Towards an Automatic Detection of Sensitive Information in a Database

Cédric du Mouza, Elisabeth Métais, Nadira Lammari, Jacky Akoka, Tatiana Aubonnet, Isabelle Comyn-Wattiau, Hammou Fadili and Samira Si-Saïd Cherfi
Lab. CEDRIC, CNAM, Paris, France
dumouza@cnam.fr metais@cnam.fr lammari@cnam.fr akoka@cnam.fr tatiana.aubonnet@cnam.fr
wattiau@cnam.fr fadili@msh-paris.fr samira.cherfi@cnam.fr

***Abstract*—In order to validate user requirements, tests are often conducted on real data. However, developments and tests are more and more outsourced, leading companies to provide external staff with real confidential data. A solution to this problem is known as Data Scrambling. Many algorithms aim at smartly replacing true data by false but realistic ones. However, nothing has been developed to automate the crucial task of the detection of the data to be scrambled. In this paper we propose an innovative approach - and its implementation as an expert system - to achieve the automatic detection of the candidate attributes for scrambling. Our approach is mainly based on semantic rules that determine which concepts have to be scrambled, and on a linguistic component that retrieves the attributes that semantically correspond to these concepts. Since attributes can not be considered independently from each other we also address the challenging problem of the propagation of the scrambling among the whole database. An important contribution of our approach is to provide a semantic modelling of sensitive data. This knowledge is made available through production rules, operationalizing the sensitive data detection.**

***Keywords*-data privacy; semantic rule; sensitive data;**

I. INTRODUCTION

Developers need realistic data in order to test and validate their new applications. Programmers are used to perform their final tests on excerpts from the on-going production databases. However, the information in many databases is proprietary and because of its privacy aspect it must be protected. While many works have focused on how data may be protected from external attacks, the problem of the protection of sensitive data inside a company has just been recently explored. The main motivation relies on the recent phenomenon of the externalization of any development and test. Let us illustrate with two typical examples. First the database of an hospital with all personal and medical information about patients. Second the clients database of a large company. In the first case, we expect any person developing an application on the medical data not to be able to extract any personal information about a patient. In the second application, a leak of information during the development or testing phase can cause considerable business damage if transmitted to a competitor.

This work has been partially founded by F.X. Beorchia

A solution to this problem is known as *data scrambling* or *de-identification* and consists in altering the sensitive data. Several techniques have already been proposed to transform a sensitive database into a non informative one: substitution, shuffling, number and date variance, etc. The challenge is to sufficiently mask real value and prevent from any way to retrieve them by a combination of questions while preserving enough realism and data distribution to allow a good validation of program's results. However, existing proposals for de-identification lack an automatic detection of the sensitive data that may lead to human faints (ranging from the omission of a sensitive attribute to the non-detection of an hidden "link" between two attributes) with possible horrendous sensitive information leaks.

To face this problem we propose in this paper an innovative technique that automates the detection of the sensitive attributes. This technique relies on two functionalities: (1) Automatic detection of the values to be scrambled; (2) Automatic propagation to other semantically linked values. We propose a rule based approach implemented under an Expert System architecture. Rules are devoted to the selection of sensitive data with regard to their semantics. Among the modules of the expert system, one has to ensure the application of the rules on the particular values of the data base schema (*e.g.* if a rule claims that "salaries have to be scrambled" the expert system has to recognize that an attribute named `wages` has to be scrambled). Furthermore we present a deduction mechanism modelled by a semantic graph to ensure the propagation of the confidentiality on near values and the consistency with the other relations. Finally, based on this observation we propose a prototype with a set of clever interfaces to capture the rules.

The paper is organized as follows: Section II defines what a sensitive attribute is. Section III shows the different approaches we use to set the base of rules. Section IV details the propagation mechanism of attributes to be scrambled. The prototype is presented in Section V. Section VI is devoted to related work and Section VII concludes and presents perspective.

