Query-Based ℓ-Diversity

Chittaphone Phonharath
Nara Institute of Science
and Technology
Nara, Japan
Email: chittaphone-p@is.naist.jp

Ryunosuke Takayama, Kenji Hashimoto, Hiroyuki Seki
Nagoya University
Nagoya, Japan
Email: {k-hasimt,seki}@is.nagoya-u.ac.jp

***Abstract*—We propose a new privacy notion called *query-based ℓ-diversity*. A database instance T is ℓ-diverse with respect to given authorized queries if an attacker cannot narrow down the number of possible values of the sensitive information to less than ℓ by inference using the result of the authorized queries on the instance T and the meaning of the queries. We provide two approaches to deciding the query-based ℓ-diversity. In the first approach, a decision algorithm is given by using relational operations, which can be directly implemented by a relational database management system, e.g., Structured Query Language (SQL). The second approach transforms a given input to a logical formula and decides the problem by model counting using a #SAT solver. We discuss the effectiveness and scalability of the two approaches based on the experimental results.**

Keywords–Database Privacy; Diversity; Inference Attack; Relational Databases.

I. INTRODUCTION

Database security is one of the most important challenges to minimize the leakage of the sensitive information over the accesses or the data publishing, which have been growing rapidly and more powerful. Access control is a traditional mechanism for confidentially restricting accesses to a database made by a user by dividing the queries into authorized and unauthorized ones, and restricting the portion of the data that can be retrieved and updated by the user.

Inference attack is a malicious way to infer the sensitive information protected by access control. An attack is conducted by combining the result of authorized queries, the code (the meaning) of the queries and other available external information to obtain the candidate values of the sensitive information, as shown in Example 1. Thus, we need an appropriate quantitative notion of the security of a database against inference attacks.

Example 1: Table I shows a database instance consisting of six tuples. Assume that {Zipcode, Gender, Age} is the quasi-identifier and Diagnosis is the sensitive attribute. Assume that query (1) extracts tuples of {Zipcode, Age} where "Age" ≤ 60 from Table I, as shown in Table II, and query (2) extracts tuples of {Age, Diagnosis}, as shown in Table III. If we combine two results of queries (1) and (2), we can obtain candidate set of the type of "Diagnosis", as shown in Table IV. This cannot guarantee that this database instance is safe against inferencing.

As described in Related Work, there are well-known related notions such as k-anonymity and ℓ-diversity. Intuitively, a database is k-anonymous if for every individual x, there are at least k different records (or tuples in the relational database setting) which cannot be distinguished from the real record for x. A database is ℓ-diverse if for every individual x, there are at least ℓ different values of the sensitive information contained in the records which cannot be distinguished from the real record for x. Also, various methods are reported for transforming a given database into a database satisfying k-anonymity or ℓ-diversity. However, these notions do not take the effect of access control for queries into consideration.

TABLE I. A SAMPLE INSTANCE.

Zipcode	Gender	Age	Diagnosis
123-4567	F	45	A
123-5235	F	44	B
123-4567	F	44	C
378-2102	M	65	A
378-2102	M	62	B
378-2102	F	65	A

TABLE II. A RESULT OF QUERY (1).

Zipcode	Age
123-4567	45
123-5235	44
123-4567	44

TABLE III. A RESULT OF QUERY (2).

Age	Diagnosis
45	A
44	B
44	C

TABLE IV. A RESULT OF QUERIES (1) AND (2).

Zipcode	Age	Diagnosis
123-4567	45	A
123-5235	44	B
123-5235	44	C
123-4567	44	B
123-4567	44	C

The goal of this study is to introduce a notion of the security against inference attack by extending ℓ-diversity in relational databases with access control for queries. More specifically, we propose a new privacy notion called *query-based ℓ-diversity*. A database instance T is ℓ-diverse with respect to given authorized queries if an attacker cannot narrow down the number of possible values of the sensitive information for any individual to less than ℓ by inference based on the result of the authorized queries on the instance T and the queries themselves. We provide two approaches to deciding the query-based ℓ-diversity. In the first approach, a decision algorithm is given by using relational operations, which can

 ISBN: 978-1-61208-408-4

be directly implemented by a relational database management system, e.g., SQL. The second approach transforms a given input to a logical formula and decides the problem by model counting using a #SAT solver. We discuss the effectiveness and scalability of the two approaches based on the experimental results.

