UNIVERSITY OF CALGARY

PriSQL – A Privacy Preserving SQL Language

By

Sampson Pun

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

JANURARY, 2010

UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "PriSQL – A Privacy Preserving SQL Language" Submitted by Sampson Pun in partial fulfilment of the requirements of the degree of Master of Computer Science

[illegible]
Supervisor, Dr. Ken Barker
Department of Computer Science

[illegible]
Dr. Reda S. Alhajj
Department of Computer Science

[illegible]
Dr. Payman Mohassel
Department of Computer Science

[illegible]
Dr. Behrouz Far
Department of Electrical and Computer Engineering

January 05 2010
Date

Abstract

PriSQL is a privacy extension of the structured query language. PriSQL's extension allows data providers to feel confident that their private data is used according to the privacy policy specified by the corporation. Through a new privacy aware access mechanism, the privacy of data is imposed by technological means introduced within this thesis. This thesis has two major contributions. The first contribution of the thesis is the description of a robust privacy model which is capable of encompassing the definitions of other academic researchers. This model ensures that all aspects of privacy are considered within PriSQL. The second contribution is the extension of the structured query language. PriSQL is capable of allowing any DBMS to understand the semantic and syntactic requirements of privacy. Lastly, the usefulness of PriSQL is demonstrated through the implementation of the IBM web privacy policy.

Table of Contents

List of Figures and Illustrations

Glossary

Symbol	Definition
Access Purpose	The reason for attempting to view sensitive data.
Database Administrator	The individual(s) responsible for creating and maintaining the DBMS.
Data Provider	The individual(s) providing sensitive information to the DBMS.
Data Requestor	A user of the DBMS which is submitting a PriSQL query for information.
Data Retention	The period of time that sensitive data can remain within the DBMS.
Generalization	The abstraction of data to a less detailed level
Information Privacy	The right to control the dissemination of information about oneself by a third party.
Intended Purpose	The reason that the data was collected and its allowable use.
PriSQL	Privacy Structured Query Language
Privacy Retention	The length of time that privacy permissions are valid.
Sensitive Data	Personal information that is meant to be released on a need to know basis.
Trusted Users	A set of users that will not perform malicious acts toward the DBMS in order to violate the privacy of the data providers.

Epigraph

“All that may come to my knowledge in the exercise of my profession or in daily commerce with men, which ought not to be spread abroad, I will keep secret and will never reveal.

If I keep this oath faithfully, may I enjoy my life and practice my art, respected by all men and in all times; but if I swerve from it or violate it, may the reverse be my lot.”

- Hippocratic Oath

Chapter One: Introduction

Databases are an essential part of all successful businesses. These databases store vast amounts of customer and corporate information. Techniques to analyze this kind of information are becoming more and more complex; ranging from simple statistics to association rule mining and other data mining methods. The ability to retrieve user and business patterns, and their selected statistics, is a major benefit for companies interested in achieving their business goals and strategies. The data used to determine these trends is often considered sensitive information to the individuals involved. To protect the privacy of these individuals; numerous privacy related guidelines and legislations have been adopted throughout the world. These guidelines and legislations aim to bring privacy preservation to the forefront of corporate activity[1].

To comply with the legislation enacted, corporations have been forced to alter their corporate procedures and implement new training regiments. Both of these methods are human-based solutions to a technological problem. Many corporations such as IBM™, Microsoft ™, Facebook ™ and MySpace ™ participate in the EU Safe Harbour Privacy Framework [SH00] to comply with privacy legislation. This framework outlines seven privacy principles which compliant corporations must follow:

- Notify Internet users about the type of data collected at the Web site, the manner in which it is collected, for what purpose, and whether it will be disclosed to third parties. They must also inform users of options for limiting the use and disclosure of that information.

[1] Some examples of these legislations and guidelines are: OECD Privacy Guidelines in Europe, the Canadian Privacy Act, the Australian Privacy Amendment Act, the Japanese Privacy Code, the Health Insurance Portability and Accountability Act (HIPAA), and Gramm-Leach-Bliley Consumer Privacy Rule.

- Provide individuals with the chance to opt out of having their personal data collected or disseminated to third parties.
- Guarantee that data will be transferred only to other Safe-Harbor compliant parties.
- Facilitate individuals' access to their personal data and provide a means for them to correct inaccurate information.
- Undertake "reasonable precautions" to secure the data from loss, alteration, or unauthorized access or disclosure.
- Utilize the data only for purposes that have been disclosed to the individuals.
- Put in place enforcement mechanisms that will ensure compliance. These include providing accessible, affordable, and independent venues through which individuals can lodge complaints for breach of Safe Harbor principles and through which justifiable damages can be awarded, and a system to verify that the company has in fact implemented the Safe Harbor principles. [EU Safe Harbor]

Corporate procedures and the development of simple opt-out programs can fulfill the requirements of the first four principles. Reasonable precautions to safe guard data and utilizing it only for intended purposes are the two principles that are more challenging to enforce. PriSQL supplements the protection of private data through a new data access methodology.

PriSQL allows corporations to control the privacy of its databases similar to security-based access control methods. Rather than only authenticating a user, PriSQL adds a layer of protection which is focused entirely on protecting the privacy of the data providers. This allows privacy to become a first-order principle of the DBMS rather than an afterthought that is too often ignored or forgotten.

1.1 The Goal

The goal of developing PriSQL is to increase the trust between data providers and data collectors. By increasing the trust among users, data collectors will receive better quality data while the data providers will not have to worry about their privacy being violate through inappropriate use of their information. In order to achieve this goal, privacy must be implemented as a first order principal within the database management system. As a first order principle, privacy would never be sacrificed due to time and space limitations. Protecting the privacy of the data providers will become the most important task within the DBMS. PriSQL's objective is to ensure that users of PriSQL cannot violate any of the published privacy policies agreed upon by the corporation and their customers.

1.2 The Contribution

This thesis has two major contributions to the field of database privacy. The first contribution is an extension to the Structured Query Language (SQL). This extension, known as PriSQL, is capable of recognizing the syntactic and semantic requirements for making privacy a first order principle. The second contribution of this thesis is the development of a privacy model that is robust enough to encompass all the different privacy definitions currently in the literature.

Chapter Two: Background

This chapter outlines the definition of privacy which is used throughout this thesis. This privacy definition is a novel contribution as it aims to be robust enough to encompass the definition of privacy that other academic researcher's use; while still being specific enough to implement a feasible access control methodology through SQL. The chapter is divided as follows: Section 2.1 defines the term *privacy* as well as what it means for PriSQL to *protect* privacy. Section 2.2 defines a privacy data model that considers all the different parameters which affect privacy. Section 2.3 through 2.6 reviews privacy research literature which has been conducted in each of the different parameters of privacy.

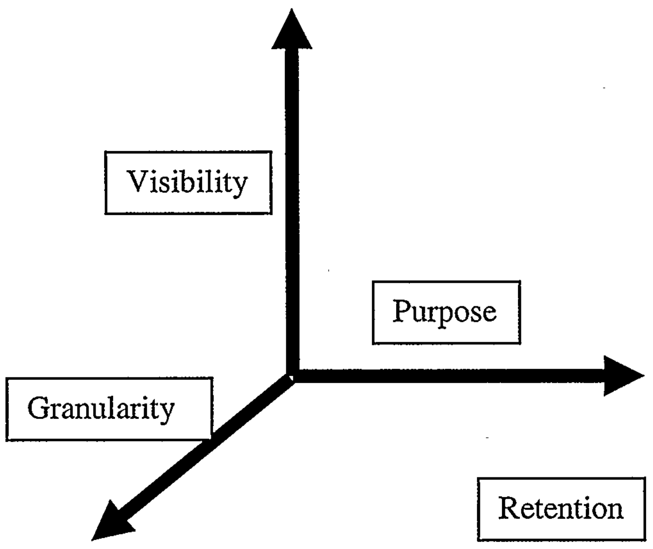
2.1 Privacy

The backbone of PriSQL is its definition of *privacy*. Every privacy researcher seems to have their own unique definition of the term. There are undeniable commonalities among all the variations and PriSQL's use of the term, privacy, aims to facilitate all variations. Privacy was defined by the Supreme Court of America as the right of the individual to control the dissemination of information about oneself [Ric95]. The focus of PriSQL is *information privacy*, the right to control the dissemination of information about oneself by a third party (government or corporations). Oftentimes we provide personal information to government or corporations out of necessity (e.g., taxes) or as an exchange of services (e.g., credit card application). The ability to control how this information is used after it is collected is essential in determining whether privacy has been protected.

Ideally, each data provider would be able to define their own privacy requirements and a corporation should be obliged to follow each individual policy. This would fully satisfy the definition of information privacy. Unfortunately, corporations wish to have as much control over the data they have as possible. This allows corporations to extract the most information possible from any data collected. Thus, variations in data-use permissions would frustrate corporate users. PriSQL aims to strike a balance between these two conflicting desires. PriSQL will allow a corporation to publish their privacy policy and guarantee that the corporation will only use it for the purposes stated. While PriSQL will add a layer of privacy protection for the information; the protection of the information from unauthorized users lies within the realm of security. PriSQL does not provide any additional protection against such adversaries.

2.2 Privacy Model

To enforce information privacy within PriSQL, there must be a formal definition of privacy. To create such a definition we further explore the concept of privacy. Imagine this scenario: you have just received your final grade in a class that you have taken with a few friends. Three different individuals are now asking for your grade: your friends, your parents, and a recruiter for a company for which you wish to work. How will you respond? One appropriate response would be determine the *reason* these individuals want your information. Given these reasons you might still choose to *generalize* the response to respond appropriately to each request. You could provide your friends and the recruiter with your actual grade while your response to your parents would be 'alright'. This scenario has already shown three major parameters when considering privacy: generalization, visibility, and purpose. For information privacy, PriSQL must also take into account the retention period for which any of these privacy tuples are valid. The formal model of privacy is shown in Figure 1 and includes *generalization, visibility, purpose and retention* [BaM09]. This privacy model is capable of incorporating other academic researchers' definition of privacy and is the most comprehensive definition available currently. Each privacy parameter will be explained in the following sections along with a discussion at some of the privacy work by other researchers.



Privacy: <Purpose, Generalization, Visibility, Retention>

Figure 1 – Formal Specification of Privacy.

2.3 Purpose

Purpose is a major element of privacy, and arguably the most important. Purpose is the reason that a piece of information will be accessed or viewed. Acceptable reasons may come from three categories: user requests, government requests, and business requests. Users may often times request their own data to maintain its accuracy but these request have no foreseeable privacy concerns. Government requests for information may pose a privacy concern, however in today's society, government provisions and laws override any privacy concerns of data providers. Government requests for data come in many forms such as subpoenas for specific information to controversial wire taps by enforcement officers. One recent example of a government request for data is the US Congress subcommittee requesting for the release of AIG executive bonus payments [DoL09]; a situation where both the individuals involved and the corporation has no input as to whether the information is released. Government access to data is at the current forefront of privacy advocate's concerns. Many privacy advocates argue that governments are gaining too much power and individuals' rights to privacy must be respected even at the cost of social benefits. Whether or not the government is ignoring basic human privacy is not a concern of PriSQL. PriSQL simply supports government data access as defined by the law.

PriSQL was initially designed and created to support business based data access. Thus, PriSQL ensures that the listed business purposes for data collection are the only reasons that access will be given. Currently, many corporations publish a privacy policy indicating their possible reasons for using the information collected. However, none of the DBMSs used by these corporations has system-level data privacy protection. Access to private data is simply controlled by user authentication. To enforce privacy policies, corporations develop internal procedures which employees follow. Unfortunately, human error and lapses of judgement can easily circumvent the procedures preventing privacy violations [HPP03]. PriSQL implements privacy as a first-ordered principal where human error and malicious users are not able to violate the privacy of the data providers.

2.3.1 Purpose Tree

There are no ways of predicting the number of purposes that a corporation will have to access your data. To properly coordinate these purposes, a purpose tree was developed by Byun *et al.* Given a corporation's privacy policy, a list of data access purposes can be inferred and a purpose tree stores these purposes in an effective manner [ByB05]. Stored in each node of a purpose tree is the purpose itself. Each edge of the purpose tree represents a hierarchical relation between two purposes. The hierarchical relation may be a concept hierarchy of the purposes in a given domain. An example of a purpose tree is given in Figure 2. When data is collected, the information is stored along with a purpose indicating the *intended purpose.* This purpose tree provides a list of possible reasons that the data is allowed to be accessed. *Access purpose* is the purpose of a specific data request. If the access purpose is part of the intended purpose tree, then access is granted to the data requestor.

