for relational selectivity estimation. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Chicago, IL, USA, 27–29 June 2006.
52. Kim, J.W.; Candan, K.S. PICC counting: Who needs joins when you can propagate efficiently? In Proceedings of the SIAM International Conference on Data Mining, Sparks, NV, USA, 30 April–2 May 2009.
53. Han, Y.; Wu, Z.; Wu, P.; Zhu, R.; Yang, J.; Tan, L.W.; Zeng, K.; Cong, G.; Qin, Y.; Pfadler, A.; et al. Cardinality Estimation in DBMS: A Comprehensive Benchmark Evaluation. *arXiv* **2021**, arXiv:2109.05877.
54. PostgreSQL: The World's Most Advanced Open Source Relational Database. 2021. Available online: https://www.postgresql.org/ (accessed on 5 November 2021).
55. Health Insurance Review and Assessment Service in Korea. 2012. Available online: http://opendata.hira.or.kr (accessed on 5 November 2021).
56. MySQL. 2017. Available online: https://www.mysql.com/ (accessed on 5 November 2021).
57. Iyengar, V.S. Transforming data to satisfy privacy constraints. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, AB, Canada, 23–26 July 2002.