II. SENSITIVE DATA

A database in production may contain sensitive information that must not be visible (or at least exploitable) when the

database is used during development or test phases. We distinguish *identity information* that allows to identify a person or entity stored in the database from *confidential information* whose content may be harmful if revealed. We are convinced that both kinds of information must be considered when sanitizing a database. Thus we consider the following definitions.

Let $\mathcal{D}$ be a database and $\mathcal{S}$ be the set of all attributes in $\mathcal{D}$. Let k be a parameter that depends on the application and that represents the minimal number of occurrences required for assuming anonymity (see the k-anonymity approach [1]).

Definition *1 (Confidential attribute):* The confidential attributes set, denoted $\mathcal{S}_c \subseteq \mathcal{S}$ is the set of attributes whose content is confidential, whatever the number of occurrences they have.

Our notion of *identifying attribute* is similar to the notion of quasi-identifier in [1] except it can not be considered independently from the confidential attributes.

Definition *2 (Identifying attribute):* The identity attributes set, denoted $\mathcal{S}_i \subseteq \mathcal{S}$ is the set of attributes such that for any $x \in \mathcal{S}_i$ it exists a subset $s_i \subseteq \mathcal{S}_i$ within a single table $\mathcal{T}$ of $\mathcal{D}$ and with $x \in s_i$, such that (i) each instance of s_i occurs less than k times in the records from $\mathcal{T}$ and (ii) there is an attribute $y \in \mathcal{S}_c$ in $\mathcal{T}$. $\mathcal{S}_i$ is called the k-*identity set*.

Within other words, each instance of an (or a group of) identifying attribute has less than $k-1$ occurrences, and is considered as selective enough to identify a small number of persons. Since there is a confidential attribute in the same table, that means the individual privacy is endangered. Note that we assume k set for the application, but we can easily extend our definition to capture applications where a different value for k is set for each table.

Finally, we define a sensitive attribute as following.

Definition *3 (Sensitive attribute):* The sensitive attributes set, denoted $\mathcal{S}_s$, is the set of identifying and confidential attributes for the table $\mathcal{D}$, *i.e.*, $\mathcal{S}_s = \mathcal{S}_i \cup \mathcal{S}_c$.

Observe that $\mathcal{S}_i \cap \mathcal{S}_c$ may not be empty. Our rationale for considering both confidential and identifying attributes in the scrambling process is based on the following observations. The scrambling of the identity attributes preserves anonymity while confidential attributes keep their initial distribution. However, this is clearly not sufficient when the presence of some instances of attributes must remain itself confidential or at least unexploitable. Oppositely, the scrambling of the confidential attributes aims at protecting individual privacy by modifying the value of confidential attributes while information that identifies persons remains unchanged. But in that case local (*e.g.*, value range, precision, etc.) and global (*e.g.*, average, min, max, etc.) properties of the concerned attributes are changed what may invalidate a test or development phase. Consequently both types of attributes must be simultaneously considered as sensitive and candidates for scrambling.

Example *1:* Assume a Human Resources Department (HRD) database that stores information concerning employees like employee's id, name, city, department, name of the superior, wage, etc. Unlike the first two properties that permit to identify an employee ($\mathcal{S}_i = \{id, name\}$), and thus to consult all his data, address, department and wages properties are apparently less confidential. Nonetheless one may avoid to reveal the highest salary or the average salary of a given department. Such properties must then also be considered as sensitive ($\mathcal{S}_c = \{wage\}$). Moreover in smaller companies one can argue that the couple (city,department) is sufficient to identify a small subset of employees and consequently must be added to the k-identity set also, while for larger companies this information is not identifying enough. So finally for our large company we have to scramble $\mathcal{S}_s = \{id, name, wage\}$.

III. Detecting sensitive data

While most existing tools need as an input the attributes to be scrambled, our tool aims at helping in the detection of sensitive attributes. We automate the detection of sensitive attributes with a combination of techniques based on Deduction Rules, Natural Language Processing (NLP) and statistics. Deduction Rules are mainly used to build $\mathcal{S}_c$, Statistics to compute $\mathcal{S}_i$ and NLP to expand $\mathcal{S}_s$ with semantically close attributes. The whole process includes a human expert validation that can at any time add new rules or modify proposed sensitivity scores.