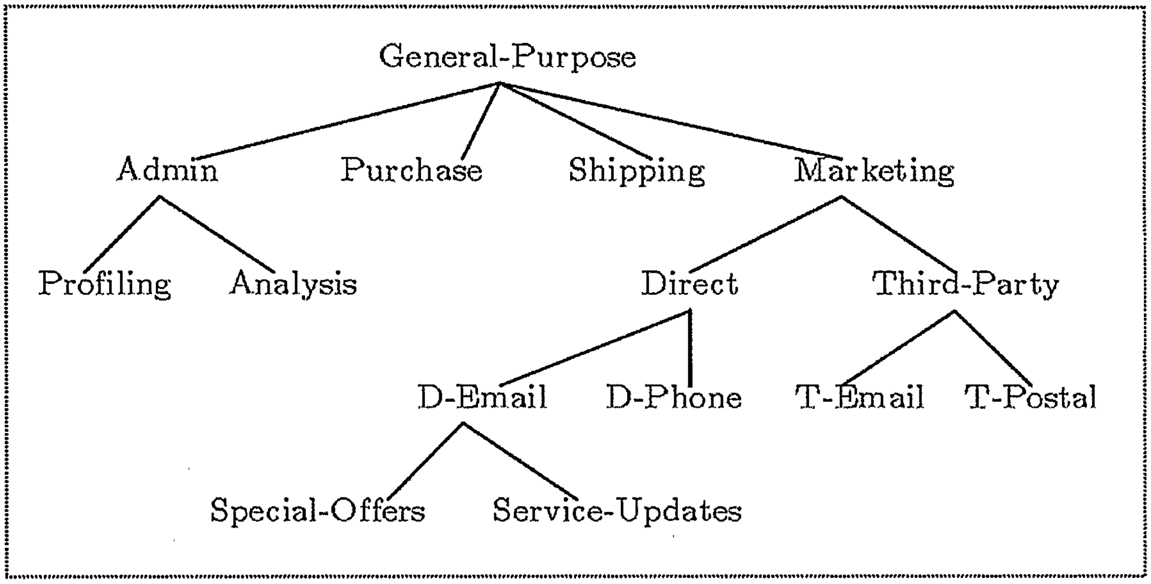


Figure 2 – Purpose Tree [ByB05]

For example, Joyce provides her personal and credit card information to Facebook^{TM}. Her intentions are that the information is used for the purchase of a virtual gift for her friend. When a Facebook employee attempts to access the information with the purposes of marketing other virtual gifts, her information will not be released since it is not part of the intended purposes. If the employee was accessing the information to complete the purchase of the gift (i.e. a credit card transaction) then the information would be available as it matches the intended purpose.

2.4 Generalization

For years privacy has been synonymous with data encryption and anonymization. By suppressing or altering private data, sensitive information is kept safe and secure from adversaries. While this notion is not entirely true (supposedly safe algorithms are later determined to be incomplete or inadequate to protect against all adversaries), individuals do feel more comfortable about their privacy when information is altered prior to its disclosure. This is evident by the recommendation by the Technology and Privacy Advisory Committee to perform data anonymization upon databases known or reasonably likely to include personally identifiable information [Ohm09]. This devotion to data anonymization has led to an abundance of work within this field. The following sections describes the major concepts central to this privacy field and describes in detail two of the most popular data anonymization concepts, k-anonymity and l-diversity.

2.4.1 Identifiers and Non-Identifiers

From a privacy perspective, information within a database can be placed into two categories: *identifiers* and *non-identifiers*. The identifier category is further divided into two groups, keys and quasi-identifiers [Dal86, Swe02]. Keys are pieces of information, if published, that will immediately identify a data provider in the database. Social insurance numbers, birth certificate numbers, and passport numbers are examples of keys. Quasi-identifiers are sets of information which when combined together can explicitly or implicitly identify a person. Addresses, gender, and birthdays are examples of quasi-identifiers. Alone, none of these pieces of information can identify an individual but together they form a key which allows data providers to be identified. Non-identifiers can also be divided into two categories: *sensitive* and *non-sensitive* information. Sensitive information is the data we are interested in keeping private. Information collected on an individual's health status, income, purchase history, and/or criminal past would all be considered sensitive information. Non-sensitive data are facts that data providers would have no privacy concern about if they were revealed. Defining whether an attribute is sensitive information or a quasi-identifier is very application specific. In privacy research it is often assumed that an unknown adversary has an unbounded knowledge of information aside from the sensitive attributes [XiT06, MaK07, Swe05, Swe02 and NiL07]. In practice, quasi-identifiers themselves are often targets of attack. Address, emails and phone numbers are considered quasi-identifiers if the adversary was a marketer, email spammer, or a telemarketer so releasing any one of those quasi-identifiers unperturbed may result in a breach in privacy. PriSQL treats all data as either sensitive or non-sensitive information. However, it is up to the data collector to define this appropriately within their own domain. The privacy of sensitive values is guaranteed to be protected by the privacy policy agreed upon by the data providers.

2.4.2 Concept Hierarchy

One way of protecting privacy is to generalize the information. Generalization alters or obscures the data in a way that provides less information while maintaining some level of truthfulness in the result. To generalize data, the notion of a *concept hierarchy* must be introduced. Concept hierarchies are defined as a sequence of mappings from a set of low-level concepts to higher-level ones that have more general forms [HaK01]. Generalization takes data stored as low level concepts and transforms it into higher level concepts. Release of data at higher concept levels increases the privacy protection of the participants since less information can be inferred. A concept hierarchy is shown in Figure 3. Concept hierarchies are not limited to categorical domains such as "education". Information stored as numerical values may also be placed into a concept hierarchy.

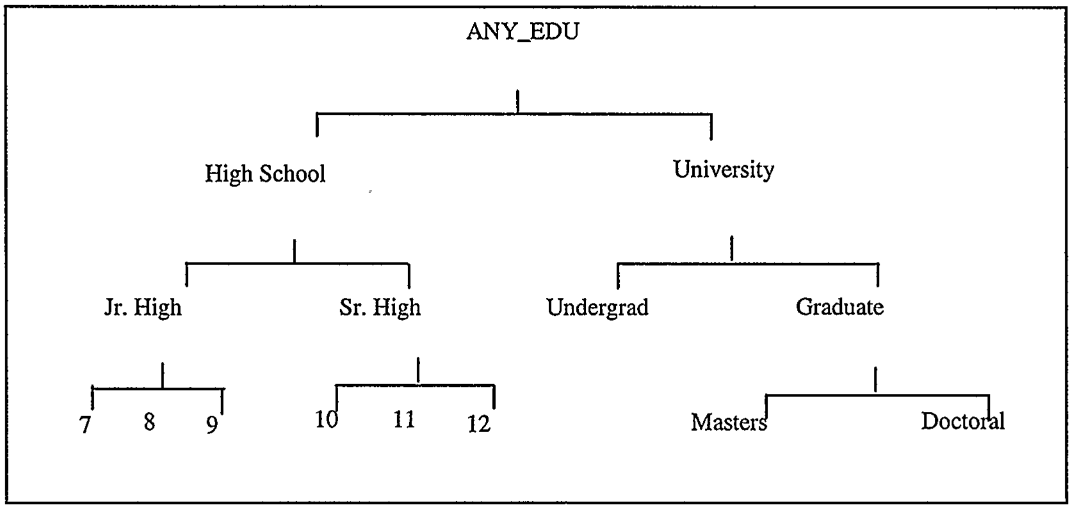


Figure 3 – Concept Hierarchy of the Education Domain

2.4.3 P-Level Protection

The notion of defining privacy using concept hierarchies was first introduced by Xiao and Tao [XiT06]. Xiao and Tao describe a privacy protection mechanism where an individual's information privacy is preserved if data released, pertaining to that individual, is at a level with which the individual is comfortable. Williams and Barker formally define a model for such a concept hierarchy-based approach to privacy preservation known as the P-level model [WiB08]. A *P-Level* is the degree of generalization, x, of a piece of sensitive information A, such that, $x = \{0,1,2,...,h\}$ and h is the highest level of generalization possible on attribute A. A P-level of 0 indicates that the data has not been generalized; a P-level of h is the highest level of privacy possible for a particular attribute's domain. A comparison between the relationship of a concept hierarchy and p-levels is shown in Figure 4. P-level privacy allows the privacy requirements of each attribute to be expressed in terms of P-levels where P is the level of detail that will be released.

Formally the P-level model is defined as follows:

Let $T = \{t_1,t_2,t_3,...,t_n\}$ denote a table with *n* records

Let $A = \{A_1,A_2,A_3,...A_m\}$ represent the set of all attributes in T.

All possible values for A can be represented as a hierarchy of concepts ordered from general concepts to specific concepts such that: $A_j^{Ph} \rightarrow A_j^{P(h-1)} \rightarrow ... \rightarrow A_j^{P0}$. Therefore, the privacy requirements of table T can be represented as the P-level for each attribute in T. In PriSQL, all identifiers and sensitive attributes will have an assigned concept hierarchy for generalization as well as a corresponding P-Level for data release.

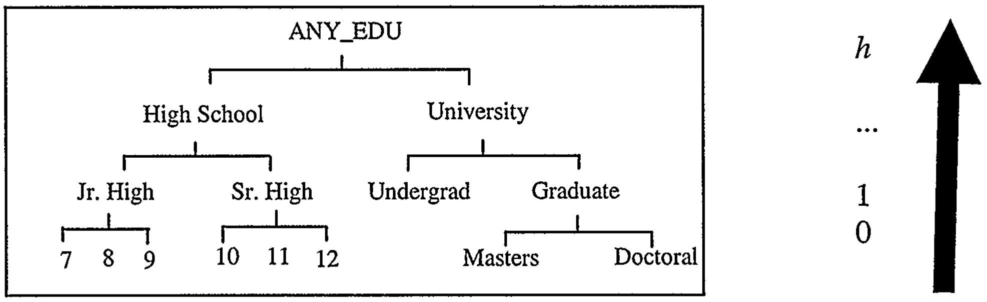


Figure 4 – Concept hierarchy and P-level of Education Domain

2.4.4 Generalization-based Privacy

Using conceptual hierarchies for privacy is not unique to P-level protection. There is a plethora of work in the data anonymization community which uses conceptual hierarchies as a method for anonymizing information [XiT06, MaK07, Swe05, Swe02, NiL07, WoL06 and BaA05]. This group argues that release of non-identifiable data is not a privacy violation since none of the individuals within the dataset are known. Prior to publishing any dataset, the identifiers are generalized such that none of the rows within the set is unique. The two most prominent versions of privacy aware data anonymization are k-anonymity and l-diversity. These two data anonymization methodologies have been established as a privacy standard in many government legislations as well as being accepted as being privacy best practices [Ohm09 and Ber06].

2.4.4.1 K-Anonymity

Sweeney defines K-Anonymity as:

> Let T $(A_1, A_2, \ldots, A_n)$ be a table and QI_T be the quasi-identifier associated with it. T is said to satisfy *k*-anonymity if and only if each sequence of values in T $[QI_T]$ appears with at least *k* occurrences in T $[QI_T]$. [Swe05 and Swe02]

To protect the privacy of the data providers, any identifying information related to each individual must be anonymized to the point that its row cannot be identified. K-anonymity achieves this by requiring that any identifying information be anonymized to the point where an individual data provider's identity within the dataset can be predicted at a rate of no greater than 1/k. While there are numerous approaches to transform a dataset into its k-anonymous equivalent, the most common mechanism used is a concept hierarchy to combine identifiers into higher level concepts [FuW07, Swe05, Win02, and BaA05].

2.4.4.2 l-diversity

Research has shown that there is vulnerability in the k-anonymity model [MaK07]. This occurs when the values for a sensitive attribute lack significant variation.

Since k-anonymity only generalizes quasi-identifiers, when a sensitive attribute has no variation the privacy of the data provider is violated. The private information of the data provider will be known to the recipients even if their row within the table cannot be identified. Exploitation of this weakness is known as a homogeneity attack. *l*-Diversity prevents such an attack from happening.

Machanavajjhala *et al.* define l-diversity as:

> Each set of unique sequences of values in T[QI_T] is called a Q-block. A Q-block is *l*-diverse if it contains at least *l* "well-represented" values for the sensitive attribute S. Table T is *l*-diverse if and only if every Q-block is *l*-diverse. [MaK07]

l-Diversity prevents a homogeneity attack simply by ensuring that there are at least *l* different values for each sensitive attribute within each Q-block. This is known as distinct *l*-diversity. However, uneven distribution of sensitive values limits the models ability to protect privacy. In other words, if the volume of sensitive attributes is skewed towards one value, the recipients of the dataset will still know the private information of a data provider with a high degree of certainty. To prevent this from occurring, Machanavajjhala developed two other *l*-diversity models: entropy and recursive l-diversity [MaK07]. These two models ensure that there is an evenly distributed variation of sensitive values within each Q-block. Having evenly distributed sensitive values prevents the adversary from knowing the sensitive value of a victim in a high degree of certainty.