A. Related Work

The anonymization technique enforces the preservation of privacy of personal data or sensitive information. Generalization of the data is one of the well-known techniques for anonymizing information accordingly with the domain generalization hierarchy from which the quasi-identifier value can be generalized such as numeric values are generalized to intervals. There are a few well-known notions for database privacy, k-anonymity [11][13], ℓ-diversity [8] and t-closeness [7]. These notions assume the following basic concepts on relational databases. The set of attributes are divided into sensitive and nonsensitive attributes. Also, a subset of the nonsensitive attributes, called the quasi-identifier, is assumed. The value of the quasi-identifier is potentially used to identify the tuple of a target individual by linking the disclosed information with external data.

k-anonymity A database instance satisfies k-anonymity if for any value of the quasi-identifier, there are k or more tuples having that value of the quasi-identifier. A maximal subset of tuples having same values of the quasi-identifier is called an equivalence class. k-anonymity means that the cardinality of each equivalence class is at least k. A transformation of a given instance to another instance satisfying k-anonymity is called a k-anonymization for the original instance. [13] proposed a method for k-anonymization by hiding some information of individuals by generalization and suppression. Generalization replaces a value with less specific but semantically consistent value. While suppression hides the data or does not release the entire data. Various anonymization methods have been reported using clustering, branch-and-bound search, and so on [1][2]. k-anonymity is a simple notion and has been frequently used. As discussed in [8], however, a k-anonymous database may still have some issues because the database may lack the diversity in the sensitive attributes. ℓ-diversity has been proposed by [8] to overcome the weakness of k-anonymity. Though [8] proposes a general definition of ℓ-diversity, we just review a simple and frequently used one, called distinctive non-recursive ℓ-diversity.

ℓ-diversity A database instance satisfies distinctive non-recursive ℓ-diversity (or simply, ℓ-diversity) if for each equivalence class, there are at least ℓ different values of the sensitive attributes.

Example 2: As shown in Table I, all tuples have different values of the quasi-identifier and hence this database instance does not satisfy k-anonymity for any $k \geq 2$. Assume that in this database instance, the lower four digits of the values of "Zipcode" are hidden, the values of "Gender" is hidden and the values of "Age" are generalized to the intervals of ten years. Then we obtain the database instance shown in Table V. This instance consists of two equivalence classes and the number of tuples in each class is three. Hence, this instance satisfies 3-anonymity and the above transformation is a 3-anonymization for the original instance. Also, the first class has three different values of the sensitive attribute and the second class has two different values. Hence, the transformed instance satisfies 2-diversity but does not satisfy ℓ-diversity for any $\ell \geq 3$.

TABLE V. A 3-ANONYMOUS AND 2-DIVERSE INSTANCE.

Zipcode	Gender	Age	Diagnosis
123-****	-	[40,49]	A
123-****	-	[40,49]	B
123-****	-	[40,49]	C
378-****	-	[60,69]	A
378-****	-	[60,69]	B
378-****	-	[60,69]	A

k-secrecy A related but different quantitative notion on database security is given in [5] based on access control on queries. Assume that a database instance T of a schema $\mathbf{R}$, authorized queries $q_1, q_2, \ldots, q_m$ and an unauthorized query q_U are given. An attacker knows $\mathbf{R}$, $q_1, q_2, \ldots, q_m, q_U$ (the meaning of the authorized and unauthorized queries) and $q_1(T), \ldots, q_m(T)$ (the result of the authorized queries), but he does not know T. The goal of the attacker is to obtain $q_U(T)$, which is the result of the unauthorized query q_U on T. For a positive integer k, a database instance T is k-secret with respect to $\mathbf{R}, q_1, \ldots, q_m, q_U$ if the attacker cannot narrow down the number of the candidates of $q_U(T)$ to less than k. T is ∞-secret if the candidates of $q_U(T)$ are infinite. We say that a database schema $\mathbf{R}$ is k-secret with respect to $q_1, \ldots, q_m, q_U$ if every database instance of $\mathbf{R}$ is k-secret. [5] showed that k-secrecy is decidable for XML databases where queries are given as tree transducers in a certain subclass that can use relabeling and deletion. Also, [10] showed that the problem for deciding whether a given XML schema is k-secret is undecidable for any finite k while the problem is decidable when $k = \infty$. Although [5] deals with XML databases, the notion of k-secrecy is general enough for other kinds of databases. More sophisticated notions have been also proposed. For example, [3][9] proposed stronger notions where the probability distribution of possible secrets does not change after observing (authorized) information. The notion of query-based ℓ-diversity proposed in this paper is a combination of k-secrecy and ℓ-diversity in the relational database setting.

The organization of this paper is as follows. Section II provides basic notions and notations on relational database that will be used in the paper. We define the query-based ℓ-diversity in Section III. In Section IV, a decision algorithm based on relational operations as the first approach is given. Section V provides the second approach based on model counting. Experimental results conducted on SQL for the first approach and using a #SAT solver sharpCDCL for the second approach are shown in Section VI. We conclude the paper in Section VII.

II. MODELS

In this section, we introduce a simple relational database model , which will be used in the rest of the paper. A relational database instance (or simply a database) can be seen as a *table*, of which columns are *attributes*. There are two types of attributes, namely *sensitive* and *nonsensitive* attributes. The values of sensitive attributes are considered as secret, that is, the data owner keeps them confidentially and restrictively and protects them from unauthorized accesses.