A. The Rule Based Approach

Our approach to automate the identification of sensitive attributes relies essentially on rules that represent the knowledge of experts on the sensitivity of the data in a given context. The rule based approach is divided into two steps: (i) the acquisition step that implies the human expertness, and (ii) the rules application step that can be fully automated.

The rules that we consider here may be of different kinds. We distinguish the two following kinds of rules:

- *intentional rules* with conditions on database schema (mainly attribute's names);
- *extensional rules* with conditions on attribute's instances.

Example *2:* Rules like "`salary` is a highly confidential attribute", "attributes with type `autoincrement` must be scrambled" (generally denote identifiers), or "the whole extension of the PASSWORDS relation is confidential" are all examples of the first rules kind. Oppositely, rules based on the fact that a column with some instances that contain words like `euros` or `street` may refer respectively to private data on salary or on address of the employees, belong to the second kind of rules.

Let Δ be the set of all possible domains of application, Θ the set of all possible table names, Φ the set of all possible attribute names and Ψ the set of all possible attribute values. We call an attribute instance an instance $(\delta, \theta, \phi, \psi) \in \Delta \times \Theta \times \Phi \times \Psi$. of the quadruple (domainName,tableName,attributeName,attributeValue).
While theoretically rules may be complex, we adopt the

simple following rule definition.

Definition *4 (rule):* A *rule condition* $\chi = \chi_1 \boxplus \chi_2$ is a condition with $\chi_1 \in \{domainName, tableName, attributeName, attributeValue\}$, $\chi_2 \in \Delta \cup \Theta \cup \Phi \cup \Psi$, and $\boxplus$ is an operator in $\{=, !=, <, >, \leq, \geq, contains, !contains\}$. Finally, a *rule* is composed by disjunctions and conjonctions of rule conditions along a rule sensitivity score $\sigma \in [0, 1]$, where σ permits to evaluate how sensitive is an attribute that satisfies the rule.

This sensitivity score allows us to sort the different attributes according to their sensitivity. The user can then decide the security level she wants for her application by fixing the sensitivity threshold. All attributes with a score above this threshold must be scrambled. Here is a rule example.

Example *3:* Assume we consider that a column whose name contains "salar" if the domain is HRD and there are values greater than 15,000 or lower than 5,000 is highly sensitive (score=0.9). The corresponding rule is expressed by the following expression:

$$((domainName =' HRD') \\ \wedge(attributeName\ contains\ 'salar') \\ \wedge(attributeValue > 15000 \\ \vee attributeValue < 5000))\ ,\ 0.9$$

Finally, if an attribute α has one or several instances or metadata that satisfy at least one rule, this attribute is candidate for scrambling. The sensitivity score of α for a given set of rules is defined as follows.

Definition *5 (attribute's sensitivity score):* Let $\mathcal{I}$ be the set of instances and metadata for α and $\mathcal{R}$ be the set of rules such that $\forall \rho \in \mathcal{R}, \exists \iota \in \mathcal{I}, \iota \vDash \rho$. The *sensitivity score* of the attribute α is defined as:

$$score(\alpha) = \begin{cases} 0 & if\ \mathcal{R} = \emptyset \\ max_{\rho \in \mathcal{R}}(\sigma_\rho) & otherwise \end{cases}$$

σ_ρ denotes the score of the rule ρ. In other words, we consider that either the attribute is concerned by no rule and its sensitivity score is zero, or several rules are satisfied for this attribute and consequently its sensitivity score is the highest of all the rule sensitivity scores. We have chosen this way of computation among other candidate formulae (min, average, Bonczek-Eagin, hybrid mixture,etc, see [2] for more) since we give priority to the highest security.