2.5 Visibility

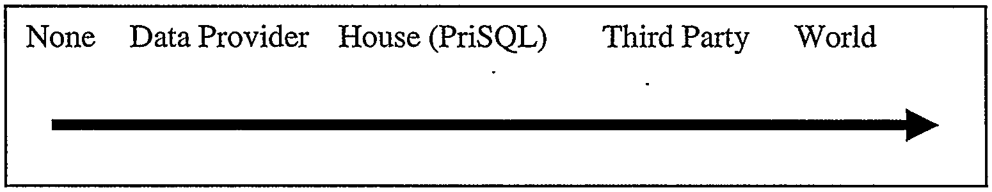


Figure 5 – Visibility Axis

The visibility parameter refers to who can view and use the data provided by the data provider. The visibility axis is shown in Figure 5 and has five main points: none, data providers, house, third party, and world [BaM09]. Each point on the axis represents a possible visibility parameter within the privacy tuple defined in Figure 1. *None* represents the notion that no one is allowed to view the data collect under any circumstances. This point, within the visibility axis, represents the most privacy afforded to the data. *Data provider* indicates either the individual to which the data pertains or the individual that provided the information; in most cases this is the same individual. *House* is the body that collected the data (corporations, governments, *etc.*) and is probably interested in analyzing it for further benefits. Users within the house may be further defined by the corporate hierarchy. For example, a marketing employee and a statistician are both members of the house but rather than permitting access for employees under the umbrella *house*, their visibility may be separated to support a more precise privacy definition. *Third parties* are entities that are not part of the house but may still require that the data be released to them (e.g., obligations by laws to release the information to the public or government). The *World* point indicates that any individual that wishes to access the data may do so without violating privacy. The ability to separate different visibility parameters within PriSQL will be discussed in the next sections.

2.5.1 Security

For decades, the notion of privacy, security and cryptography have been intermingled. While the database community is moving away from this approach, the use of this term is often ambiguous in the literature [WeS04, GaJ05, RaC01]. The RFID and biometrics community often argue that preventing unauthorized access naturally protects the privacy of their consumers. Current privacy research involves developing security protocols and technologies which prevent unauthorized users from accessing a system. These methodologies are robust and well established [ElN03]; PriSQL assumes the use of these security systems for user authentication. The aim of a privacy aware system such as PriSQL is to ensure that data is used for the reasons it was collected. Security measures ensure that only valid users have access to the system and its functionality. The ultimate goal of PriSQL is to provide a system in which a security violation does not automatically mean a privacy violation and vice versa.

2.5.2 Role-based Access Control (RBAC)

Visibility is probably the best understood privacy parameter within the SQL framework because of its similarity to security itself. The emergence of role-based access control in the late 1980s allowed corporations to manage large scale corporate system efficiently [FeK92, FeK07]. These corporate systems have a multitude of users which make granting and revoking individual user access a tedious process. The vastness of the problem caused Ferraiolo and Kuhn [FeK92] to outline a role based approach to assigning user permissions within a DBMS. A role is defined by a set of accessible objects and a set of individual users. A simple role relationship diagram is shown in Figure 6. This diagram indicates that Role A has two user members capable of executing transactions on object 1 and object 2. While not shown in the diagram, users are allowed to be members of multiple roles giving them access to more transactions within the DBMS.

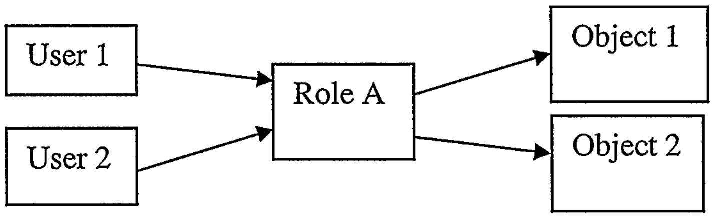


Figure 6 – Role Relationship

2.5.3 Privacy Aware Role Base Access Control (PRBAC)

Privacy policies and access control policies are closely related, since each governs access to the same piece of data within a DBMS. This common interest has led to the development of a series of privacy aware role based access control models. These models utilize the mechanisms of RBAC within the DBMS and extend it to facilitate privacy principles. There are four PRBAC models that combine to form a robust privacy enforcement definition; these are the core, hierarchy, condition and universal models [NiL07, NiT07]. Core PRBAC allows the simple definition of privacy policies. Hierarchical PRBAC allows a hierarchical organization of data, and purpose. Conditional PRBAC permits the definition of more complex privacy policies through the use of AND, OR and NOT connectors. Universal PRBAC is the combination of the other three models and presents its own set of conflicts.

2.5.3.1 PRBAC Privacy

PRBAC assigns a privacy tuple to each role similar to our privacy definition. PRBAC's privacy specification includes: users, roles, data, purpose, actions, obligations and condition. This definition is consistent with PriSQL's privacy specification with the exception of conditions and obligations. Conditions provide PRBAC the ability to define the state of the data prior to the action being permitted. i.e. "Users must be 18 years or older for data to be analyzed". Obligations are actions that must occur after the usage of the requested data (i.e. an email will be sent if your data is analyzed). It is argued [NiL07] that conditions and obligations are part of everyday privacy requirements. PriSQL considers conditions to be implicitly included in the definition of *purpose*. Using the previous example, "Users must be 18 years or older for data to be analyzed", the intended purpose of collecting data is data analysis. A data provider either agrees to this purpose or refuses to provide their information. Thus, when data is collected the information inside already fulfills the conditions imposed otherwise it will not be stored. Obligations were not considered as a privacy parameter within PriSQL. The reason is that obligations are business requirements rather than privacy requirements. Once information has been revealed to the *data requestor*, the usage of that data is beyond the control of any DBMS. In PRBAC, obligations are enforced after the data is revealed to the data requestor. PriSQL argues that any privacy violation to the data providers would have occurred by the time obligations are met.

2.5.4 PRBAC data access

Utilizing the power of RBAC, PRBAC assigns privacy data permissions to the different roles within the DBMS. Users are granted access to privacy sensitive information by requesting membership into a specific role. Users then inherit a predefined list of privacy data permissions as part of the roles they are members of. This role based privacy data permission relationship is shown in Figure 7. This privacy access method simplifies the data access procedures for the users of the system. However, this also introduces possible conflicts as a result of conflicting privacy permissions. Since users are allowed multiple roles, different roles may have conflicting privacy requirements. When a system cannot differentiate which role a user is acting upon there are three basic options, assume deny, assume-permit and not allowing conflicts to occur. Ni *et al.*[NiT07] outlined their own conflict resolution algorithm to deal with possible conflicts however their focus is on conflicting conditions and not conflicting privacy permissions. In *assume-deny* conflict resolutions, users are denied access if there is any ambiguity with their access privileges. Often though, these resolution mechanisms promote the creation of a single new role. This role is created with complete access to all the required data objects. For *assume-permit* conflict resolutions schemes, a user will be permitted access if any of their roles permits it. Both scenarios have potential privacy violation; because the system does not know the purpose of this data action, access may be granted simply because the data requestor has the privacy permissions through a different role within the corporation. PriSQL's data access model prevents this situation from occurring and is described in Chapter 4. PriSQL uses the users' identity (User /

Role) as part of a privacy tuple which is supplied with the query for privacy access validation rather than relying solely upon it.

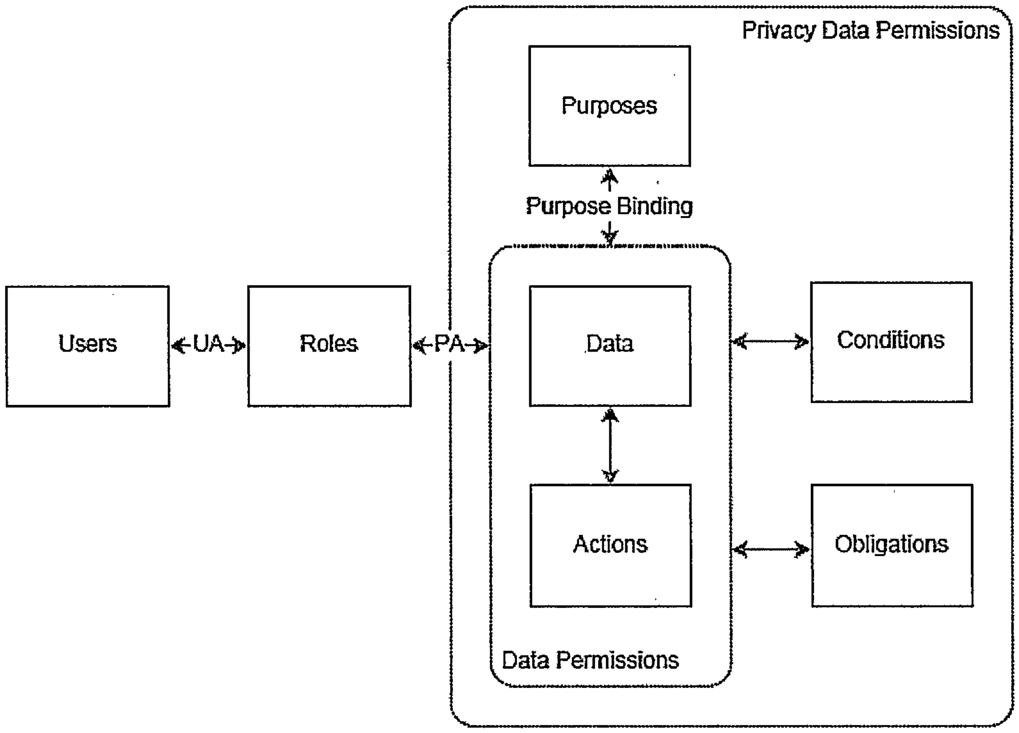


Figure 7 – PRABC Access Model [NiL07]

2.6 Retention

2.6.1 Data Retention

Currently, one of the major privacy debates is focused around *data retention* or the length of time that sensitive data should be stored within a database. The cost of storage and hard drives is decreasing rapidly, making storage of information relative cheap. This allows corporations to store information indefinitely. Privacy legislation in Europe and the United States of America explicitly limit the amount of time that corporations can store private data. In 2007 and 2008, Google Inc. announced that it would be reducing the data retention time of users sensitive search data to comply with new privacy legislation [BBC07, CWC08]. PriSQL requires that a data retention period be specified for sensitive data entered into the system. This parameter allows PriSQL to conform to legislation as well as protect the privacy of data providers.

2.6.2 Privacy Retention

Data retention aims to limit the amount of time sensitive data is stored within the database; privacy retention aims to limit the length of time privacy data permissions are valid for. In many privacy scenarios, a time limit on the privacy data permissions is warranted. Some privacy permission should be restricted by either number or uses, or expiration date. Using the scenario describe in Section 2.3.1. Joyce provided her credit card information for the purpose of purchasing a gift. Her expectation is that her credit card information will only be used once. Associating a privacy retention parameter with her information ensures that the information cannot be access multiple times without further permission being requested first. Placing expiry dates on data permissions is another possibility. Current email legislation allows mass emails by corporations to existing customers. These emails are continuous until you opt-out. For example, because Joyce previously purchased a gift from Facebook, Facebook is free to send emails to Joyce about other gifts she may be interested in. These emails continue until Joyce explicitly requests that the emails discontinue by opting out of commercial contact. A better option would to be to limit the date that Facebook is allowed to send emails. Limiting Facebook, to three months of permitted commercial contact would be a better option. The expiry date on data permission can be extended by further purchases or through another privacy agreement. Otherwise the emails would automatically stop after three months. By placing a time limit on privacy policies themselves, corporations will be forced to renew privacy approvals rather than abusing the approvals previously attained.

Chapter Three: PriSQL

The Structured Query Language (SQL) is used for information querying, modifying data, and managing databases within relational database management systems (DBMS). SQL in its various forms first appeared in 1970s [ChP81] and was adopted into the International Standards Organization (ISO) in 1987 as SQL-87. Over the years, new functionality and concepts have been added to SQL such as recursive querying, expression matching and support of XML formatted data. The current ISO version of SQL is SQL:2008. While SQL:2008 defines the SQL standard, many commercial database management systems have their own variations. These databases include DB2, Oracle, SQL Server, PostgreSQL and My/SQL; which all have their own SQL dialect. The examples provided in this chapter refer to the My/SQL 6.0 formatting of SQL. My/SQL was chosen as the base language in this work because of its open source and free license agreement. Although examples are shown in My/SQL, the concepts introduced and discussed are applicable to the other SQL dialects.

PriSQL is a privacy extension of the structured query language. PriSQL enhances SQL so it is able to comprehend the privacy principles described in Chapter 2. Chapter 2 described four major privacy parameters: generalization, visibility, purpose and retention. To support PriSQL's definition of privacy three new privacy keywords are introduced to SQL as well as a generalization parameter. These keywords are *retention*, *expires*, and *because.* To support the privacy information within a DBMS two new system catalogues are also introduced, SysPrivacyPolicies and SysTranformFunctions. These tables are described in detailed in the following sections of this chapter along with how each keyword is used.

3.1 System Catalogues

System catalogues within a database management system are responsible for the management of information about the database itself. Information such as the database settings, number of tables, number of indexes, size of each table, type of data, size of memory to allocate, *etc.* are all stored within these tables. PriSQL introduces two new system catalogues to support the privacy preferences, SysPrivacyPolicies and SysTransformFunctions. SysPrivacyPolicies is responsible for storing a machine-level interpretation of the privacy policy. SysTransformFunctions stores information related to the data generalization methodology prior to its release.

3.1.1 SystemPrivacyPolicies

The privacy policies of a corporation are stored within the SysPrivacyPolicies table. Once the relational database is designed and created, a set of tables and columns will exist within the DBMS. Each column may or may not store sensitive data; however those columns that do, must be protected according to the privacy policy. Figure 8 shows the parameters of the privacy catalogues. SysPrivacyPolicies has seven configurable parameters: table, column, operation (op), visibility, generalization, purpose, and retention. Each row within SysPrivacyPolicies represents a single privacy permission granted to a user on a specific piece of sensitive data for a specific purpose described in the privacy policy agreed upon by the clients of the database.