Definition 1: A *relational database schema* (or simply a schema) is a finite set of attributes. Let $\mathbf{R} = \{A_1, \ldots, A_n\}$ be

a schema. We assume that for each attribute A_i $(1 \leq i \leq n)$, a finite set of values, denoted by $dom(A_i)$ is associated. A *tuple* (or a record) over $\mathbf{R}$ is $t = (d_1, \ldots, d_n)$ where $d_i \in dom(A_i)$ for each $1 \leq i \leq n$. Let $t[A_i] = d_i$, which is called the value of attribute A_i in t. That is, $t = (t[A_1], \ldots, t[A_n])$. A *relational database instance* (or simply an instance) of $\mathbf{R}$ is a finite set of tuples over $\mathbf{R}$. An instance is sometimes called a *table*. Let $I[\mathbf{R}]$ denote the set of all instances of $\mathbf{R}$.

Let $\mathbf{R}$ be a schema. We assume that $\mathbf{R}$ is divided into two disjoint subsets, namely, $\mathcal{S}e$ and $\mathcal{NS}e$, which are the set of *sensitive attributes* and the set of *nonsensitive attributes*, respectively. We furthermore assume that a subset $Qi \subseteq \mathcal{NS}e$ of nonsensitive attributes is given as a *quasi-identifier* of $\mathbf{R}$. We intend that the values of the quasi-identifier can be potentially used to identify the values of the sensitive attributes by linking the attribute values of the quasi-identifier with external data sets.

We define projection, selection and join in the usual way. Let $\mathbf{R}$ be a schema. For a subset of attributes $\alpha = \{A_{j_1}, \ldots, A_{j_m}\} \subseteq \mathbf{R}$ and a tuple t over $\mathbf{R}$, let $\pi_\alpha(t)$ denote the tuple $(t[A_{j_1}], \ldots, t[A_{j_m}])$, which is called the projection of t on α. Also, for an instance $T \in I[\mathbf{R}]$, let $\pi_\alpha(T) = \{\pi_\alpha(t) \mid t \in T\}$. Let $T_1 \in I(\mathbf{R}_1)$ and $T_2 \in I(\mathbf{R}_2)$. For a filtering condition F, and an instance T, let $\sigma_F(T)$ denote the set of tuples in T that satisfy F. The natural join of T_1 and T_2 is the instance obtained by "linking" every possible pair of tuples in T_1 and T_2:

$$T_1 \bowtie T_2 = \{t \; over \; \mathbf{R}_1 \cup \mathbf{R}_2 \mid for \; some \; u \in T_1, \\ w \in T_2, \; t[U] = u \; and \; t[W] = w\}. \quad (1)$$

Since the natural join operator is associative and commutative, we sometimes view the natural join as a polyadic operator and write $T_1 \bowtie .... \bowtie T_m$.

III. Proposed Framework

In order to provide the definitions, we need to introduce the candidate set of instances of which results of queries are the same as those of the real instance. For a given instance T and queries $q_1, ..., q_m$, let $cand(q_1, ..., q_m, T)$ be the set consisting of all instances that give the same result as T with respect to all queries $q_1, ..., q_m$:

$$cand(\mathbf{R}, q_1, ..., q_m, T) = \{T' \in I(R) \mid \forall i(1 \leq i \leq m) \cdot \\ q_i(T) = q_i(T')\}. \quad (2)$$

Each $T' \in cand(\mathbf{R}, q_1, ..., q_m, T)$ is called a candidate instance.

Let T be a database instance over schema $\mathbf{R}$. T is query-based ℓ-diverse if for each maximal subset of a candidate instance of which tuples have the same quasi-identifier, there are ℓ or more different values of the sensitive attributes.

Suppose that the following information is available to public: a database schema $\mathbf{R}$, authorized queries $q_1, ..., q_m$, quasi-identifier Qi, sensitive attributes Se and a threshold ℓ (a positive integer). Let T be an instance of $\mathbf{R}$. An attacker infers sensitive information by taking the natural join of the results of the authorized queries $q_1, ..., q_m$ on the instance T to obtain the candidate set of sensitive information. We now show three options for the definition of query-based ℓ-diversity as follows.