The existence or not of the $domainName$ in a rule allows us to classify the rules in two families. On the one hand there are context-free rules (*i.e.* when no $domainName$ is set) that are applied whatever the domain of the application is. On the other hand we have noticed there exist domain-dependent rules: they may be valid in a given domain and false in other domains. A practical way to define some rules is based on experts knowledge. Simple rules concerning one single attribute may be acquired from the experts by the mean of a matrix. The given marks allow them to set the sensitivity scores of the attributes. Of course many other techniques may be applied to populate our knowledge database.

B. Other detection techniques

a) The statistical computation: Candidates for the set $\mathcal{S}_i$ of identity attributes can be retrieved in the metabase (thanks to *primary key* and *unique* integrity constraints) or computed via the statistics since the selectivity of each attribute is generally stored in the metabase for query optimization purpose. But the statistics stored in the metabase are generally not sufficient to supply all the required information on the selectivity since they consider only single attributes. Although it has been shown that determining all the subsets of attributes that are quasi-identifiers is a NP-hard problem [3], several heuristics [3], [4], [5] have been proposed. Finally, to determine $\mathcal{S}_i$ for the different candidates found we must check if the table they belong to also presents confidential attributes.

b) Necessity of Natural Language Processing: Rules are stated upon concepts. However, in a given application the attributes may not have been named with exactly the same word that the one used in the rules. The matching between the word used in the rule and the attributes name involves NLP techniques. Stating on the similarity between two words has been a research topic for a long time and since the nineties numerous works have been proposed using ontologies like WordNet [6], [7], [8]. In our WordNet based solution the matching between names in the rules and names in the relations requires a function $\mathsf{APPR0X}(att\_name, att\_name\_in\_rule) \rightarrow \sigma$. The inputs of this function are att_name the attribute name in the relation and $att\_name\_in\_rule$ the name of the attribute as specified in the rule. The output of $\mathsf{APPR0X}$ is the sensitivity score σ corresponding to the rule on the attribute with $att\_name\_in\_rule$ name.

IV. Propagating sensitivity scores

Applying the previous techniques to a database results in a set of attributes $\mathcal{S}_s^{init}$ identified for scrambling. Halting the process at that step would probably lead to an incomplete result since there exist links between attributes from different tables and any sensitivity score for an attribute must be propagated to another. This section presents the different links between attributes that we consider and how we use them to propagate the sensitivity scores.

A. The propagation graph model

We consider two kinds of links between attributes. Links explicitly defined in the database schema as integrity referential constraints, and implicite links based on semantics.

1) Integrity referential links:

Since a foreign key attribute references a primary or secondary key attribute, any modification of the former must

impact the latter. However, the foreign keys are generally not detected neither as identity attribute since their selectivity is low (a primary key value is referenced by the foreign key of many tuples) and nor as confidential data since they are not explicitely targetted by rules.

Example *4:* In our HDR database example, assume that the table which stores information about employees has an attribute NameOfManager refering to the attribute name of the employee. Since many employees share the same manager, the techniques presented above do not detect this attribute as sensitive. However, if we decide to scramble the name of the employee (primary key), then we have to cascade and to scramble also the attribute NameOfManager (foreign key).

Since integrity referential constraints are explicitly stored in the database we can extract them to propagate sensitivity scores. Assume the set PK of primary or secondary keys, we use the following notation to refer to the integrity referential constraints: $\gamma_r : 2^{|\mathcal{S}|} \rightarrow 2^{|\mathcal{S}|}$ ($2^{|\mathcal{S}|}$ denotes the power set of $\mathcal{S}$) defined as (with FK standing for "foreign key")

$$\forall x \in 2^{|\mathcal{S}|}, \gamma_r(x) =$$

$$\begin{cases} \{y \mid y \in 2^{|\mathcal{S}|},\ y\ FK\ referring\ to\ x\} & if\ x \in PK \\ \emptyset & otherwise \end{cases}$$

Finally, we denote for any set $P \subseteq \mathcal{S}$, the result set

$$\Gamma_r(P) = \bigcup_{x \in 2^{|P|}} \gamma_r(x)$$

2) Semantical links:

Integrity referential constraints are not the only links that exist between attributes. E.g. an attribute in a table may have the same semantics than another one in another table. The NLP approach for the rules allows to apply rules on attributes based on the semantics, whatever the attribute's name is. So if a rule is applied for an attribute, this same rule must be applied to any other attributes sharing the same meaning. The expert may also decide that an attribute has to be scrambled independently of what our system proposed. Such a decision must also propagate to all the "semantically linked" attributes.