Values stored in table and column attributes specify an attribute of a table about which the privacy policy concerns. The value stored in op specifies the privilege that a user or group of users (stored in visibility) can exercise on the attribute but only if they have the legitimate intention (stored in purpose). The value stored within generalization is used to specify the p-level anonymization that should be applied on the attribute before the disclosure. The anonymization function is stored in SysTransformFunctions (Section 3.1.2). Retention is used to indicate the period of time for which the privacy row is valid.

3.1.2 SystemTransformFunctions

The SysTransformFunctions catalogue stores the information relating to the *privacy transformation functions* of each sensitive column. The catalogue consists of three attributes: table, column, and generalization function (genFunction). Values stored within table and column indicates the specific attribute which must pass through the transformation function prior to data being released. Privacy transformation functions are black box functions that must conform to two rules. The first rule is that it must take in two arguments, a privacy level parameter and a set of data values. The second rule is that it must return a set of values which conform to the privacy level indicated. Section 2.4 describes different generalization methods which would preserve privacy at an indicated privacy level. Currently the list of supported generalization functions include: remove, date of birth, address, and phone number and is limited to p-level generalizations. While the list of generalization functions currently provided is limited, other could be readily envisioned. The pseudo code of these privacy transformation functions are provided in Appendix B.

SysPrivacyPolicies

table	column	op	visibility	generalization	purpose	retention

SysTransformFunction

table	column	genFunction

Figure 8 – Privacy Catalogues

3.2 Data Definition Language (DDL)

3.2.1 CREATE TABLE

In SQL, database users can create data containers with the CREATE TABLE statement. We assume only security officers (e.g. database administrators) are privileged to execute this statement. The following is a simplified syntax of the CREATE TABLE statement [SUN09]:

```
CREATE TABLE tbl
        (col dataType [column_options] ...,
        [table_options] ...);
```

where *tbl* is the name of table being created and *col* is the name of a column included in table *tbl*. D*ataType* indicates the type of data stored in column *col*; examples of data types are INT, CHAR, DATE, *etc.* Column_options allow different rules to be applied to column *col* such as providing the column with a default value, and ensuring the column is not NULL. T*able_options* are rules which can be applied to *tbl*. Examples of these rules are limiting the minimum and maximum number of rows in *tbl* or setting a password for table access.

PriSQL follows a similar format for the creation of data containers. PriSQL extends the CREATE TABLE command to include two optional parameters: one for *generalization* and one for *retention*, as follows:

```
CREATE TABLE tbl
        (col dataType [column_options] [generalization] ...,
        [table_options] ...,
[retention {Number of accesses or Length of Time}]);
```

Where *tbl*, *col*, *dataType*, *column-options*, and *table-options* are as described previously. *Generalization* is an optional parameter for column *col*. Any column storing sensitive information must provide a generalization function so PriSQL can protect the privacy of the data stored within it. The provided generalization function is used to anonymize the data of the column, based on a privacy tuple (Section 3.1.1). R*etention* is also an optional parameter that specifies the expiry condition of each row of table *tbl*. This clause allows PriSQL to support the data retention privacy requirement describe in Section 2.6.1. The retention period provided may be in the form of a time period or the number of accesses permitted. When a time period is provided, tuples within the table *tbl* may not be stored for longer than the time period specified. If access to a tuple is limited by occurrences then the data may not be accessed more than the specified retention parameter.

3.2.2 ALTER TABLE

In evolving companies the requirements of the database and the privacy policy may change from time to time. Altering a data table by adding or removing columns, while rare, must be permitted. For a table to be altered, that operation must be within the privacy catalogue. This indicates that the original privacy policy permits the altering of the privacy requirements in a database. Removing columns from a data table has few foreseeable privacy violations. Since sensitive data within that column is being removed from the database, the privacy of the data providers is being increased. Adding in new columns will follow a similar format as the CREATE TABLE. If the new column will store sensitive data, a generalization function for that column must be provided. After the column has been added, new privacy tuples for that column must be imported into the privacy catalogue by the privacy administrator. PriSQL syntax for altering data tables is as follows:

ALTER TABLE *tbl*

ADD/DROP/CHANGE *column* [**Generalization**]

[**BECAUSE** *"reason"*];

3.2.3 INSERT INTO (Privacy Catalogue)

This section describes how the SQL INSERT command is extended to facilitate the principals of privacy. The major aspect of privacy involved in the PriSQL insert statement is retention. There are two concepts of retention supported with this extension: privacy retention and data retention. To facilitate the requirements of retention we introduce a new extension to SQL called the EXPIRES parameter. This parameter is used to indicate the period of time a specific tuple is stored within its table. In order to populate the privacy catalogue, a series of insert statements is executed. Examples of insert statements into the privacy catalogue are shown in Chapter 5. The EXPIRES clause used in association with the privacy catalogue is to indicate the period of time that a privacy tuple is valid within PriSQL (Privacy Retention). While current privacy policies do not have the notion of expiring access, this type of limiting access should be introduced. This concept was motivated in Section 2.6.2.

The standard insert into statement is shown below:

```
INSERT INTO tbl VALUES (...);
```

PriSQL introduces an optional EXPIRES parameter that is provided after the values have been inserted. Inside the data table, *tbl*, a retention column is stored along with the tuple.

```
INSERT INTO tbl VALUES(...)
EXPIRES { Number of accesses or Length of Time };
```

3.3 Data Manipulation Language (DML)

This section describes how the SQL data manipulation language (DML) is extended to support the privacy policies. Recall that the proposed concepts to enrich the SQL security model are purpose, visibility, generalization, and retention. Generalization can be automatically applied when a column is being accessed. Similarly, visibility and retention can be verified when a column is being accessed. If, due to the visibility or retention parameter, a column is no longer accessible for the corresponding operation and purpose, the operation is rejected.

To support purpose, we must extend all DML operations by adding a new clause, BECAUSE, users (or processes) specify what their purpose is for attempting to execute that query. If the corresponding user, operation, and purpose are found in the privacy catalogues (Figure 8), the operation may continue based on the corresponding generalization and retention.

PriSQL's DML syntax allows the BECAUSE keyword to be an optional parameter. In cases where users are accessing non-sensitive data, the BECAUSE keyword is unnecessary, this format is consistent with the MY/SQL 6.0 standard. When a user is attempting to access sensitive data they must have a purpose for the query, this purpose must match the reasons specified in the privacy policy of the corporation. If a data requester's reason for accessing the data is not consistent with the privacy policy, the query is rejected.

3.3.1 SELECT

A SELECT statement is the most common data access command within SQL. It is used to retrieve information stored within the relational database. Privacy violations most often occur with a SELECT statement so they pose the greatest threat to user privacy. The PriSQL SELECT statement is extended in the following format:

```
SELECT   column, ...
FROM     table, ...
[WHERE condition], ...
```

[**BECAUSE** *"reason"*];

The PriSQL SELECT statement allows for an optional BECAUSE statement consistent with the reasoning described in the previous section. A select query may have multiple tables and columns within its statement however only one purpose will be permitted. Each column being accessed must have a matching privacy tuple within the SysPrivacyPolicies (i.e. a valid purpose, visibility, and retention for the select operation). For cases where there is only a partial match (i.e. some columns fail the privacy requirements), the query may be rejected completely, or the minimal set of valid columns can be returned. Rejecting the query completely provides more privacy protection for the data providers as it prevents an exhaustive existential search by an adversary for columns of information to which they have access. However, a complete rejection will also greatly decrease the usability of the system for valid users. PriSQL's chose a complete rejection when this case occurs since privacy was the main focus of our system.

3.3.2 INSERT INTO

Prior to the retrieval of information, data must be populated within the tables. To populate the database INSERT statements are used. To incorporate privacy requirements the same EXPIRES keyword describe in Section 3.2.3 is used. The retention period provided is the period of time that data will be stored within a data table. The [*retention*] parameter described in Section 3.2.1 (Create Table) is used to indicate the maximum period of time that data may be stored in the table. Here, the EXPIRES parameter is optional and used to indicate the period of time a specific tuple is stored in a table. The period of time provided in the EXPIRES clause must be less than or equal to the retention clause specified during the create table phase. When the EXPIRES keyword is not used, the tuple will have a retention period equal to the maximum retention period permitted for the table.

3.3.3 DELETE FROM

The delete clause within SQL may not seem to pose a privacy risk since it is removing sensitive data from the system. However unsupervised access will allow existence attacks upon the system. Adversaries can exhaustively attempt to delete a single tuple from the system and fully identify the single individual. Thus, the delete operation is also included in the set of supported privacy conscious DML commands. A user may perform a delete operation on a table only if there is a tuple in the privacy catalogue that allows them to do so. The delete command has the following scheme:

```
DELETE FROM tbl
[WHERE condition]
[BECAUSE "reason"];
```

3.3.4 UPDATE

Updating personal information is an essential part of everyday business. Corporations rely on up to date information to make the most accurate decisions possible. An UPDATE operation within SQL allows users to identify information within a data table and correct it as necessary. The privacy exposure to data providers is similar to a combination of the possible privacy violations in a SELECT, DELETE and INSERT statements. Therefore, the PriSQL UPDATE command is added to the set of supported privacy DML commands. Any user wishing to UPDATE information of a sensitive table may only do so if they have a valid privacy specification. The update command has the following scheme:

```
UPDATE tbl
SET column1 = value, column2 = value2,...
[WHERE condition]
[BECAUSE "reason"];
```

3.4 Summary of PriSQL Extensions

SQL	PriSQL

CREATE TABLE *tbl*
(*col dataType* [*column_options*] ...,
[*table_options*] ...);
	CREATE TABLE *tbl*
	(*col dataType* [*column_options*]
	[*generalization*] ...,
	[*table_options*] ...,
	[*retention* {Number of accesses or Length of Time}**]**);

ALTER TABLE *tbl* ADD/DROP/CHANGE *column*;	ALTER TABLE *tbl* ADD/DROP/CHANGE *column* [***generalization***] [**BECAUSE** "*reason*"];
INSERT INTO *tbl* VALUES(...);	INSERT INTO *tbl* VALUES(...) **EXPIRES** { Number of accesses or Length of Time };

SELECT column,...
FROM *tbl*, ...
	SELECT *column*, ...
	FROM *tbl*, ...

[WHERE *condition,…*];	[WHERE *condition*], … [**BECAUSE** "*reason*"];
DELETE FROM *tbl* [WHERE *condition*];	DELETE FROM *tbl* [WHERE *condition*] [**BECAUSE** "*reason*"];
UPDATE tbl SET column1 = value, column2 = value2,… [WHERE condition];	UPDATE tbl SET column1 = value, column2 = value2,… [WHERE condition] [**BECAUSE** "reason"];

Chapter Four: PriSQL System

A proof of concept system is developed to demonstrate the practicality and potential of a privacy aware database management system. However, not all concepts discussed in the previous sections have been implemented. Only the essential components required to demonstrate PriSQL's usefulness have been included. PriSQL was implemented as a wrapper around an existing database management system (MY/SQL 6.0) to maximize the utilities of existing systems. The PriSQL wrapper consists of four major components, the Language Parser, the Privacy Enforcement Component, the Database Connector and the Database Maintenance Component. These four components combine to provide a DBMS that offers privacy as a first order principle. An outline of the system architecture is shown in Figure 9.

4.1 Privacy Enforcement Algorithm

One of the major goals of PriSQL is to make certain that the privacy of data providers cannot be violated. A privacy violation within PriSQL occurs when a user of the system is able to access information without valid privacy permissions. To prevent privacy violations a robust algorithm was developed to verify the privacy permission of the data requestor prior to the execution of any queries. This algorithm is outlined in Figure 10.

When a PriSQL query is submitted for execution, PriSQL will first retrieve the identity of the data requestor. Then based on the query, the data being requested, the SQL operation and the access purpose are inferred. Base on this privacy information, PriSQL is able to verify that the data requestor has the privacy permission to execute the query.

The privacy principles of purpose, visibility and retention are all enforced. For data retrieval queries, generalization is also a privacy parameter. Therefore, prior to any results being display they are generalized based on the generalization function initially chosen. The remainder of this chapter outlines how each component combines in order to execute and support this privacy enforcement algorithm. Figure 11 shows the interaction diagram of PriSQL during the execution of a query.