Definition 2: An instance $T \in I(\mathbf{R})$ is l-diverse (with respect to $\mathbf{R}, Qi, Se, q_1, ..., q_m, T$)

(Option 1) if for every $t \in \pi_{Qi}(T)$,

$$|\{\pi_{Se}(t') \mid \; ^\exists T' \in cand(q_1, ..., q_m, T) \cdot \\ (\pi_{Qi}(t') = t \wedge t' \in T')\}| \geq l, \quad (3)$$

(Option 2) if for every $t \in \pi_{Qi}(T)$, there is an instance $T' \in cand(q_1, ..., q_m, T)$ such that

$$|\{\pi_{Se}(t') \mid (\pi_{Qi}(t') = t \wedge t' \in T')\}| \geq l, \quad (4)$$

(Option 3) if there is $T' \in cand(q_1, ..., q_m, T)$ such that for every $t \in \pi_{Qi}(T)$,

$$|\{\pi_{Se}(t') \mid (\pi_{Qi}(t') = t \wedge t' \in T')\}| \geq l. \quad (5)$$

By definition, (5) implies (4), and (4) implies (3).

A conjunctive query consists of projection, selection and join. In our proposed framework, we assume self-join free conjunctive queries.

Definition 3: A query q on $\mathbf{R}$ is *monotonic* if for any $T_1, T_2 \in I(\mathbf{R})$, $T_1 \subseteq T_2$ implies $q(T_1) \subseteq q(T_2)$.

Lemma 1: Every conjunctive query is monotonic.

If we restrict the class of queries to self-join free conjunctive queries, all the three definitions of ℓ-diversity become equivalent as stated in the next theorem.

Theorem 1: If we assume self-join free conjunctive queries, then the three options in definitions 2 become equivalent.

Proof: By the following properties 1 and 2. ■

Property 1: For any instances T_1, T_2 and a self-join free conjunctive query q,

$$q(T_1 \cup T_2) = q(T_1) \cup q(T_2).$$

Proof: Let T_1, T_2 be instances and q be a self-join free conjunctive query. By Lemma 1, q is monotonic and hence $q(T_1 \cup T_2) \supseteq q(T_1) \cup q(T_2)$ holds. Since q does not contain self-join, $q(T_1 \cup T_2) \subseteq q(T_1) \cup q(T_2)$ also holds. ■

Property 2: Let T be an instance and $q_1, \ldots, q_m$ be self-join free conjunctive queries. The largest candidate set in $cand(q_1, ..., q_m, T)$ (with respect to set inclusion) is the union of all instances in $cand(q_1, ..., q_m, T)$.

Proof: Let $T_c = \bigcup_{T' \in cand(q_1, ..., q_m, T)} T'$. By Property 1,

$$\begin{aligned} q_i(T_c) &= \bigcup_{T' \in cand(q_1, ..., q_m, T)} q_i(T') \\ &= \bigcup_{T' \in cand(q_1, ..., q_m, T)} q_i(T) \\ &= q_i(T) \; (1 \leq i \leq m). \end{aligned}$$

Hence, $T_c \in cand(q_1, ..., q_m, T)$. Apparently, T_c is the largest set in $cand(q_1, ..., q_m, T)$. ■

We define the query-based ℓ-diversity problem as follows:

Input : A schema $\mathbf{R}$, an instance $T \in I(\mathbf{R})$, authorized queries $q_1, ..., q_m$, quasi-identifier $Qi \subseteq \mathbf{R}$, sensitive attributes $Se \subseteq \mathbf{R}$, and a threshold $\ell \geq 1$.

Output : T is query-based ℓ-diverse or not with respect to $\mathbf{R}$, Qi, Se, $q_1, ..., q_m$.

IV. Verification by Relational Algebra

In this section, we describe our verification algorithm that solves the query-based ℓ-diversity problem. For simplicity, we only focus on projection queries. The algorithm can be extended to deal with join without self-join. However, selection cannot be allowed. Also, we assume that the set of sensitive attributes is not empty.

We assume that an attacker knows the domain of each attribute in $\mathbf{R}$, specially the domains of the sensitive attributes, so that he can infer a candidate instance by adding values of the sensitive attributes chosen from the domain even if (some of) the sensitive attributes are missing in the result of queries $q_1, ..., q_m$.

Our algorithm consists of four steps as follows:

1) Obtain the candidate set of tuples T' by taking the natural join of all results $q_1(T), ..., q_m(T)$ as follows.
$$T' = q_1(T) \bowtie ... \bowtie q_m(T).$$
2) Let Qi' ($\subseteq Qi$) be the set of quasi-identifier that exist in T'. Compute the subset T_c of T' consisting of tuples whose quasi-identifier value belongs to the original instance T.
$$T_c = T' \bowtie \pi_{Qi'}(T).$$
3) Divide T_c into subsets (equivalence classes) $g_1, \ldots, g_h$ such that
   a) $\pi_{Qi'}(t) = \pi_{Qi'}(t')$ for any $t, t' \in g_i$ $(1 \leq i \leq h)$ and
   b) $\pi_{Qi'}(t) \neq \pi_{Qi'}(t')$ for any $t \in g_i$ $(1 \leq i \leq h)$ and $t' \in g_j$ $(1 \leq j \leq h)$ with $i \neq j$.
4) Let mis_Se be the set of sensitive attributes that does not exist in T'. With mis_Se and the threshold ℓ, decide whether T is ℓ-diverse by examining the following necessary and sufficient condition for ℓ-diversity:
   $\forall g_i$ $(1 \leq i \leq h)$,
$$|g_i| \times \prod_{a \in mis\_Se} | \, dom(a) \, | \geq \ell. \qquad (6)$$