Example *5:* Assume the expert decides that the salaries must not be revealed. When she sets the attribute `salary` in one table as sensitive, she intends that all the other attributes in any table that refer to the same kind of information, like salary in another table, but also wages, bonus, income, etc, have to be set in the same way. Later she realizes that the `address` attribute must be scrambled too. Starting from this selected attribute, the sensitivity must be propagated to the couple `(street,city)` in another table for instance.

These semantical links may be either stored in the rules database or extracted from some general (*e.g.* WordNet [9]) or domain-based ontologies. We use the notation $\gamma_s : 2^{|\mathcal{S}|} \rightarrow 2^{|\mathcal{S}|}$ to refer to the semantic constraints defined as

$$\forall x \in 2^{|\mathcal{S}|}, \gamma_s(x) = \{y | y \in 2^{|\mathcal{S}|},\ x\ is\ semantically\ linked\ to\ y\}$$

Finally, we denote for any set $P \subseteq \mathcal{S}$, the result set

$$\Gamma_s(P) = \bigcup_{x \in 2^{|P|}} \gamma_s(x)$$

B. Propagation algorithm

We use the referential and semantical links between attributes to extend the set of attributes $\mathcal{S}_s^{init}$ identified for scrambling and validated by the expert using the techniques presented in Section III. We proceed to the following iterative algorithm to determine the final set $\mathcal{S}_s$ of attributes to scramble:

(i) $\mathcal{S}_s^{(0)} = \mathcal{S}_s^{init}$

(ii) $\mathcal{S}_s^{(k+1)} = \mathcal{S}_s^{(k)} \cup \Gamma_r(\mathcal{S}_s^{(k)}) \cup \Gamma_s(\mathcal{S}_s^{(k)})$

Lemma *1 (convergence):* The algorithm conver ges to $\mathcal{S}_s$ with at most $|\mathcal{S}|$ iterations.

Proof: The proof is straighforward: $\mathcal{S}_s^{(k)}, k \in \mathbb{N}$ is monotonic increasing and is bounded by $\mathcal{S}$ it converges. Moreover note that we have $\mathcal{S}_s^{(k+1)} = \mathcal{S}_s^{(k)}$ we reach the convergence and the algorithm stops since it means that no link permits to extend $\mathcal{S}_s^{(k)}$ and the result is stabilized. While convergence is not reached, the result set extends at each step by at least one attribute. Consequently, the algorithm converges in at most $|\mathcal{S}|$ steps. □

If the propagation process leads to a conflict set of different sensitivity values for the same attribute, the maximum level is preferred as presented in Section III.1.

Finally, when a candidate attribute has been selected for scrambling one must determine the adequate algorithm to apply. This is however out of the scope of the paper and remains as future work.

V. Prototype presentation

To validate our approach, we implemented a prototype in Java mainly for its portability using an Expert System's approach. We have chosen JESS [10], a rule engine and scripting environment dedicated to Java applications, as an expert system. JESS stores the rules in files with *clp* extension which allows us to easily import/export rules files. Theses files can also be completed by the expert and/or user (depending on the genericity/specificity of the rule) through the tool interface.

The NLP treatments are supported by the WordNet [9] ontology that provides, among other links, synonymy and proximity links between words. Currently our prototype takes into consideration only synonymy links to detect if a rule written for a given attribute's name applies to an attribute in

the database to be scrambled while its name is syntactically different. Our implementation relies on the JAWS API as an interface between our application and WordNet, and JDBC to connect the application to the database. Evaluation has been performed on ORACLE; next prototype will focus on SAP applications.