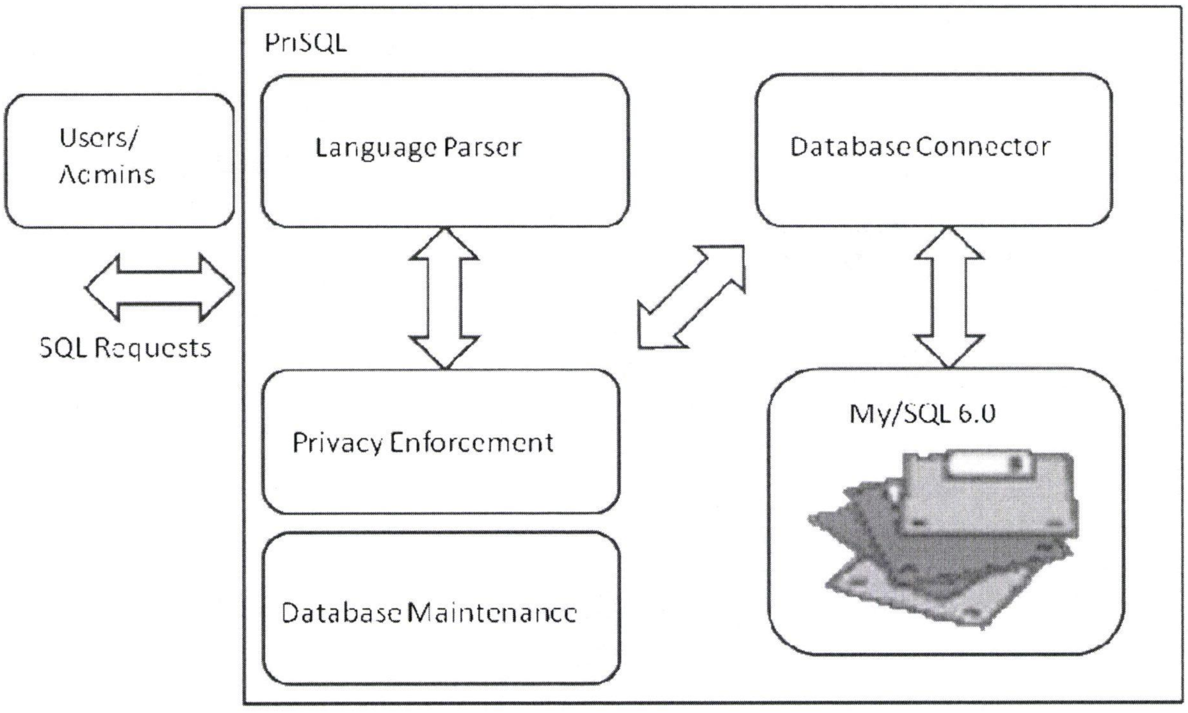


Figure 9 – PriSQL Architecture

```
Input: PriSQL Query, SysPrivacyPolicies, SysTransformFunctions
Output: Results of Query
Algorithm:
    1. Retrieve Authoirzation ID of the data requestor;
    2. Result ::= 0;
    3. If <Authorization ID, table name, column, name, operation,
       purpose> exists in SysPrivacy Policies
           a. Result ::= Query Database ie. Execute Query
           b. Result ::= SysTransformFunction(Result);
    4. Return Result
```

Figure 10- Algorithm for Extended DML Operations within SQL

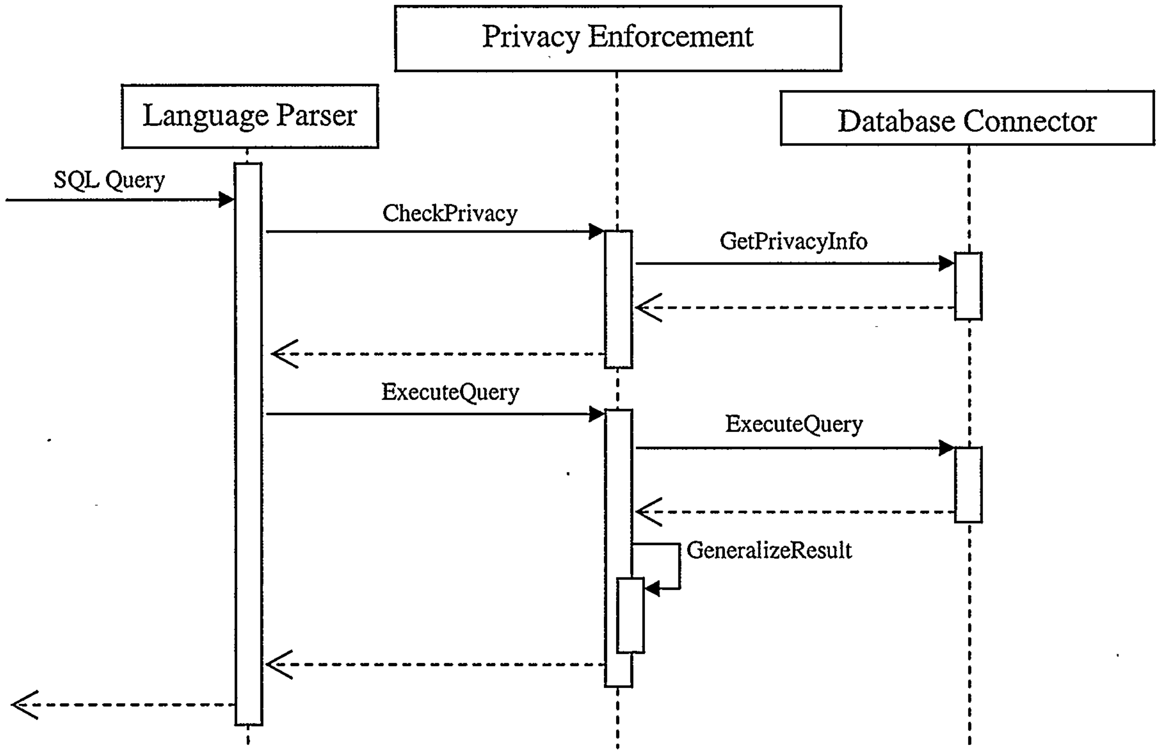


Figure 11 Sequence Diagram for a PriSQL Query

4.1.1 Language Parser

The Language Parser is responsible for two major functions. The first is to present a user interface for the users. The user interface is a text based console which is capable of inputting and displaying text. Once a SQL command has been entered, the language parser is responsible for separating PriSQL queries into two sections, a privacy component and the SQL query itself. The privacy component of a query consists of determining the privacy specification of the data request. For example, Joyce wishes to execute the following PriSQL statement.

```
SELECT First_Name, address
FROM Personal_Info
WHERE Data_Of_Birth = "Oct-01-1991"
BECAUSE "Birthday Card Promotion"
```

Based on the above query, the privacy specification created by the language parser would be the following:

Privacy:< "Birthday Card Promotion", "Joyce", {Personal_Info}, {First_Name, address}, SELECT, sysdate>

The privacy specification of the query is then passed to the Privacy Enforcement Component (Section 4.1.3). Once the privacy permissions are determined to be adequate, the language parser will submit the SQL query to the Privacy Enforcement Component for execution. Results of the query are displayed to the data requestor through the user interface.

4.1.2 Database Connector

The database communicator is responsible for communicating to the database management system. The database communicator is an ODBC connector written in C++ linking PriSQL components to a My/SQL 6.0 database. This design was chosen to demonstrate the transferability of the PriSQL wrapper. PriSQL would be fully compatible with other DBMS system simply by replacing or altering this component. The database connector has two essential stored procedures. The first stored procedure is used to execute and commit SQL queries. All DDL and DML commands are executed through this procedure with the exception of a SELECT command. The second stored procedure is used to execute SELECT statements and retrieve the results in an interpretable format. Aside from these two stored procedures, a small set of customized functions were developed to support privacy enforcement. This set of procedures included existence checks for privacy tuples, and retrieval of p-level generalizations.

4.1.3 Privacy Enforcement Component (PEC)

The Privacy Enforcement Component is responsible for privacy validation and DDL/DML privacy enforcement. Privacy validation is tasked with checking that the privacy tuple of the query exists in the privacy catalogue; meaning that the data requestor has the authority to access the data. The DDL/DML privacy enforcement is responsible for managing and maintaining the proper status of the privacy catalogue as well as the generalization of data on applicable DML operations. Details about privacy validation and DDL/DML privacy enforcement are described in the following sections.

4.1.3.1 Privacy Validation

Recall from Section 4.2.1, that a privacy specification will be provided as the input for this procedure. The privacy validation procedure will test whether a user has the data privacy permissions to access each column of data requested. If the user does not have the correct privacy data permissions, the query is rejected. This procedure is responsible for ensuring that PriSQL fulfills its obligation of protecting the purpose, visibility, and retention privacy concepts. Enforcement of the generalization privacy concept is described in Section 4.1.3.5. The basic format of the privacy specification is as follows:

Privacy :< Purpose, User, {Tables}, {Columns}, Operation, Date>

This procedure will then check that each column within the privacy specification has a privacy tuple within SysPrivacyPolicies that contains the specified purpose, user, operation and the date is within the retention period. If a single column does not have the privacy tuple then the entire query is rejected.

4.1.3.2 CREATE TABLE

Many of the data tables within a DBMS are not going to contain any sensitive information. The creation of these tables has no affect on the privacy system tables and can be executed directly. Tables containing private information must be created using PriSQL syntax in order for PriSQL to enforce the privacy requirements. The privacy enforcement component differentiates standard SQL syntax and PriSQL syntax by identifying the optional generalization functions or the retention parameter within a

CREATE TABLE command. For tables that contain generalization functions associated which each attribute, these generalization functions are placed into the SysTransformFunctions privacy catalogue. Tables that specify retention as a privacy parameter will have a privacy tuple inserted into the SysPrivacyPolicies indicating the maximum retention time that tuples within the table can be stored. Prior to the rest of the query being sent to the database connector, a hidden column is attached to the table for the purpose of storing the retention periods of each tuple.

```
CREATE TABLE tbl
        (col dataType [column_options] [generalization] ...,
        [table_options] ...,
        [retention {Number of accesses or Length of Time}]);
```

4.1.3.3 ALTER TABLE

There are three different ways to alter a table: add, drop and change. Altering tables and columns which have no sensitive data has no privacy considerations within PriSQL. These queries will be directly passed to the database connector for execution. Tables that contain sensitive information will be supported in a similar method as the CREATE TABLE command. When an add column query is submitted, the PEC will identify the generalization function provided and insert a tuple within the SysTranformFunctions table. To drop a sensitive column the PEC must perform some clean up and maintenance of the privacy catalogues. First, the tuple storing the generalization function of the attribute is removed from the SysTransformFunctions. Since the column will no longer exist within the database, the privacy permissions are also obsolete. The privacy tuples within the SysPrivacyPolicies associated with the column being drop are then removed. Once the privacy catalogues have been cleaned up, the alter command is submitted to the database connector for execution and commitment. Alter table commands that request column changes are either changing the column attributes or the generalization function. Requests that change the generalization function will invoke a maintenance procedure identical to a drop column request. The tuple storing the generalization function within the SysTransformFunctions table will be removed along with all privacy permissions related to the column in the SysPrivacyPolicies. The new generalization function is then inserted into the SysTransformFunctions catalogue.

> ALTER TABLE *tbl*
>
> ADD/DROP/CHANGE *column* [**Generalization**]
>
> [**BECAUSE** "reason"];

4.1.3.4 INSERT INTO

The insert command is used to populate both the privacy catalogues and the relational database. Populating the privacy catalogues requires database/privacy administrators to run the insert command. Database administrators are the only *trusted users* within the system. Inserting into the relational database is permitted by all users.

Inserting private data into a sensitive table/column, the PEC will require that the EXPIRES keyword is used. Once a retention period on the data has been established the PEC will check that the expiry date on the data is less than or equal to the maximum retention date of the table. Insert queries will be rejected if the date provided exceeds the maximum retention date of the table. Once the retention period has been verified to be correct then the retention period is appended to the insert and passed to the database connector for execution and commitment.

```
INSERT INTO tbl VALUES(...)
EXPIRES { Number of accesses or Length of Time };
```

4.1.3.5 SELECT

PriSQL SELECT queries have the greatest privacy considerations because it is intended to retrieve information from a database. PriSQL privacy definition contained a parameter for generalization of the data. This generalization means that data providers have agreed to release information about them but not necessarily in the micro-data format. Any PriSQL SELECT query must have its data obscured based on the data privacy permissions.

When a PriSQL SELECT query is received by the PEC, it is assumed that the data requestor has valid data privacy permissions to view the data based on visibility, purpose and retention. In order to enforce generalization, the PEC retrieves the p-level and the transformation functions (Section 3.1.2) for each column of data being retrieved. The SELECT query is then submitted to the database connector and the microdata of all results is returned to the PEC. The information from the database connector is returned on a column by column basis. As the results of each column are retrieved, they are submitted to the privacy function along with the p-level parameter. The result of the generalization function is used to replace the original microdata of the result. Once all sensitive columns have been processed, the result is returned to the language parser and displayed to the data requestor. Examples of this feature can be seen in Chapter 5.

```
SELECT   column, ...
FROM     table, ...
[WHERE condition], ...
[BECAUSE "reason"];
```

4.1.3.6 DELETE FROM

A delete command has the least amount of privacy overhead. Once it is determined the user has the privacy permissions to perform a delete command the query is simply forwarded to the database connector.

```
DELETE FROM tbl
[WHERE condition]
[BECAUSE "reason"];
```

4.1.3.7 UPDATE

As part of SQL, the update operation is responsible for providing a mechanism in which values within a data table can be altered. By having this function, it replaces the need to execute a delete and corresponding insert command. This aids in the ease of use and intuitiveness of SQL as a whole. The privacy overhead of supporting the update command is also minimal. If a user has the privacy permissions to execute an update command on a given table then the query will be forwarded to the database connector for execution. Otherwise the query is rejected.

```
UPDATE tbl
SET column1 = value, column2 = value2,...
[WHERE condition]
```

[BECAUSE “reason”];

4.1.4 Database Maintenance

The database maintenance component is to ensure that the retention requirements are being met. Both sensitive data and privacy tuples have expiry conditions that must be enforced. This component is tasked with the removal of expired tuples for space and risk exposure minimization. This component scans the database for any expired tuples and deletes them from the database as well as updating the retention period. This component is currently executed on a daily basis. Increasing the execution frequency of the database maintenance component will increase the precision of the retention parameter. However increasing the frequency will also increase the overhead of maintaining privacy information and affect business costs. Thus the balance between cost and retention precision will have to be decided on a per domain basis.

