GSJ: Volume 9, Issue 5, May 2021, Online: ISSN 2320-9186
www.globalscientificjournal.com

KEEPING SECRETS IN INCOMPLETE DATBABASES IN THE PRESENCE OF NULL VALUES

ANN JOSE

ABSTRACT

Database management systems allow for massive storage of data. In a relational database there is a need to keep sensitive information's hidden from a group of users. The specification that what a particular user is allowed or not allowed to access should be specified in a declarative manner.

The database return answers that do not reveal anything that should be kept protected from a particular user. For this secrecy views can be set. Then a query about any of those views returns no meaning full information. The databases are not physically changed for this purpose, Updates are only virtual. For maintaining privacy there is semantics of secrecy views, virtual updates and secret answers to queries are used. The virtual updates are based on null values and slicing. Slicing partitions data both horizontally and vertically, Slicing preserves better data utility than generalization and can be used for membership disclosure protection. Another important advantage of slicing is that it can handle high-dimensional data.

KEYWORDS:-

Slicing,

Secrecy Instances (SI)

Secret Answers (SA)

Secret query answering (SQA)

INTRODUCTION

1.1OVERVIEW

ABOUT KNOWLEDGE AND DATA ENGINEERING:

KE is an engineering discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise. It is used in many computer science domains such as artificial intelligence, including databases, data mining, expert systems, decision support systems and geographic information systems. Knowledge engineering is also related to mathematical logic, as well as strongly involved in cognitive science and socio-cognitive engineering where the knowledge is produced by socio-cognitive aggregates (mainly humans) and is structured according to our understanding of how human reasoning and logic works. Data & Knowledge Engineering (DKE) stimulates the exchange of ideas and interaction between these two related fields of interest.

Activities of KE specific for the development of a knowledge-based system:

- Assessment of the problem
- Development of a knowledge-based system shell/structure
- Acquisition and structuring of the related information, knowledge and specific preferences (IPK model)
- Implementation of the structured knowledge into knowledge bases
- Testing and validation of the inserted knowledge
- Integration and maintenance of the system
- Revision and evaluation of the system.

. LITERATURE SURVEY

2.1.1 Applications of Annotated Predicate Calculus to Querying Inconsistent Databases

In this paper we consider the problem of specifying and computing consistent answers to queries against databases that do not satisfy given integrity constraints. This is done by simultaneously embedding the database and the integrity constraints, which are mutually inconsistent in classical logic, into a theory in annotated predicate calculus - a logic that allows non trivial reasoning in the presence of inconsistency. In this way, several goals are achieved: (a) A logical specification of the class of all minimal “repairs" of the original database, and the ability to reason about them; (b) The ability to distinguish between consistent and inconsistent information in the database; and (c) The development of computational mechanisms for retrieving consistent query answers, *i.e.*, answers that are not affected by the violation of the integrity constraints.

2.1.2Authorization Views and Conditional Query Containment

In this paper, a recent proposal for database access control consists of defining “authorization views" that specify the accessible data, and declaring a query valid if it can be completely rewritten using the views. Unlike traditional work in query rewriting using views, the rewritten query needs to be equivalent to the original query only over the set of database states that agree with a given set of materializations for the authorization views.

With this motivation, we study conditional query containment, *i.e.* , containment over states that agree on a set of materialized views. We give an algorithm to test conditional containment of conjunctive queries with respect to a set of materialized conjunctive views. . Based on the algorithm, we give a test for a query to be conditionally authorized given a set of materialized authorization views.

2.1.3 The Impact of the Constant Complement Approach Towards View Updating

Views play an important role as a means to structure information with respect to specific users' needs. While read access through views is easy to handle, update requests through views are difficult in the sense that they have to be translated into appropriate updates on database relations. In this paper the "constant complement translator" approach towards view updating proposed by Bancilhon and Spyratos is revisited within the realm of SQL databases, and a novel characterization is established showing that constant complement translators exist precisely if users have a chance to undo all effects of their view updates using further view updates. Based on this characterization view updates with and without constant complement translators are presented. As it turns out that users cannot fully understand updates on views violating the constant complement principle, the application of this principle in the context of external schema design is discussed.