The tool finally provides as a result an XML file with the set of attributes for each table along their sensitive score. This XML file is then processed by the second module (not presented here) in charge of determining adequate scrambling strategies for each sensitive attributes. Using an XML file as an output also allows the expert or the advanced user to directly edit the XML files for adding or modifying some rules.

Our experiment has convinced us that, unlike computer scientists, domain's experts and more generally all users are more familiar to attribute's values than to attribute's names. Thus we provide them with some examples of data in order to help experts in their decisions.

Figure 1 shows such a proposal based on simple `select` queries on the different tables. A sample of a query's result is proposed with a different color for each attribute, corresponding to the level of confidentiality based on the acquired rules. The expert can change a color each time she doesn't agree. This change is propagated in cascade to other attributes connected either by referential constraint link or semantic one. Here for instance increasing the sensitivity score for the attribute `department_ID` from green (score of 0.2) to red (score of 0.6) in table `DEPARTMENT` impacted the sensitivity score for attributes `department_ID` and `dep_ID` in respectively table `EMPLOYEE` and `JOB_HISTORY` that get in turn a red label. A visual alert warns the user when tables not currently displayed have an attributes whose sensitivity has changed when cascading.

Moreover the attribute `first_name` satisfied a rule on `family_name` attribute and got a very high sensitivity score symbolized by the dark blue color.

Our tool also proposes an interface to edit, add or delete rules on attributes or instances. Due to space limitations we do not describe the interface here.

VI. Related work

Determining a sanitization strategy which guarantees that the data provided preserve confidentiality is a complex task. In [11] the authors prove that finding the sanitization that minimizes the sensitivity of the values with respect to some sensitive rules is a NP-hard problem. A large number of heuristics have been proposed like [1], [12], [13], [14], [15] to find a satisfying sanitization under precise hypothesis.

A first family of approaches are based on sensitive association rules. These approaches hide the frequent itemsets corresponding to these rules by modifying the sensitive transactions that contain those itemsets. In [12] for instance, the authors present a privacy preservation heuristic algorithm named *sliding window algorithm (SWA)* that hides in one pass on a transactional database association rules by decreasing their support. [16] proposes three heuristics also based on rules that outperform SWA in terms of maximizing data utility of the sanitized databases but that require computational overhead.

Several approaches are semantics-free and rely on the number of occurrences inside each equivalence class (*i.e.*, a set of records that could not be distinguished w.r.t. a given identifying attribute). The most famous ones are k-anonymity [1] that imposes for a class to contain at least k records, and l-diversity [13] that improves the k-diversity by forcing equivalence classes to contain at least l well-represented values for each sensitive attribute. [14] is based on both k-anonymity and l-diversity, but requires that the distribution of an attribute in an equivalence class is close to the one of the real table. Since discovering frequent patterns in large databases is largely used for commercial purposes, some approaches aim at hiding sensitive patterns like in [17], [18], [15]. However, all these approaches assume that sensitive attributes or patterns are known and do not consider links between attributes.

Other proposals have been devoted to the sanitization of free-text, mainly in the medical domain [19], [20]. However, the problem is different in free-text and consists basically in identifying sensitive words based on a specialized domain semantics. They do not consider any links between terms except potentially synonymy and usually do not aim at guaranteeing any data utility after sanitization. One interesting exception for Health information is [21] that presents a prototype for extracting information and identifying entities. They applied anonymization process for both structured and unstructured data. Here again authors rely on k-anonymity [1] and l-diversity [13] to determine sensitive attributes. [22] presents the ERASME framework for sanitization of unstructured documents based on term scoring functions for building the set of terms to delete to make the document K-safe. However, no link between attributes is considered.

Several softwares are also proposed to de-identify databases like [23], [24], [25], [26], [27]. They basically offer the same functionalities, *i.e.*, to select sensitive attributes, to choose a scrambling technique among a proposed set (shuffling, replacing with synthetic data, masking, deleting, encrypting, ...) to apply for each attribute. [23] also proposes enhanced functionalities like using templates for replacing data with adapted synthetic data or respecting integrity constraints (within tuples, between tuples or between tables). Nonetheless all these tools do not provide any help for detecting sensitive attributes that can lead to important security flaws. Our proposal aims at facing this crucial step.