Chapter Five: IBM Case Study

The previous chapters of this thesis have described the privacy extensions SQL requires to incorporate privacy concepts. In this chapter, an active privacy policy has been interpreted and implemented using PriSQL. The goal of this chapter is to demonstrate the usefulness and applicability of PriSQL. IBM's online privacy policy is chosen as the running example[2]. Some sections of the original privacy policy are not discussed in this chapter because they were repetitive in nature. The omitted privacy policies deal with: Supplier Information, Cookies, Web Beacons and other technologies, online advertising, mergers and acquisitions, information security and quality. The omitted privacy policies are either conceptually addressed in the examples covered or not related to information privacy so are not applicable to this work.

5.1 IBM Privacy Policy

IBM first describes the data collection process for personal information within their privacy policy. From this paragraph we can interpret the personal information that may be stored within the IBM database. We create the Personal_Info data table and the PriSQL statement is shown below. Since it was not explicitly identified within the policy, an 18 month data retention and 36 month privacy retention period are assumed.

```
CREATE TABLE Personal_Info(
        Customer_ID int,
        First_Name varchar(50) Remove,
```

[2] The policy used was published on July 24 2009, and can be found at: http://www.ibm.com/privacy/details/us/en/

```
Last_Name varchar(50) Remove,
Address varchar(50) Anon_Address,
Email varchar(50) Remove,
Date_of_Birth Date Anon_DOB,
Phone_Number varchar(50) Anon_Phone,
Retention 18-Month
};
```

> **Supplementing Information**
>
> From time to time we may supplement information you give us via an IBM Web site with information from other sources, such as information validating your address or other available information about businesses. This is to help us maintain the accuracy of the information we collect and to help us provide a better service.

5.1.1 Data Supplementation

This paragraph of the privacy policy indicates how the information will be corrected and enhanced to maintain the accuracy of the database. Information will need to be updated 'from time to time'. To allow this within PriSQL a set of privacy tuples must be inserted into the SysPrivacyPolicies; rows 2 through 7 of Appendix B show the state of SysPrivacyPolicies after the insertion. Supplementing information is a privacy concern because the customer representative (Rep) must be able to locate and verify (SELECT operation) a data provider's information within the database. If there are any discrepancy, the customer represented is responsible for updating the personal information. To determine the specific details about a user, the generalization level of the data must be display without any abstraction. Therefore each p-level for the privacy tuples is specified at 0. The visibility of the privacy tuple is assigned to customer representatives (Rep) in order for them to accomplish this requirement. The intended purpose of this privacy tuple is to supplement/correct any data that may no longer be correct (UPDATE operation). The privacy tuple required to support information supplementation is shown below along with the corresponding PriSQL commands.

Privacy := < "Supplement", 0, Customer Representatives, 36 Months>

```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"First_Name","Select","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Last_Name","Select","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Address","Select","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Email","Select","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Date_of_Birth","Select","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Phone_Number","Select","Rep","0","Supplement") EXPIRES 36-Month;

INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"First_Name","Update","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Last_Name","Update","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Address","Update","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Email","Update","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Date_of_Birth","Update","Rep","0","Supplement") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Phone_Number","Update","Rep","0","Supplement") EXPIRES 36-Month;
```

5.1.2 Transactions

> **Fulfilling your transaction request**
>
> If you request something from an IBM Web site, for example, a product or service, a callback, or specific marketing materials, we will use the information you provide to fulfill your request. To help us do this, we may share information, with others, for instance, other parts of IBM, IBM's Business Partners, financial institutions, shipping companies, postal or government authorities (for example, Customs authorities) involved in fulfillment. In connection with a transaction we may also contact you as part of our customer satisfaction surveys or for market research purposes.

In the event of a transaction request with IBM, IBM will require your contact information and shipping address. For these reasons the P-level of those attributes were specified to 0 or the microdata level. There are no foreseeable reasons that a Sales Associate will need to know your date of birth to complete the any transaction, therefore the values of that attribute are obscured by setting its p-level as 2. Appendix B contains the algorithm for generalizing an individual's date of birth and shows that by setting the p-level to 2 it will not be displayed. The privacy tuple and PriSQL commands of sales transaction is as follows:

Privacy := <"User Sale",{2-Date_of_Birth, 0-Others}, Sales, Not Specified>

```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"First_Name","Select","Sales","0","U-Sales") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Last_Name","Select","Sales","0","U-Sales") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Address","Select","Sales","0","U-Sales") EXPIRES 36-Month;
```

```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Email","Select","Sales","0","U-Sales") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Date_of_Birth","Select","Sales","2","U-Sales") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Phone_Number","Select","Sales","0","U-Sales") EXPIRES 36-Month;
```

5.1.3 Marketing

> **Marketing Use**
> The information you provide to IBM on certain IBM Web sites may also be used by IBM and selected third parties for marketing purposes. Before we use it, however, we will offer you the opportunity to choose whether or not to have your information used in this way. To opt-out of receiving further marketing emails from IBM, please follow the unsubscribe instructions included in each email.

The IBM privacy policy indicates that IBM may access your data for marketing purposes. Individuals may be contacted through email by IBM marketing representatives. For this purpose of "marketing" emails, home address and phone number are not required and this information should not be available to the marketers. Your birthday may be a foreseeable reason for contact so partial information of an individual's date of birth may be released. The remaining personal information will be required by the marketers for proper identification. The ability for individuals to opt-out of these marketing emails is a corporate and legal feature, and not a privacy feature of PriSQL. For corporations to comply with email legislations, a company will create an internal 'do not contact' list of individuals whom have opted out. The privacy considerations for PriSQL are for those individuals which have not opted out. The privacy tuple is as follows:

Privacy := < “Marketing”, {2-Phone number Address, 1-Date of Birth, 0- Others}, {Marketing, 3rd Party Marketers}, Not Specified>

```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"First_Name","Select","Marketing","0","Marketing") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Last_Name","Select","Marketing","0","Marketing") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Address","Select","Marketing","2","Marketing") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Email","Select","Marketing","0","Marketing") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Date_of_Birth","Select","Marketing","1","Marketing") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Phone_Number","Select","Marketing","2","Marketing") EXPIRES 36-Month;
```

5.1.4 Recruitment

> **IBM Human Resources Web Sites (including Recruitment)**
> In connection with a job application or inquiry, whether advertised on an IBM Web site or otherwise, you may provide us with information about yourself, such as a resume or curriculum vitae. We may use this information throughout IBM and its related entities for the purpose of employment consideration or your inquiry. Except where you tell us not to, we will keep the information for future consideration.

When personal information is provided to IBM in connection with job applications, IBM recruiters may access you personal data for assessment. Under this scenario it is understandable that the recruiter will require the complete set of information without any need for generalization. The privacy tuple to support this privacy policy is as follows:

Privacy := <"Recruitment", 0, Recruiters, Not Specified}

```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"First_Name","Select","Recruiter","0","Employment") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Last_Name","Select","Recruiter","0","Employment") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Address","Select","Recruiter","0","Employment") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "Email","Select","
Recruiter ","0","Employment") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Date_of_Birth","Select","Recruiter","0","Employment") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Phone_Number","Select","Recruiter","0","Employment") EXPIRES 36-Month;
```

5.1.5 Data Analysis

> **Use of Suppliers**
> In some cases IBM uses suppliers to collect, use, analyze and otherwise process information on its behalf. It is IBM's practice to require such suppliers to handle information in a manner consistent with IBM's policies.

IBM's privacy policy indicates that 3rd party suppliers may perform data analysis on sensitive data within the database. IBM indicates that its suppliers must conform to privacy standards that IBM maintains. While this statement sounds good in principle, legally and practically this cannot be enforced. When a problem occurs the parent companies actually have very little control over the actions of its suppliers[3]. Taking this into account, personal information will not be released to the 3rd party supplier as microdata. Each attribute will be generalized to level 1 so that a certain level of privacy protection is enforced. While this is not a direct interpretation of the privacy policy, it does demonstrate PriSQL's ability to enhance the privacy requirements. The resulting privacy tuple is shown below:

Privacy := < “Data Analysis”, 1, 3rd Party Analyst, Not Specified>

```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "First_Name","Select","3rd-
Party","1","Data Analyses") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "Last_Name","Select","3rd-
Party","1","Data Analyses") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "Address","Select","3rd-
Party","1","Data Analyses") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "Email","Select","3rd-
Party","1","Data Analyses") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "Date_of_Birth","Select","3rd-
Party","1","Data Analyses") EXPIRES 36-Month;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info", "Phone_Number","Select","3rd-
Party","1","Data Analyses") EXPIRES 36-Month;
```

5.1.6 Legal Requirements

Disclosures Required By Law

Please be aware that in certain circumstances, it is possible that personal information may be subject to disclosure pursuant to judicial or other government subpoenas, warrants, or orders.

[3] “ Iphone maker in China is under fire after a suicide” [DBa09]. This article shows that while certain standards are maintained and expected of major corporations such as Apple Inc. and IBM inc, their suppliers located in other parts of the world are not held to the same standards.

IBM's last privacy policy is that in the event of a judicial or other government subpoenas, warrants or orders, an individual's personal data will be provided. Since this is not a reoccurring business requirement, the retention date is specified as infinite. In our society, a government's right to request information can occur at anytime so the privacy access of government official will never change or expire. The data tuple itself, may no longer be valid past the statute of limitation however the privacy permission will never need to be reviewed or renewed. The privacy tuple is shown below:

Privacy := < "Subpoenas", 0, Government, Infinite>

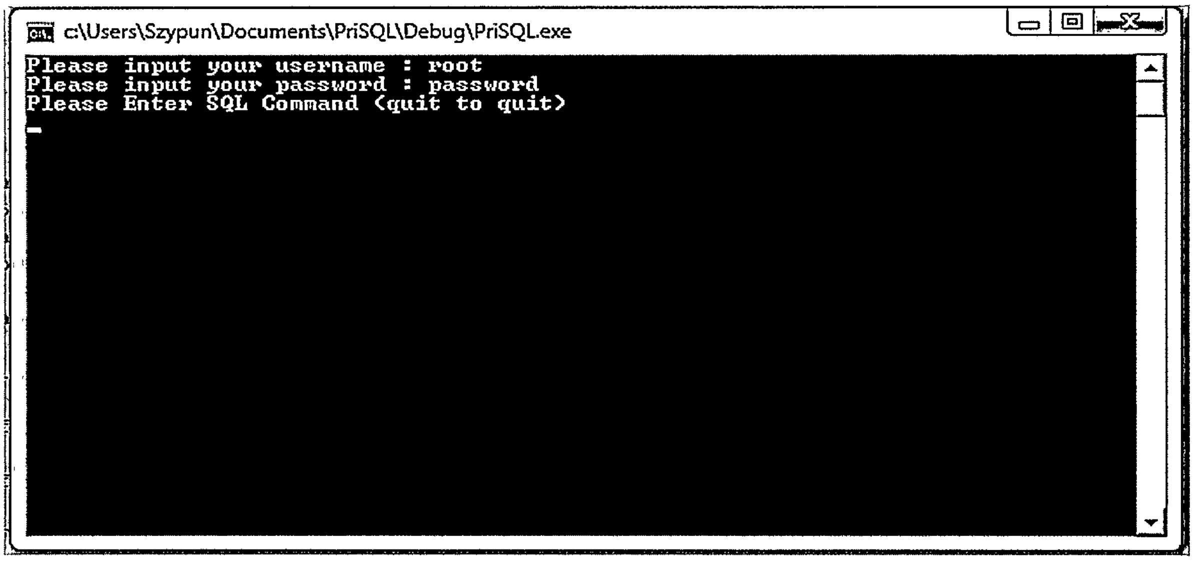
```
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"First_Name","Select","Government","0","Subpoenas") EXPIRES Infinité;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Last_Name","Select","Government","0","Subpoenas") EXPIRES Infinité;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Address","Select","Government","0","Subpoenas") EXPIRES Infinité;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Email","Select","Government","0","Subpoenas") EXPIRES Infinité;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Date_of_Birth","Select","Government","0","Subpoenas") EXPIRES Infinité;
INSERT INTO SysPrivacyPolicies VALUES("Personal_Info",
"Phone_Number","Select","Government","0","Subpoenas") EXPIRES Infinité;
```

5.2 Use Cases

The greatest consideration businesses have prior to upgrading technologies or procedures is its affect on current business requirements. Technologies that create impedance or cannot demonstrate 'value added' properties are rarely adopted. The greatest value of PriSQL is its ability to enforce privacy requirements; this was demonstrated in the previous chapters. The goal of this section is to demonstrate that PriSQL is a viable option for businesses and is capable of accomplishing business functions without much greater overhead on their employees. SQL is rarely used as the interface for end users. Applications that consider human computer interactions are developed to increase the ease of using a database. This type of application and research is not within the scope of this thesis. The following sections is a demonstration of how users can interact with PriSQL using a console based interface.