In the last step, the number of different sensitive values in each equivalence class g_i $(1 \leq i \leq h)$ is computed by using the domains of the missing sensitive attributes mis_Se. If for every equivalence class g_i $(1 \leq i \leq h)$, the left-hand side of (6) is greater than or equal to the threshold ℓ, the algorithm answers that the given input is ℓ-diverse. If there is at least one equivalence class g_i such that the left-hand side of (6) is less than ℓ, the algorithm answers that the given input is not ℓ-diverse.

V. Verification by Model Counting

In this section, we provide another method for deciding the query-based ℓ-diversity. The method transforms a given input of the problem to a logical formula, and decides the problem by model counting using a #SAT solver. The advantage of this method is that it can handle self-join free conjunctive queries, consisting of projection, selection and join without self-join. Henceforth, we assume queries in the class.

Before we explain our method, we give some definitions. For a formula Ψ, let $\#models(\Psi)$ denote the number of different models (assignments to variables that make Ψ true). If a formula Ψ contains only variables in Σ, we call Ψ a Σ-formula. For a Σ-formula Ψ and $\Delta \subseteq \Sigma$, let $\Psi_{|\Delta}$ denote the strongest Δ-formula implied by Ψ when considered as a Σ-formula where A is stronger than B if and only if $A \Rightarrow B$ holds. We say that $\Psi_{|\Delta}$ is the projection of Ψ onto Δ.

Assume that a schema $\mathbf{R}$ where $n = |\mathbf{R}|$, an instance $T \in I(\mathbf{R})$, queries $q_1, \ldots, q_m$ on $\mathbf{R}$, quasi-identifies $Qi \subseteq \mathbf{R}$, sensitive attributes $Se \subseteq \mathbf{R}$, and a threshold l are given. For simplicity, suppose that $Qi = \{A_1, \ldots, A_k\} \subseteq \mathbf{R}$, and $Se = \{A_{k+1}, \ldots, A_m\} \subseteq \mathbf{R}$ where $1 \leq k < m < n$. The summary of the method is as follows.

1) Construct a logical formula $\Phi(x_1, \ldots, x_n)$ such that
   $\Phi(c_1, \ldots, c_n)$ is satisfiable
   if and only if $(c_1, \ldots, c_n) \in T_c$ $(*)$
   Note that $\phi(x_1, \ldots, x_n)$ has free variables other than $x_1, \ldots, x_n$ in general.
2) Decide if for all tuple $(c_1, \ldots, c_k) \in \pi_{Qi}(T)$,
$$\#models(\Phi_p(x_{k+1}, \ldots, x_n)_{|Xs}) \geq l.$$
   where $Xs = \{x_{k+1}, \ldots, x_m\}$ and
$$\Phi_p(x_{k+1}, \ldots, x_n) = \Phi(c_1, \ldots, c_k, x_{k+1}, \ldots, x_n).$$

1) Constructing Constraint: Let $n = |\mathbf{R}|$ and $n_i = |\mathbf{R_i}|$ where $\mathbf{R_i}$ is the output schema of q_i $(1 \leq i \leq m)$. To construct a formula $\Phi(x_1, \ldots, x_n)$ satisfying $(*)$, we first construct subformulas ϕ_{q_i} and O_{q_i} for $1 \leq i \leq m$.

(1-i) For $1 \leq i \leq m$, ϕ_{q_i} represents the input-output relation of the query q_i. The formula ϕ_{q_i} contains free variables $x_1, \ldots, x_n, y_1, \ldots, y_{n_i}$ and satisfies:

for any $t = (c_1, \ldots, c_n)$ and $t'_i = (d_1, \ldots, d_{n_i})$, $\phi_{q_i}(c_1, \ldots, c_n, d_1, \ldots, d_{n_i})$ is satisfiable if and only if $q_i(\{t\}) \subseteq \{t'\}$.

(Construction)
If $q = T$ then
$$\phi_q(x_1, \ldots, x_n, y_1, \ldots, y_n) = \bigwedge_{i=1}^{n} (x_i = y_i).$$

projection: If $q = \pi_\alpha(q')$ where $\alpha = \{A_{j_1}, \ldots, A_{j_{n'}}\}$,
$$\phi_q(x_1, \ldots, x_{n_I}, z_1, \ldots, z_{n'}) = \phi_{q'}(x_1, \ldots, x_{n_I}, y_1, \ldots, y_{n_O}) \wedge \bigwedge_{i=1}^{n'} (y_{j_i} = z_i).$$

selection: If $q = \sigma_F(q')$,
$$\phi_q(x_1, \ldots, x_{n_I}, z_1, \ldots, z_{n_O}) = \phi_{q'}(x_1, \ldots, x_{n_I}, y_1, \ldots, y_{n_O}) \wedge \left(P_F(y_1, \ldots, y_{n_O}) \Rightarrow \bigwedge_{i=1}^{n_O} (y_i = z_i) \right).$$

where $P_F(y_1, \ldots, y_{n_O})$ is a formula representing the filtering condition F of σ_F.