2.1.4 Extending Query Rewriting Techniques for FineGrained Access Control

Current day database applications, with large numbers of users, require fine-grained access control mechanisms, at the level of individual tuples, not just entire relations/views, to control which parts of the data can be accessed by each user. Fine-grained access control is often enforced in the application code, which has numerous

drawbacks; these can be avoided by specifying/enforcing access control at the database level. We present a novel fine-grained access control model based on authorization views that allows "authorizationtransparent" querying; that is, user queries can be phrased in terms of the database relations, and are valid if they can be answered using only the information contained in these authorization views. We extend earlier work on authorization-transparent querying by introducing a new notion of validity, conditional validity. We give a powerful set of inference rules to check for query validity. We demonstrate the practicality of our techniques by describing how an existing query optimizer can be extended to perform access control checks by incorporating these inference rules.

2.1.5 Handling Inconsistency In Databases and Data Integration Systems

For several reasons a database may not satisfy certain integrity constraints (ICs), for example, when it is the result of integrating several independent data sources. However, most likely, information in it is still consistent with the ICs; and could be retrieved when queries are answered. Consistent answers with respect to a set of ICs have been characterized as answers that can be obtained from every possible minimal repair of the database. The goal of this research is to develop methods to retrieve consistent answers for a wide and practical class of constraints and queries from relational databases and from data integration systems. We will put special interest on databases with null values. We will give a semantics of satisfaction of constraints in the presence of null that generalizes the one used in commercial DBMS.

Since there are interesting connections between the area of consistently querying virtual data integration systems and other areas, like querying incomplete databases merging inconsistent theories, semantic reconciliation of data, schema mapping, data exchange, and query answering in peer data management systems, the results of this research could also be applied to them. In our research, we explore in more depth the connection with virtual data integration systems and peer data management systems.

2.1.6 Minimal-Change Integrity Maintenance Using Tuple Deletions

We address the problem of minimal-change integrity maintenance in the context of integrity constraints in relational databases. We assume that integrity-restoration actions are limited to tuple deletions. We focus on two basic computational issues: repair checking (is a database instance a repair of a given database?) and consistent query answers [3] (is a tuple an answer to a given query in every repair of a given database?). We study the computational complexity of both problems, delineating the boundary between the tractable and the intractable cases. We consider denial constraints, general functional and inclusion dependencies, as well as key and foreign key constraints. Our results shed light on the computational feasibility of minimal-change integrity maintenance. The tractable cases should lead to practical implementations. The intractability results highlight the inherent limitations of any integrity enforcement mechanism, e.g., triggers or referential constraint actions, as a way of performing minimal-change integrity maintenance.

2.1.7 Preprocessing for Controlled Query Evaluation with Availability Policy

Controlled Query Evaluation (CQE) defines a logical framework to protect confidential information in a database. By modeling a user's a priori knowledge appropriately, a CQE system not only controls access to certain database entries but also accounts for information inferred by the user. In this article, we present a static (preprocessing) CQE-approach for propositional databases with an availability policy. The resulting inference-proof and availability-preserving database ensures confidentiality of secret information while guaranteeing availability of certain database entries to a highest degree possible. We illustrate the semantics of the system by a comprehensive example and state the essential requirements for an inference-proof and availability-preserving database. We present an algorithm that accomplishes the preprocessing by combining SAT solving and “Branch and Bound”.

2.1.8View Determinacy for Preserving Selected Information in Data Transformations

The view determinacy introduces a characterization of selected preservation, investigates its fundamental problems and establishes their complexity bounds. There are two criteria to specify the preservation of selected informations.Consider the setting in which data transformations are specified in terms of a view from source to target and the selected information is identified by a query Q defined on the data source. The view determinacy problem is undecidable when either queries or views are in first order(FO).