5.2.1 Logging In

Users are prompted for their username and password to determine their visibility attributes for this session. This name and password is verified to be valid by using the DBMS user authentication. Once the login process is complete PriSQL will permit PriSQL queries to be executed. The username is used as the visibility parameter for any privacy sensitive information access. A screen shot of the login procedure is shown in Figure 12.



```
Please input your username : root
Please input your password : password
Please Enter SQL Command (quit to quit)
_
```

Figure 12 – PriSQL Login Process

5.2.2 Data Access

To demonstrate how data retrieval will work in each use cases, the Personal_Info data table within PriSQL was populated with the information shown in Figure 13. To populate the table a series of insert statements were executed within PriSQL. The user logged in was a user with administrator privileges. In PriSQL any user with administrative authorities is permitted to insert data. Conceivably, the PriSQL data access mechanism can be altered to control which individuals are capable of inserting information into data tables. However, the insertion of information can never result in privacy violation so this feature was not implemented. Appendix A contains the current status of the privacy system catalogues.

User_ id	First_ Name	Last_ Name	Address	Email	DOB	Phone_ Number	Retention
1	Sampson	Pun	469 Hawkstone Dr. NW., Calgary	Spun @gmail.com	May 1 1984	403- 397-0389	18 Months
2	Joyce	Fu	3807 S St.SE., Washinton	Jfu368 @hotmail.com	Dec 24 1986	403- 891-8970	18 Months
3	Desmond	Chiu	93 Hawktree Close. N.W., Los Angeles	Dbschiu @hotmail.com	May 5 1984	403- 241-9137	18 Months
4	Jemma	Poon	327 Simcrest Hts. SW., Calary	Jemma_poon @hotmail.com	Nov 18 1996	403- 685-6168	18 Months
5	Eric	Lam	704 128 2nd Ave SE., Vancouver	Ericl @yahoo.ca	Jul 15 1990	403- 399-8421	18 Months

Figure 13 – Information stored within the Personal_info table

5.2.3 Marketing

This scenario involves a new marketing project aimed at individuals who are between the ages of 25-27 (1984-1986). This project will contact customers through emails and those who respond will be sent a gift from IBM to the address provided. For the marketing users, date of birth for individuals is generalized to the p-level of year and address is not visible. Therefore, if IBM wanted to give birthday presents to select individuals through marketing campaigns they would not be able to do so under the current privacy agreement. This scenario fully exemplifies how PriSQL can enhance the privacy of its customers by only displaying the results necessary to complete a task. For a marketing user to identify the customers applicable to the event, they will submit the following query:

```
(User: Marketing)
SELECT first_name, last_name, email, address
FROM personal_info
WHERE data_of_birth > = 1984 AND date_of_birth <= 1986
BECAUSE Marketing;
```

The following results are returned to the data requestor.

First_Name	Last_Name	Address	Email	DOB
Sampson	Pun	-	Spun@gmail.com	May 1 1984
Joyce	Fu	-	Jfu368@hotmail.com	Dec 24 1986
Desmond	Chiu	-	Dbschiu@hotmail.com	May 5 1984

Note how address was requested as part of the query but none of the actual addresses were displayed to the marketing user. This is because according to the privacy policy, marketing users do not have the permission to view this sensitive piece of data. Instead when a specific user responds to the email and wishes to receive the free gift, this request is submitted to a sales representative. A sales representative will then complete the transaction by verifying the address of the recipient. By using PriSQL, the marketer was not able to access the home address of each customer and the sales representative was not able to learn of the age of each customer.

```
(User: Sales)
SELECT first_name, last_name, address
FROM personal_info
WHERE first_name = “Desmond” and last_name = “Chiu”
BECAUSE U-Sales
```

The following results are returned the data requestor.

First_Name	Last_Name	Address
Desmond	Chiu	93 Hawktree Close. N.W., Los Angeles

5.2.4 3rd Party Data Analysis

The storage of massive amounts of information has led to the development of data mining techniques. Some corporations do not have the resources to complete this analysis internally so they hire 3rd party suppliers to complete the data mining. In this scenario information is going to be released to the 3rd party supplier for data analysis according to the privacy policies agreed upon with the data provider.

(User: 3rd Party Analyst)

SELECT *

FROM personal_info

BECAUSE Data Analysés

The following results are returned to the data requestor.

User_ Id	First_ Name	Last_ Name	Address	Email	DOB	Phone_ Number
1	-	-	Calgary	-	1984	403
2	-	-	Washinton	-	1986	403
3	-	-	Los Angeles	-	1984	403
4	-	-	Calgary	-	1996	403
5	-	-	Vancouver	-	1990	403

From a data analyst's point of view, this data may be very limited compared to the previous amount of information they had access too. However, this amount of information is consistent with the privacy policy and fully respects the privacy of the data providers which should be respected even over the wishes of the data analysts.

5.2.5 Government Subpoena

In this scenario a government agent has a subpoena requesting any information related to an individual with the first name "Sampson". The user will log in as *government* and provide their passwords as requested. Next, any queries completed should provide *subpoenas* as the access purpose of the query.

(User: Government)

SELECT *

FROM personal_info

WHERE first_name = "Sampson"

BECAUSE "Subpoenas"

The following results are returned to the data requestor.

User_ id	First_ Name	Last_ Name	Address	Email	DOB	Phone_ Number
1	Sampson	Pun	469 Hawkstone Dr. NW., Calgary	Spun @gmail.com	May 1 1984	403- 397-0389

Chapter Six: Conclusions and Future Work

This thesis presents a novel approach for protecting the privacy of data providers within a relational database. Chapter Two describes *privacy* in detail and describes what it means to preserve privacy. Chapter Three explains the PriSQL extensions themselves along with how DML and DDL commands work. Chapter Four features the implementation of PriSQL as a wrapper around a My/SQL database. In order to demonstrate the applicability and completeness of the solution PriSQL is applied to the IBM privacy policy in Chapter Five. This chapter summarizes how PriSQL addressed common technical and legislative privacy concerns as well as future works that will enhance PriSQL.

6.1 Hippocratic Database

The seminal work in privacy aware databases was brought forward by Argawal *et al.* through their paper Hippocratic Databases [ArS02]. Much of the work within PriSQL has been motivated by concepts discussed in this paper. For a database to claim it is a Hippocratic database it must address ten privacy principles. These ten principles highlight essential technical and legislative privacy concerns. These 10 privacy principles are: Purpose Specification, Consent, Limited Use, Limited Disclosure, Limited Retention, Compliance Accuracy, Safety, Openness, and Limited Collection.

6.1.1 Purpose Specification

Purpose specification is defined as follows: For personal information stored in the database, the purposes for which the information has been collected shall be associated with that information. One of PriSQL major concepts is in the storage of intended purpose. Each privacy policy shown in Chapter 5 demonstrates an intended purpose for the personal information that is collect. These policies are represented and stored in the SysPrivacyPolicies system table. Within each tuple of SysPrivacyPolicies is a table and column attribute (Section 3.1.1) allowing the intended purpose to be associated with personal information.

6.1.2 Consent

PriSQL's objective is to ensure that users of PriSQL cannot violate any of the published privacy policies agreed upon by the corporation and their customers. One assumption within PriSQL is that the data provider of personal information is from the individual themselves. Therefore, by providing personal information to the corporation the individual has implicitly consented to the use of their information in accordance to the privacy policy.

6.1.3 Limited Use

The usefulness of collecting data is ultimately to analyze the information. Unrestrained access to personal information will lead to privacy violations. Hippocratic databases shall run only those queries that are consistent with the purposes for which the information is collected. Much of PriSQL's concepts and motivation has been to ensure that this privacy principle is enforced. Current database systems rely on internal procedures and employee judgement. PriSQL has introduced a new SQL keyword "BECAUSE" (Section 3.3). The BECAUSE keyword is used to indicate the access purpose of a specific query. Verifying the access purpose and the intended purpose allows PriSQL to run only those queries which are consistent.

6.1.4 Limited Disclosure

Aggrawal's vision [ArS02] of limited disclosure was that personal information shall not be communicated outside of the database for purposes other than those for which there is consent from the donor of the information. PriSQL expands on this concept with the addition of a visibility parameter within our privacy specification (Section 2.5). Rather than only limiting disclosure of personal information on an internal and external basis; PriSQL limits disclosure between internal users as well. Many of the employees within an organization do not need to know the personal information inside the database. By allocating a visibility parameter, employees will not be able to access private information beyond the privacy policy consented upon.

6.1.5 Limited Retention

Limited retention is a noble concept. Personal information should be retained only as long as necessary for the fulfillment of the intended purposes. The low cost of storage relative to the high cost of collecting information deters many companies from actually implementing this concept. PriSQL makes this an explicit requirement for personal information. When personal information is inserted into a PriSQL database, the EXPIRES keyword is used to indicate the retention period of the data (Section 3.3.2). Once a piece of data has past the expiration date, the data will be removed automatically.

6.1.6 Compliance

A data provider should be able to verify compliance of the privacy policy and principles. Verifying compliance is a retroactive task, if a company is not fulfilling its obligations to the privacy principles than the data provider's privacy has already been violated. Privacy is a unique theoretical concept in that once it has been violated; it can never be fully restored. Rather than auditing compliance, PriSQL enforces compliance through technological means. By placing privacy as a first order principle within the DBMS, users of the system cannot violate the privacy of the data providers.

6.1.7 Others

The remaining Hippocratic principles were not directly address by PriSQL. These principles, safety, openness, accuracy, limited collection, apply to other areas of research separate from privacy. Safety of the data applies to the security domain more than the privacy domain. Openness and accuracy are business features rather than privacy features. Not being able to view data about oneself or correcting it does not necessarily violate the privacy of the individual. Limited collection is also a business requirement. While it is ideal that only the minimum amount of information is collected, the definition of 'minimum' is difficult to quantify. Therefore, we must rely on corporation to self govern this aspect of a Hippocratic database.

6.2 Closing Statements

The initial intent of this research was to study the state of information privacy within academia and corporate settings. However, because of a lack of a proper definition for the term privacy it was difficult to comprehend the areas of privacy that needed improvement. The privacy taxonomy was created to address that need. Using the privacy taxonomy, privacy can be defined by four parameters: purpose, generalization, visibility and retention. Under this notion, it was noticed that much of the privacy research failed to completely address all the dimensions. This left corporate users without the tools necessary to properly protect the privacy of data providers. The goal was to develop a method/tool that corporations could use in order to protect the privacy of their customers.

The result of this research is PriSQL. PriSQL is an extension of the structure query language that allows users to associate privacy preferences with sensitive information. New data access mechanisms are also brought in place so that privacy preferences are enforced. The implemented system has generally accomplished the goals set out in Chapter 1 (Section 1.1). Currently, the implementation has reached the point where more complex tools can be built to enhance this system. The mechanisms and structure has been developed within PriSQL so that the work of other privacy researchers can be utilized. This allows for a more robust privacy protection database management system.

Based on the syntactic and semantics requirements of privacy, four new SQL keywords were introduced to create PriSQL. These words are *because*, *expires*, *retention*, and *generalization* (Chapter 3*)*. Each of these keywords is used in association with select data definition language and data manipulation language terms. Using the four PriSQL keywords allow SQL based systems to interpret the privacy requirements of the corporation and data providers.

To demonstrate the usefulness and practicality of PriSQL, a portion of the IBM privacy policy was implemented within the PriSQL system (Chapter 5). This section shows how a privacy policy can be interpreted under the privacy taxonomy establish in Chapter 2. Once the privacy policies were properly interpreted and inserted into the system, a set of use cases are shown to demonstrate the enforcement of privacy. This enforcement of privacy and PriSQL in general, is also shown to enhance the vision of a Hippocratic database (Chapter 6).

6.3 Future Work

The PriSQL system was developed as a proof of concept application. The result of this is that some available features were not fully implemented. Chapter 2 described different privacy enforcement technologies ranging from generalization methodologies to purpose hierarchies. In order to develop PriSQL into a complete privacy aware database these other privacy generalization algorithms should be implemented as possible generalization functions. Furthermore, even though PriSQL's goal is to make privacy a first order principle, attention to efficiency must be given. Currently, purpose is stored as a series of unique strings; this is both ineffective and hard to manage when the number of purposes is expanded to a corporate level. In order to reduce this complexity, a purpose hierarchy could be developed such that the intended and access purposes are nodes on a hierarchy rather than individual strings.

The development of PriSQL has shown that there is an obvious need for more privacy tools to aid in the management of privacy. PriSQL by itself allows for SQL level manipulation of the database. Unfortunately, corporate users are not all knowledgeable in the SQL fundamentals. This makes it difficult to convince corporations to adopt this new technology. One possible solution is the development of a privacy visualization interface. Rather than relying entirely on text command and inputs, privacy could be visualized in a multi-dimensional space. Such a visualization tool could help everyone from data providers to database administrators in order to understand the privacy requirement being imposed upon them.