 ISBN: 978-1-61208-408-4

cross product: If $q = q' * q''$,

$$\begin{aligned}\phi_q(x_1,\ldots,x_{n'_I},x'_1,\ldots,x'_{n''_I},z_1,\ldots,z_{n'_O+n''_O}) \\ = \ \phi_{q'}(x_1,\ldots,x_{n'_I},y_1,\ldots,y_{n'_O}) \\ \wedge\ \phi_{q''}(x'_1,\ldots,x'_{n'_I},y'_1,\ldots,y'_{n''_O}) \\ \wedge \bigwedge_{i=1}^{n'_O}(y_i = z_i) \wedge \bigwedge_{i=1}^{n''_O}(y'_i = z_{n'_O+i}).\end{aligned}$$

(1-ii) O_{q_i} is defined as

$$\begin{aligned}O_{q_i}(y_1,\ldots,y_{n_i}) \\ = \bigvee_{(d_1,\ldots,d_{n_i})\in q_i(T)} ((y_1 = d_1) \wedge \cdots \wedge (y_{n_i} = d_{n_i})).\end{aligned}$$

(1-iii) Finally, Φ is defined as

$$\begin{aligned}\Phi(x_1,\ldots,x_n) \\ = \bigwedge_{i=1}^{m}(\phi_{q_i}(x_1,\ldots,x_n,y_{i,1},\ldots,y_{i,n_i}) \wedge O_{q_i}(y_{i,1},\ldots,y_{i,n_i})).\end{aligned}$$

Remember that in the algorithm of the previous section, we introduce the subsets $g_1,\ldots,g_h$, each of which shares same values of the quasi-identifier. For g_j ($1 \leq j \leq h$), let $(c^j_1,\ldots,c^j_k)$ be the values of the quasi-identifier shared by tuples in g_j. Let $\Phi^j_p(x_{k+1},\ldots,x_n) = \Phi(c^j_1,\ldots,c^j_k,x_{k+1},\ldots,x_n)$. By $(*)$, $\Phi^j_p(c_{k+1},\ldots,c_n)$ is satisfiable if and only if $(c^j_1,\ldots,c^j_k,c_{k+1},\ldots,c_n) \in g_j$. Furthermore, $\Phi^j_p(x_{k+1},\ldots,x_n)_{|Xs}$ is the strongest Xs-formula implied by $\Phi^j_p(x_{k+1},\ldots,x_n)$. Hence, the number of assignments to variables in Xs that make $\Phi^j_p(x_{k+1},\ldots,x_n)_{|Xs}$ true coincides with the number of different values of Se appearing in tuples that belong to g_j. Hence, we obtain the following lemma.

Lemma 2: Let $\mathbf{R}$ be a schema, $Qi, Se \subseteq \mathbf{R}$ be the quasi-quantifier and sensitive attributes, respectively, $q_1,\ldots,q_m$ be self-join free conjunctive queries on $\mathbf{R}$ and $T \in I(\mathbf{R})$ be an instance. Let $g_1,\ldots,g_h$ be the subsets of T_c, each of which shares same values for the quasi-identifier. For each j ($1 \leq j \leq h$), the number of different values of sensitive attributes in g_j is

$$\#models(\Phi^j_p(x_{k+1},\ldots,x_n)_{|Xs}).$$

2) Counting Candidates: To count the different values of sensitive attributes for each g_j ($1 \leq j \leq h$), we transform $\Phi^j_p(x_{k+1},\ldots,x_n)$ to an equivalent propositional formula Φ^j_{cnf} in conjunctive normal form (CNF) by using Sugar [12]. Next, for each $t' = (c_1,\ldots,c_k) \in \pi_{Qi}(T)$, we construct a CNF formula $\psi_{t'}$ that represents $x_1 = c_1 \wedge \cdots \wedge x_k = c_k$, and then count $\#models(\Phi^j_{cnf} \wedge \psi_{t'})_{|P(Xs)}$, where $P(Xs)$ is the set of the propositional variables in Φ^j_{cnf} corresponding to Xs in $\Phi^j_p(x_{k+1},\ldots,x_n)$. We use sharpCDCL [6], which is a #SAT solver (an automatic tool for counting the models of a given propositional formula). Among other #SAT solvers that can count models, the advantage of sharpCDCL is that it can automatically count $\#models(\Psi_{|\Delta})$ only by giving a formula Ψ and a subset Δ of propositional variables. If some $t' \in \pi_{Qi}(T)$ such that $\#models(\Phi^j_{cnf} \wedge \psi_{t'})_{|P(Xs)} < \ell$ is found, we say that T is not ℓ-diverse. Otherwise, T is ℓ-diverse.