VII. Conclusion

Scrambling test databases is a crucial need for an increasing number of companies. As shown in the related work many techniques have been developed for scrambling the data; however nothing has been proposed to automatically determine which part of the database needs scrambling. In this paper we have proposed an approach to detect sensitive attributes and its implementation based on an expert system architecture.

We have proposed a rule based approach for determining the attribute's sensitivity level. Primary keys, indices and statistics

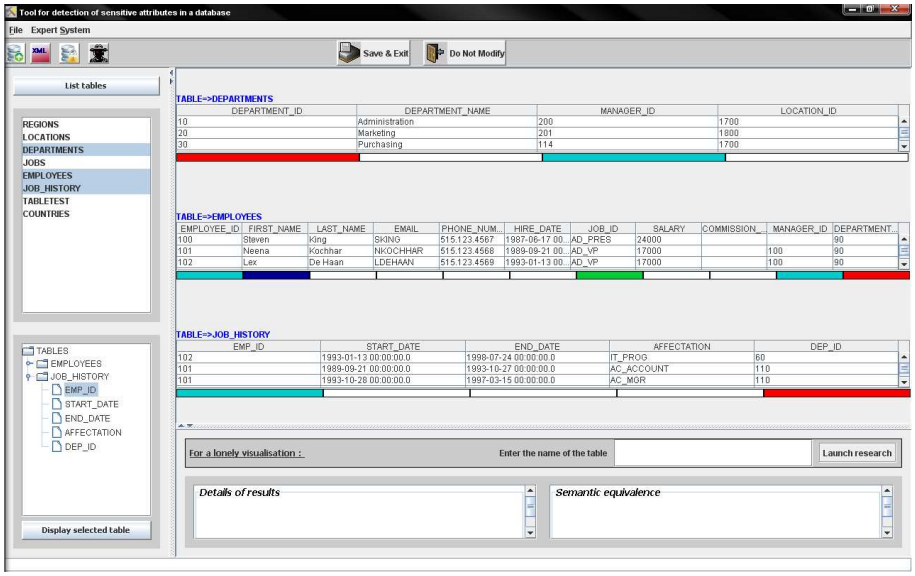

Fig. 1. Prototype's interface

on the database stored in the DBMS for optimization purpose are used to detect attributes that are quite identifying for the tuples. Labelling rules calling the Wordnet ontology are provided to match the attribute's names used in the rules with the exact names of the attributes in a given application. These rules may also be general or specific to one application. Integrity referential constraints and other semantic links are exploited for the propagation of the sensitivity among attributes.

Our next work will focus on the evaluation of the resulting base. In particular it is difficult to be certain that the scrambled base doesn't contain any inconsistency due to a bad propagation of the scrambling among all the tables. An experimentation will be preformed on an SAP application, where data are very connected together, sometimes through complex deduction and management rules.

REFERENCES

[1] L. Sweeney, "Achieving k-Anonymity Privacy Protection Using Generalization and Suppression," *Intl. Jour. of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 571–588, 2002.

[2] R. Blanning, "Sensitivity Analysis in Hierarchical Fuzzy Logic Models," in *Proc. Intl Conf. on Decision Support and Knowledge Based Systems Track*, 1988, pp. 471–476.

[3] A. Meyerson and R. Williams, "On the Complexity of Optimal K-Anonymity," in *Proc. Intl. Symp. on Principles of Database Systems (PODS)*, 2004, pp. 223–228.

[4] R. J. B. Jr. and R. Agrawal, "Data Privacy through Optimal k-Anonymization," in *Proc. Intl. Conf. on Data Engineering (ICDE)*, 2005, pp. 217–228.

[5] H. Park and K. Shim, "Approximate Algorithms for K-Anonymity," in *Proc. Intl. Conf. on Management of Data (SIGMOD)*, 2007, pp. 67–78.