PriSQL is currently capable of preventing authorized users from unknowingly violating the corporate privacy agreements. PriSQL is still not entirely sufficient to protect against malicious internal users. One area of future work with great personal interest is purpose validation. PriSQL's current assumption is that the purpose specified is truthful. It is currently undetectable whether or not a user is lying about the access purpose. PriSQL limits the disclosure of information in this scenario because each user only has a handful of assigned data permissions. Current proposed solutions to this problem include implementing a workflow approach to data access. When a workflow is initiated a series of conditions and tasked must be fulfilled prior to the release of information. This solution shows promise in that completing a workflow can verify that a user is going to use the data retrieved for the purpose specified.

PriSQL was developed to enforce corporate privacy policies meaning that every data provider would have to conform to a single privacy policy. With the rapid adoption of social networking sites a need for individual privacy setting is easily envisioned. Due to the sheer number of users careful consideration is required to alter PriSQL. Otherwise a row explosion would occur within the privacy system table that results in a greater decrease in efficiency as well as unmanageable privacy policies. However I believe this work is critical as it will finally provide society with the means to define how their personal information is used and thus fully enforce the spirit of information privacy.

References

[ArS02] R. Agrawal, J. Kiernan, R. Srikant and Y. Xu. Hippocratic Databases. Proceedings of the 28th international conference on Very Large Databases. 2002, page 143-154.

[BaA05] R. Bayardo, R. Agrawal. Data Privacy through Optimal k-Anonymization. In Proceedings of the 21st International Conference on Data Engineering, 2005, page 217-228.

[BaM09] K. Barker, M. Askari, M. Banerjee, K. Ghazinour, B. Mackas, M. Majedi, S. Pun and A. Williams. A Data Privacy Taxonomy. British National Conference on Databases. July 2009, pages 42-54.

[Bar09] David Barboza. Iphone Maker in China Is Under Fire After a Suicide. New York Times. July 26 2009.

[BBC07]Unknown. Google cuts data retention times. BBC News Corp, June 12 2007.

[Ber06] B. Bergstein. Research explores data mining, privacy. USA Today, June 18 2006.

[ByB05] J. Byun, E. Bertino, N. Li. Purpose Based Access Control of Complex Data for Privacy Protection. Symposium on Access Control Models and Technologies, June 200, pages 102-110.

[ChP81] D. Chamberlin, T. Price, M. Astrahan, F. Putzolu et al. A History and Evaluation of System R. Communications of the ACM, Volume 24 Number 10 pp: 632 – 646. October 1981.

[CWC08] Unknown. Google cuts data retention policy from 18 to 9 months. Computer World Canada, Sept 11 2008.

[Dal86] T. Dalenius. Finding a needle in a haystack – or identifying anonymous census record. Journal of Official Statistics 2(3):329-336, 1986.

[DoL09] W. Douglas and D. Lightman. AIG Chief tells Congress: 'We had to give them Bonuses'. McClatchy Newspapers. March 18 2009.

[ElN03] R. Elmasri and S. Navathe. Fundamentals of Database Systems 4th Edition. Addison-Wesley Publishing. 2003.

[FeK92] D.F. Ferraiolo and D.R. Kuhn. Role Based Access Control, 15th National Computer Security Conference, Oct 13-16, 1992, page 554-563.

[FeK07] D.F. Ferraiolo, R. Kuhn, R. Sandhu. RBAC Standard Rationale: comments on a Critique of the ANSI Standard on Role Based Access Control. IEEE Security & Privacy, vol. 5, no. 6 (Nov/Dec 2007), page 51-53.

[FuW07] B. Fung, K. Wang, P. Yu. Anonymizing Classification Data for Privacy Preservation. IEEE Transactions on knowledge and Data Engineering, Volume 19 Issue 5, May 2007, page 711-725.

[GaJ05] S.L. Garfinkel, A. Juels, R. Pappu. RFID privacy: an overview of problems and proposed solutions. Security & Privacy, IEEE, Volume 3, Issue3, May-June 2005, page 34-43.

[HaK01] J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Francisco, USA, 2001, page 56-58.

[HPP03] Unknown. Medical Privacy Stories. Health Privacy Project, November 10, 2003.

[Iye02] V. Iyengar. Transforming Data to Satisfy Privacy Constraints. In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pages 279—288.

[Ric95] L. Rich. Right to Privacy in the Workplace in the Information Age. The Publishing Law Center. 1995.

[LiL07] N. Li, T. Li, S. Venkatasubramanian, t-closeness: Privacy Beyond k-Anonymity and l-diversity, IEEE 23rd International Conference on Data Engineering 2007, pages 106-115.

[MaK07] A .Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data. Volume 1, Issue 1. March 2007.

[Mar02] M. Marchiori. The platform for privacy preferences 1.0. (P3P1.0) specification W3C recommendation. W3C, Apr. 2002.

[MeW04] A. Meyerson, R. Williams. On the complexity of optimal k-anonymity, Symposium On Principles of Database Systems. June 2004, page 223-228.

[NiL07] Q. Ni, D. Lin, E. Bertino, and J. Lobo. Conditional privacy-aware role based access control. In European Symposium on Research in Computer Science. volume 4734 of Lecture Notes in Computer Science, Springer, 2007, pages 72–89.

[NiT07]Q. Ni, A. Trombetta, E. Bertino, and J. Lobo. Privacy-aware role based access control. In Symposium on Access control Models and Technologies, ACM, June 2007, pages 41–50.

[Ohm09] P. Ohm. Broken Promises of Privacy: Responding to the surprising failure of Anonymization. Social Science Research Network. Ver. 0.99. August 14 2009.

[RaC01] N. Ratha, J. Connell, R. Bolle. Enhancing security and privacy in biometrics-based authentication systems. IBM Systems Journal, Volume 40 Number 3, March 2001, pages 614-634

[SH00] 2000/520/EC: Commission Decision of 26 July 2000 pursuant to Directive 95/46/EC of the European Parliament and of the Council on the adequacy of the protection provided by the safe harbour privacy principles and related frequently asked questions issued by the US Department of Commerce. Official Journal L215, 25/08/2000 P. 0007-0047

[SUN09] MySQL 6.0 Reference Manual. Sun Microsystems Inc. July 23 2009.

[Swe02] L. Sweeney. K-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, Feb. 2002, 10(5): pages 557–570

[Swe05] L. Sweeney, Achieving K-Anonymity Privacy Protection Using Generalization and Suppression, International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, Feb. 2002, 10(5): pages 571–588

[WeS04] S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. Security in Pervasive Computing, LNCS, 2004, pages 201-212

[WiB08] P. Williams and K. Barker. Controlling Inference: Avoiding P-level Reduction During Analysis. Journal of research and Practice in Information Technology, 2008, 40(3): pages 183-204

[Win02] W. Winkler, Using Simulated Annealing for k-anonymity. Research Report Series Statistics #2002-07, November 19 2002

[WoL06] R. Wong, J. Li, A. Fu, and K. Wang. (α, k)Anonymity: An Enhanced k-Anonymity Model for Privacy Preserving Data Publishing, International Conference on Knowledge Discovery and Data Mining, August 2006, pages 754-759

[XiT06] X. Xiao and Y. Tao. Personalized Privacy Preservation. ACM's Special Interest Group on Management of Data, June 2006, pages 229-240

[XiT07] X. Xiao, Y. Tao, m-Invariance: Towards Privacy Preserving Re-publication of Dynamic Datasets In ACM Symposium International Conference On Management of Data, June 2007, pages 689-700

Appendix A – Contents of the Privacy Catalogues after Fulfillment of Privacy Policies

SysTransformFunctions

Table	Column	GenFunction
Personal_Info	First_Name	Remove
Personal_Info	Last_Name	Remove
Personal_Info	Address	Anon_Address
Personal_Info	Email	Remove
Personal_Info	Date_of_Birth	Anon_DOB
Personal_Info	Phone_Number	Anon_Phone

SysPrivacyPolicies

	Table	Column	Operation(Op)	Visibility	Generalization	Purpose	Retention
1	Personal_Info	-	Insert	-	-	Retention	18-Month
2	Personal_Info	First_Name	Update	Rep	0	Supplement	36-Month
3	Personal_Info	Last_Name	Update	Rep	0	Supplement	36-Month
4	Personal_Info	Address	Update	Rep	0	Supplement	36-Month
5	Personal_Info	Email	Update	Rep	0	Supplement	36-Month
6	Personal_Info	Date_of_Birth	Update	Rep	0	Supplement	36-Month
7	Personal_Info	Phone_Number	Update	Rep	0	Supplement	36-Month
8	Personal_Info	First_Name	Select	Rep	0	Supplement	36-Month
9	Personal_Info	Last_Name	Select	Rep	0	Supplement	36-Month
10	Personal_Info	Address	Select	Rep	0	Supplement	36-Month
11	Personal_Info	Email	Select	Rep	0	Supplement	36-Month
12	Personal_Info	Date_of_Birth	Select	Rep	0	Supplement	36-Month
13	Personal_Info	Phone_Number	Select	Rep	0	Supplement	36-Month
14	Personal_Info	First_Name	Select	Sales	0	U-Sales	36-Month
15	Personal_Info	Last_Name	Select	Sales	0	U-Sales	36-Month
10	Personal_Info	Address	Select	Sales	0	U-Sales	36-Month
11	Personal_Info	Email	Select	Sales	0	U-Sales	36-Month

12	Personal_Info	Date_of_Birth	Select	Sales	2	U-Sales	36-Month
13	Personal_Info	Phone_Number	Select	Sales	0	U-Sales	36-Month
14	Personal_Info	First_Name	Select	Marketing	0	Marketing	36-Month
15	Personal_Info	Last_Name	Select	Marketing	0	Marketing	36-Month
16	Personal_Info	Address	Select	Marketing	0	Marketing	36-Month
17	Personal_Info	Email	Select	Marketing	0	Marketing	36-Month
18	Personal_Info	Date_of_Birth	Select	Marketing	1	Marketing	36-Month
19	Personal_Info	Phone_Number	Select	Marketing	2	Marketing	36-Month
14	Personal_Info	First_Name	Select	Recruiter	0	Employment	36-Month
15	Personal_Info	Last_Name	Select	Recruiter	0	Employment	36-Month
16	Personal_Info	Address	Select	Recruiter	0	Employment	36-Month
17	Personal_Info	Email	Select	Recruiter	0	Employment	36-Month
18	Personal_Info	Date_of_Birth	Select	Recruiter	0	Employment	36-Month
19	Personal_Info	Phone_Number	Select	Recruiter	0	Employment	36-Month
20	Personal_Info	First_Name	Select	3rd-Party	1	Data Analyses	36-Month
21	Personal_Info	Last_Name	Select	3rd-Party	1	Data Analyses	36-Month
22	Personal_Info	Address	Select	3rd-Party	1	Data Analyses	36-Month
23	Personal_Info	Email	Select	3rd-Party	1	Data Analyses	36-Month
24	Personal_Info	Date_of_Birth	Select	3rd-Party	1	Data Analyses	36-Month
25	Personal_Info	Phone_Number	Select	3rd-Party	1	Data Analyses	36-Month
26	Personal_Info	First_Name	Select	Government	0	Subpoenas	Infinité
27	Personal_Info	Last_Name	Select	Government	0	Subpoenas	Infinité
28	Personal_Info	Address	Select	Government	0	Subpoenas	Infinité
29	Personal_Info	Email	Select	Government	0	Subpoenas	Infinité
30	Personal_Info	Date_of_Birth	Select	Government	0	Subpoenas	Infinité
31	Personal_Info	Phone_Number	Select	Government	0	Subpoenas	Infinité

Appendix B – Algorithms for Generalization Transform Functions

Function Name: Remove

Input: P-Level, Attribute Data

Output: Privacy Preserving Attribute Data

Algorithm:

```
Remove(p-level, attribute_data)
Begin:
    If p-level == 0:
        Return attribute_data
    Else
        Return Null
End
```

Function Name: Anon_Address

Purpose: Converts an address to its corresponding privacy level generalization

Input: P-Level, Attribute Data (Address)

Output: Privacy Preserving Attribute Data

Algorithm:

```
Anon_Address(p-level, attribute_data)
Begin:
    If p-level == 0;
        Return attribute_data
    Else if p-level == 1;
```

```
                For elements in attribute_data:
                    Elements := Elements.city
                Return Attribute_data
            Else
                Return null
        End
```

Function Name: Anon_DOB

Purpose: Converts a date of birth to its corresponding privacy level generalization

Input: P-Level, Attribute Data (Date of Birth)

Output: Privacy Preserving Attribute Data

Algorithm:

```
    Anon_DOB(p-level, attribute_data)
    Begin:
        If p-level == 0;
            Return attribute_data
        Else if p-level == 1;
            For elements in attribute_data:
                Elements := Elements.year
            Return Attribute_data
        Else
            Return null
    End
```

Function Name: Anon_Phone

Purpose: Converts a phone number to its corresponding privacy level generalization

Input: P-Level, Attribute Data (Phone Number)

Output: Privacy Preserving Attribute Data

Algorithm:

```
Anon_Phone(p-level, attribute_data)
Begin:
    If p-level == 0:
        Return attribute_data
    Else if p-level == 1:
        For elements in attribute_data:
            Elements := Elements.area_code
        Return Attribute_data
    Else :
        Return null
End
```