VI. EXPERIMENTS

A. Experimental Result of Relational Algebra

The purpose of the experiment was to investigate the scalability of our approach.

1) Setup: Experiment were performed on a 3.33 GHz Intel(R) Core(TM) i7 CPU with 6GB of RAM. The operating system was Microsoft Windows 8.1 Enterprise, and implementation was built and run in MySQL Workbench, version 6.1. We used available dataset, Employees Sample Database [14], Copyright (C) 2007, 2008, MySQL AB, version 1.0.6. The database contains about 300,000 tuples with 2.8 million salary entries. In our experiment, the schema consists of ten attributes, where five attributes $\{Gender,\ DeptName,\ BirthDate,\ HireDate,\ FromDate\}$ were designated as the quasi-identifier and the sensitive attribute is $\{Salary\}$.

2) Datasets and Queries: The proposed algorithm was implemented in MySQL and was performed on three instances (datasets) with $n = 37,500, 75,000, 150,000, 300,000$ tuples. Also, we prepared three queries, each of which is the projection onto the following attributes:

$q_1 : \{EmpNo, LastName, Gender\}$.
$q_2 : \{EmpNo, Salary, HireDate\}$.
$q_3 : \{DeptName\}$.

In the experiment, we used three sets of queries, namely, $\mathbf{Q_A} = \{q_1, q_2\}$, $\mathbf{Q_B} = \{q_3\}$, and $\mathbf{Q_C} = \{q_1, q_2, q_3\}$. For example, for $\mathbf{Q_A}$, the verification algorithm took the natural join of the results of q_1 and q_2 on each of the datasets in Step 1. In Step 3, the algorithm constructed the table from the candidate set T_c obtained in Step 2 by grouping tuples that have same values of the quasi-identifier. Lastly, in Step 4, the algorithm tested ℓ-diversity ($\ell = 2$ in the experiment).

TABLE VI. TOTAL TIME OF VERIFYING 2-DIVERSITY.

Dataset	Cases	Total time
$37,500$	q_1, q_2	7sec
	q_3	4sec
	q_1, q_2, q_3	11sec
$75,000$	q_1, q_2	19sec
	q_3	8sec
	q_1, q_2, q_3	31sec
$150,000$	q_1, q_2	1min 7sec
	q_3	14sec
	q_1, q_2, q_3	1min 47sec
$300,000$	q_1, q_2	4min 8sec
	q_3	26sec
	q_1, q_2, q_3	8min 48sec

Table VI shows the total running time of our algorithm. For example, for $\mathbf{Q_A}$ and the dataset n = $37,500$, the total running time is 7sec. Also the running time of each set of queries, $\mathbf{Q_A}, \mathbf{Q_B}$ and $\mathbf{Q_C}$ on the datasets of size $n = 37,500, 75,000, 150,000$ and $300,000$. We can observe that the decision algorithm is efficient, in general, and also the computation time depends on the size of the datasets.

B. Experimental Result of Model Counting

1) Setup: Experiment were performed on a 3.10 GHz Intel(R) Core(TM) i5 CPU with 8GB of RAM. The operating system was Ubantu 14.04. We performed the experiment on a dataset, having 50,000 tuples.

2) Datasets and Queries: In the experiment, we used two instances T_1 and T_2, having ten and eleven attributes, respectively. Both of T_1 and T_2 have $5,000$ tuples. Actually, T_1 was obtained from T_2 by projecting out one of the eleven attributes. We conducted the experiment on the following two settings:

$D1$: a query $\sigma_{state=Iwate}(T_1)$, $Qi = \{ID, state\}$ and $Se = \{Name\}$.

$D2$: two queries $\Pi_{BirthYear,BirthMonth}(T_2)$, $\Pi_{BirthYear,BirthMonth}(\sigma_{Carrier=SoftBank}(T_2))$, $Qi = \{ID, BirthMonth, Carrier\}$ and $Se = \{BirthYear\}$.

The experimental results for these settings are shown in Table VII where clauses and variables are those in the transformed CNF formula, projected variables are the variables corresponding to sensitive attributes, min count is the minimum number of different values of sensitive attributes among $g_1, \ldots, g_h$. That is, D1 is ℓ-diverse if and only if $\ell \leq 50$ and D2 is ℓ-diverse if and only if $\ell \leq 42$. Next, we

TABLE VII. PERFORMANCE OF MODEL COUNTING METHOD.