[6] E. Métais, J.-N. Meunier, and G. Levreau, "Database Schema Design: A Perspective From Natural Language Techniques to Validation and View Integration," in *Proc. Intl. Conf. on Conceptual Modeling (ER)*, 1993, pp. 190–205.

[7] R. Richardson, A. Smeaton, and J. Murphy, "Using WordNet as a Knowledge Base for Measuring Semantic Similarity Between Words," in *Proc. Irish Conf. on Artificial Intelligence and Cognitive Science (AICS)*, 1994.

[8] F. Lin and K. Sandkuhl, "A Survey of Exploiting WordNet in Ontology Matching," in *Proc. Intl. Conf. on Intelligent Information Processing, Artificial Intelligence in Theory and Practice (IFIP AI)*, 2008, pp. 341–350.

[9] "WordNet: An Electronic Lexical Database," http://wordnet.princeton.edu.

[10] "JESS, the Rule Engine for the Java Platform," http://www.jessrules.com.

[11] M. Atallah, A. Elmagarmid, M. Ibrahim, E. Bertino, and V. Verykios, "Disclosure Limitation of Sensitive Rules," in *Proc.Intl. Workshop on Knowledge and Data Engineering Exchange (KDEX)*, 1999, pp. 45–52.

[12] S. R. M. Oliveira and O. R. Zaïane, "Protecting Sensitive Knowledge By Data Sanitization," in *Proc. Intl. Conf. on Data Mining (ICDM)*, 2003, pp. 613–616.

[13] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-Diversity: Privacy Beyond k-Anonymity," in *Proc. Intl. Conf. on Data Engineering (ICDE)*, 2006, p. 24.

[14] N. Li, T. Li, and S. Venkatasubramanian, "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity," in *Proc. Intl. Conf. on Data Engineering (ICDE)*, 2007, pp. 106–115.

[15] E. T. Wang and G. Lee, "An Efficient Sanitization Algorithm for Balancing Information Privacy and Knowledge Discovery in Association Patterns Mining," *Data Knowl. Eng.(DKE)*, vol. 65, no. 3, pp. 463–484, 2008.

[16] A. Amiri, "Dare to Share: Protecting Sensitive Knowledge with Data Sanitization," *Decision Support Systems (DSS)*, vol. 43, no. 1, pp. 181–191, 2007.

[17] M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi, "Blocking Anonymity Threats Raised by Frequent Itemset Mining," in *Proc. Intl. Conf. on Data Mining (ICDM)*, 2005, pp. 561–564.

[18] Z. Wang, W. Wang, and B. Shi, "Blocking Inference Channels in Frequent Pattern Sharing," in *Proc. Intl. Conf. on Data Engineering (ICDE)*, 2007, pp. 1425–1429.

[19] L. Sweeney, "Datafly: A system for providing anonymity in medical data," in *Proc. Intl. Conf. on on Database Security(DBSec)*, 1997, pp. 356–381.

[20] B. A. Beckwith, R. Mahaadevan, U. J. Balis, and F. Kuo, "Development and Evaluation of an Open Source Software Tool for Deidentification of Pathology Reports," *BMC Medical Informatics and Decision Making*, vol. 6, no. 12, 2006.

[21] J. Gardner and L. Xiong, "HIDE: An Integrated System for Health Information DE-identification," in *Proc. Intl. Symp. on Computer-Based Medical Systems (CBMS)*, 2008, pp. 254–259.

[22] V. T. Chakaravarthy, H. Gupta, P. Roy, and M. K. Mohania, "Efficient Techniques for Document Sanitization," in *Proc. Intl. Conf. on Information and Knowledge Management (CIKM)*, 2008, pp. 843–852.

[23] "Datamasker," http://www.datamasker.com.

[24] "Camouflage," http://www.datamasking.com.

[25] "Solix," http://www.solix.com.

[26] "Datavantage Globa," http://www.datavantage.com.

[27] "Pse Data Security," http://www.psedatasecurity.com.