2.1.9 A review of privacy preserving data publishing technique

Preserving data publishing technique introduces several anonymization techniques such as generalization and bucketization,have been designed for privacy preserving micro data publishing. Work has shown that generalization loses considerable amount of information. especially for high-dimensional data.on the other hand bucketization does not prevent membership disclosure, Whereas slicing preserves better utility than generalization and also prevents membership disclosu This paper focus on effective method that can be used for providing data utility and can handle high-dimensional data.

2.1.10 On the Anonymization Of Sparse High-Dimensional Data

Several microdata anonymization techniques have been proposed.The most popular ones are generalization for k-anonymity and bucketization. In both approaches,attributes are partitioned in to 3 categories(1)some attributes are identifiers that can uniquely identify an individual (2)some attributes are Quasi-Identifiers(QI),Which the adversary may ready to know and which when taken together can potentially identify an individual(3)some attributes are Sensitive Attributes(SA) which are unknown adversary .In both generalization and bucketization,one first removes identifiers fro the data and then partitions tuples in to buckets.

METHODOLOGIES

Slicing partitions the data set both horizontally and vertically. Vertical portioning is done by grouping attributes in to columns based on the correlations among the attributes. Each column contains a subset of attributes that are highly correlated. Horizontal portioning is done by grouping tuples in to buckets.

The basic idea of slicing is to break the association cross columns ,but to preserve the association within each column. This reduces the dimensionality of the data and preserves better utility than generalization and bucketization. Slicing preserves utility because it groups highly correlated attributes together and preserves the correlation between such attributes. Slicing protects privacy because it breaks association between correlated attributes which are infrequent and thus identifying.

AGE	SEX	ZIP CODE	DISEASE
22	M	47906	DYSPEPSIA
22	F	47906	FLU
33	F	47905	FLU
52	F	47905	BRONCHITIS
54	M	47905	FLU
60	M	47302	DYSPEPSIA
60	M	47304	DYSPEPSIA
64	F	47304	GASTRITIS

ORIGINAL DATA

1)Generalization and Suppression

A value is replaced by a less specific more general value that is faithful to the original.In figure the original zip codes{02138,02139} can be generalized to 0213*,thereby stripping the rightmost digit and semantically indicating a larger geographical area

$$\begin{tikzcd}
Z_{3} = \{*****\} \\
Z_{2}=\{021**\} \arrow[u] \\
Z_{1}=\{0213*,0214*\} \arrow[u] \\
Z_{0}=\{02138,02139,02141,02142\} \arrow[u]
\end{tikzcd}$$

DGH_{Z0}

In a classical database system ,domains are used to describe the set of values that attributes assume.For example ,there might be a ZIP domain ,a number domain and a string domain extend this notion of a domain to make it easier to describe how to generalize the values of an attribute .In the original database, where every value is as specific as possible ,every attribute is considered to be in a ground domain.

AGE	SEX	ZIP CODE	DISEASE
20-52	*	4790*	DYSPEPSIA
20-52	*	4790*	FLU
20-52	*	4790*	FLU
20-52	*	4790*	BRONCHITIS
54-64	*	4790*	FLU
54-64	*	4730*	DYSPEPSIA
54-64	*	4730*	DYSPEPSIA
54-64	*	4730*	GASTRITIS

The Generalized Table

2) Bucketized Data

This illustrate the efficiency of slicing in membership disclosure protection. For this purpose calculate the number of fake tuples in the sliced data. Also compare the number of matching buckets for original tuples and that for fake tuples. Experimental results illustrate that bucketization does not prevent membership disclosure as almost every tuple is distinctively identiable in the bucketized data.