	Clauses	Variables	Projected variables	Min count	Time
$D1$	$34,271$	$14,916$	$4,961$	50	6min 15sec
$D2$	$22,941$	$11,121$	96	42	2min 1sec

increased the number of tuples in T_2 to $10,000$, $30,000$ and $50,000$ and examined the scalability of the proposed method by using the setting $D2$. The result is shown in Table VIII. In sharpCDCL, an upperbound U of the model counting can be specified. That is, when sharpCDCL detects that the current number of models reaches U, sharpCDCL terminates. The computation times in Table VIII are those when this upperbound is specified as $U = 20$. The transformation to a

TABLE VIII. SCALABILITY OF MODEL COUNTING METHOD.

Tuples	Clauses	Variables	Projected variables	Time
$5,000$	$22,941$	$11,121$	96	1min 58sec
$10,000$	$42,334$	$20,877$	96	10min 2sec
$30,000$	$117,557$	$58,541$	106	2hrs 28min 30sec
$50,000$	$187,820$	$93,633$	106	8hrs 56min 8sec

CNF formula takes less than one second, and model counting dominates the running time.

VII. CONCLUSION

We have introduced query-based ℓ-diversity as a privacy notion for a realistic database system that assumes access control for queries. This new notion inherits from ℓ-diversity of [8] the quantitative notion for the diversity of sensitive attributes. Also, the notion utilizes k-secrecy of [5] by taking attacker's inference on the authorized information into consideration.

We proposed two approaches to deciding whether a given database instance satisfies query-based ℓ-diversity with respect to given queries. The first approach is based on relational algebra computation that counts the candidate values of the sensitive attributes. The second approach transforms a given input to a logical formula and then decide the problem by counting models of a formula by a #SAT solver. The first approach can directly be implemented by an existing relational database system such as SQL, and the experimental results show that this approach is fairly efficient. The weakness is that it cannot deal with selection queries. The second approach, on the other hand, can deal with selection queries. However, the model counting in a #SAT solver is generally time consuming and we have not yet customized the solver to our problem and hence, the performance is not good compared with the first approach.

Applying the proposed approach to other kind of databases such as object-oriented or XML databases is left as a future study.

REFERENCES

[1] R. J. Bayardo and R. Agrawal, "Data Privacy Through Optimal k-Anonymization," the 10th International Conference on Database Theory (ICDT), 2005, pp. 217-228.

[2] J. W. Byun, A. Kamra, E. Bertino, and N. Li, "Efficient k-Anonymization Using Clustering Techniques," the 12th International Conference on Database Systems for Advanced Applications (DASFAA), 2007, pp. 188-200.

[3] A. Deutsch and Y. Papakonstantinou, "Privacy in Database Publishing," the 10th International Conference on Database Theory (ICDT), 2005, pp. 230-245.

[4] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu, "Privacy-Preserving Data Publishing: A Survey of Recent Developments," ACM Computing Surveys, vol. 42, 2010, pp. 14:1-53.

[5] K. Hashimoto, K. Sakano, F. Takasuka, Y. Ishihara, and T. Fujiwara, "Verification of the Security Against Inference Attacks on XML Databases," IEICE Transactions on Information and Systems, vol. E92-D, 2009, pp. 1022-1032.

[6] V. Klebanov, N. Manthey, and C. Muise, "SAT-Based Analysis and Quantification of Information Flow in Programs," 10th International Conference on Quantitative Evaluation of Systems (QEST), 2013, pp. 177-192.

[7] N. Li, T. Li and S. Venkatasubramanian, "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity, the 23rd IEEE International Conference Data Engineering (ICDE), 2007, pp. 106-115.

[8] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-Diversity: Privacy Beyond k-Anonymity," 22nd International Conference on Data Engineering (ICDE), 2006, also in ACM Transactions on Knowledge Discovery from Data (TKDD), 2007, vol. 1(3), 52 pages.

[9] G. Miklau and D Suciu, "A Formal Analysis of Information Disclosure in Data Exchange," Journal of Computer and System Sciences, vol. 73, 2007, pp. 507-534.

[10] C. Phonharath, K. Hashimoto, and H. Seki, "Deciding Schema k-Secrecy for XML Databases," IEICE Transactions on Information and Systems, vol. E96-D, 2013, pp. 1268-1277.

[11] P. Samarati, "Protecting Respondents' Identities in Microdata Release," IEEE Transactions on Knowledge and Data Engineering, vol. 13, 2001, pp. 1010-1027.

[12] "A SAT-Based Constraint Solver," URL: http://bach.istc.kobe-u.ac.jp/sugar/[accessed: 2015-01-23]

[13] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, vol. 10, 2002, pp. 557-570.

[14] F. Wang and C. Zaniolo, "Employees Sample Database," 2008, URL: https://launchpad.net/test-db/[accessed: 2014-11-13].

 ISBN: 978-1-61208-408-4