AGE	SEX	ZIP CODE	DISEASE
22	M	47906	FLU
22	F	47906	DYSPEPSIA
33	F	47905	BRONCHITIS
52	F	47905	FLU
54	M	47905	GASTRITIS
60	M	47302	FLU
60	M	47304	DYSPEPSIA

The Bucketized Table

3)One-Attribute Per Column Slicing Data

One –Attribute- Per- Column Slicing preserves attribute distributional information ,it does destroy attribute correlation, for the reason that each attribute is in its own column.In slicing one group associated attribures together in one column and save their correlation.For Instance, in the sliced table shown in table,correlations between age and sex and zip code and disease are conserved.In fact the sliced table encodes the same amount of information as the original data .

AGE	SEX	ZIP CODE	DISEASE
22	M	47906	FLU
22	F	47906	FLU
33	F	47905	DYSPEPSIA
52	F	47905	BRONCHITIS
54	M	47905	DYSPEPSIA

60	M	47302	GASTRITIS
60	M	47302	DYSPEPSIA
64	F	47304	FLU

One-Attribute Per Column Slicing Data table

4) Sliced data

Another important advantage of slicing is its capability to handle high dimensional data .By dividing attributes into columns ,slicing condense the measurements of the data .Each of which column of the table can be viewed as a sub-table with a lesser dimensionality. Slicing is also not similar from the approach of publishing multiple independent sub-tables in that these sub-tables are associated by the buckets in slicing.

AGE	SEX	ZIP CODE	DISEASE
22	M	47905	FLU
22	F	47906	DYSPEPSIA
33	F	47905	BRONCHITIS
52	F	47905	FLU
54	M	47905	GASTRITIS
60	M	47302	FLU
60	M	47304	DYSPEPSIA
64	F	47304	DYSPEPSIA

The Sliced Table

CONCLUSION

Slicing is a promising technique for handling high-dimensional data. Slicing overcomes the limitations of generalization and bucketization and preserves better utility while protecting against privacy threats, Slicing prevent attribute disclosure and membership disclosure .The experiments shows that slicing preserves better data utility than generalization and is more effective than bucketization in workloads involving the sensitive attribute, The general methodology proposed by this work is that before anonymizing the data, one can analyze the data characteristics and use these characteristics in data anonymization. The rationale is that one can design better data anonymization techniques when the data is better .Attribute correlations can be used for privacy attacks .In future there is an extension of over-lapping slicing which duplicates an attribute in more than one columns .This releases more attribute correlations and this could provide better data utility.

REFERENCE PAPERS

1. S. Abiteboul, R. Hull, and V. Vianu, "Foundations of Databases. Addison-Wesley", 1995.
2. P. Barcelo, "Applications of Annotated Predicate Calculus and Logic Programs to Querying Inconsistent Databases," 2002.
3. L. Bertossi, "Consistent Query Answering in Databases," ACM Sigmod Record vol. 35, no. 2, pp. 68-76, June 2006.
4. L. Bertossi, "From Database Repair Programs to Consistent Query Answering in Classical Logic (Extended Abstract)," Proc. Alberto Mendelzon Int'l WorkshopFoundations of Data Management (AMW '09), vol. 450, 2009.
5. L. Bertossi, Database Repairing and Consistent Query Answering. Morgan & Claypool, 2011.
6. J. Biskup and T. Weibert, "Confidentiality Policies for Controlled Query Evaluation," Data and Applications Security, 4602, pp. 1-13, 2007.
7. J. Biskup and Weibert, "Keeping Secrets in Incomplete Datbabases," Int'l J. Information Sercurity, vol. 7, no. 3, pp. 199- 217, 2008.

8. J. Biskup, C. Tadros, and L. Wiese, “Towards Controlled Query Evaluation for Incomplete First-Order Databases,” Proc. Sixth Int’l Conf. Foundations of Information and Knowledge Systems (FoIKS ’10), pp. 230-247, 2010.

9. L. Bravo and L. Bertossi, “Semantically Correct Query Answers in the Presence of Null Values,” Proc. EDBT WS on Inconsistency and Incompleteness in Databases (IIDB ’06), pp. 336-357, 2006.

10. L. Bravo, “Handling Inconsistency in Databases and Data integration Systems,” PhD thesis, Dept. Computer Science, Carleton Univ.,