

ANGUILT TECHNOLOGY TO PREVENT DATA LEAKAGE AND ITS DETECTION ON CLOUD

¹S. Manojkumar, ²S. Arthi, ³R. Divya, ⁴K. Gayathri, ⁵K.B. Suganya¹Professor, ^{2,3,4,5}Student, Department of Information Technology,
Karpagam College of Engineering, Coimbatore.¹callsmk@gmail.com

Abstract: In the data mining project the analyzing purpose can be done by preparing the dataset. There are many other existing aggregations which have many kind of limitation and in result it results only one column per aggregated group but in preparing a data set for analysis is generally the most time consuming task in a data mining project because it requires many complex SQL queries where by joining many tables and also by aggregating columns. And so where we propose a new class of functions and which is called as horizontal aggregations. Horizontal aggregations is defined as returning a group of numbers instead of returning only one number per row. Horizontal aggregations are used to build the datasets with a horizontal denormalized layout and it is evaluated by using the main three methods which are the 1.CASE 2.SPJ3.PIVOT.

Keywords: SPJ, horizontal aggregations, data leakage, data privacy.

1. Introduction

Data leakage is the difficult challenge in industries. For security, most of the systems are designed by using different encrypt algorithms. It is very hard to determine that which agent leaks the data and it creates ethical issues in office environment. The data may unknowingly or maliciously leaked by the agents so there is no need to hand over the data to agents. Assume that we have the situation that we have to bring sensitive data to the agents and so we can trace its origins of the each and every object absolutely. Not all the agents are 100% trusted. We can easily identifying the leaker by generating the algorithm to implement the different data distribution strategies.

We have to use the perturbation technique. It is known as we can made the data as less sensitive by modifying it before handed to the agents. Distributor is known as owner of the data. Main goal is to detect the agent who leaks the distributor's sensitive data. In existing system, watermarking is used for detection of the data leakage. In watermarking unique code is embedded in data. Any unknown person can get this data and there is a chance to destroy the watermarks. So

anyone can modify the original data. In proposed system, we have to detect the agent who leaks the data and what data has been leaked by the agent using encrypted fake objects. Encryption is defines as process of encode messages and it cannot be read by any unknown person but authorisable persons can read it. Single key is used for encryption. By this key the sender encrypt the data into unreadable form and the receiver can decrypt the data by using a private key. The unauthorized person does not know the private key.

In relational database, effort is required to prepare the data set, it can be used as the input for data mining. Algorithms contain required number of input and a horizontal layout column. Research discipline uses different terminology to describe as input for data set. Statistics generally uses observation. Machine learning research uses instance feature. This article introduce a new class of aggregate functions that can be used to build data sets in a horizontally out denormalized with aggregations, automating SQL query writing and extending SQL query. This task requires writing long SQL statements or customizing SQL code generated by some tool. There exist many aggregation functions and operators in SQL. These aggregations have limitations to build data sets for data mining come from On-Line Transaction Processing (OLTP) systems in which databases are highly normalized. Data mining, statistical or machine learning algorithms are generally require to aggregated data in the summary form. Such effort is due to the amount and complexity of SQL code needs to be written, optimized. The further practical reasons are to return aggregation results in a horizontal layout. To perform analysis of exported tables into spread sheets it may be more convenient to have aggregations on the same group in one row. OLAP tools generate SQL code to transpose results.

The SQL code need to be written, optimized, tested by every time. There are many practical reasons to return aggregation results in horizontal layouts. Standard aggregations are difficult to interpret when grouping attributes have more cardinalities. To perform analysis of exported tables into spread sheets it may be more convenient to

have aggregations on the same group in a row. OLAP tools can generate SQL code to transpose the results sometimes called as PIVOT.

Transposition are very efficient, if mechanism of combine the aggregation and transposition. We proposes a new class of aggregate function that aggregates the numeric expressions and transposes the result to produce a horizontal layouts. Functions belonging to this class is known as horizontal aggregation. Horizontal aggregation represents traditional SQL aggregations, returns the set of values in a horizontally out instead of single value per row.

2. Related Work

The data leakage depends on the source from which the data is taken and the process of extracting data from it, which are given as the provenance of the data [1]. It determines the quality and amount of trust one places on the results [2]. We consider applications where the original sensitive data cannot be perturbed. The idea of perturbing data to detect leakage is not new. In most cases, separate objects are perturbed, i.e., by adding the random noise to sensitive salaries, or by adding the watermark to an image. In this case, perturbing the set of distributor objects by adding fake elements is done. In some applications, fake objects may cause fewer problems that perturbing real objects. For example, say the distributed data objects are medical reports and the agents are in the hospitals.

In this case, even small modifications to the records of actual patients may be undesirable. Perturbation is a very useful technique where the data are modified. The data can be made "less sensitive" before being handed to agents [8]. One can add random noise to certain attributes, or one can replace exact values.

There exist many proposals that have extended SQL syntax. The closest data mining problem associated to OLAP processing is association rule mining [18]. SQL extensions to define aggregate functions for association rule mining are introduced in [19]. In this case, the goal is to efficiently compute itemset support. Unfortunately, there is no notion of transposing results since transactions are given in a vertical layout. Programming a clustering algorithm with SQL queries is explored in [14], which shows a horizontal layout of the data set enables easier and simpler SQL queries. Alternative SQL extensions to perform spreadsheet-like operations were introduced in. Their optimizations have the purpose of avoiding joins to express cell formulas, but are not optimized to perform partial transposition for each group of result rows. The PIVOT and CASE methods avoid joins as well. Our SPJ method proved horizontal aggregations can be evaluated with relational algebra, exploiting outer joins, showing our work is connected to traditional query optimization [20]. The problem of optimizing queries with outer joins is not new. Optimizing joins by reordering operations and

using transformation rules is studied in. This work does not consider optimizing a complex query that contains several outer joins on primary keys only, which is fundamental to prepare data sets for data mining. Traditional query optimizers use a tree-based execution plan, but there is work that advocates the use of hypergraphs to provide a more comprehensive to potential plans[12]. This approach is related to our SPJ method. Even though the CASE construction SQL feature commonly used in-practice optimizing queries that have a list of similar CASE statements has not been studied in depth before.

The guilt detection [7] approach presented in paper is related to the data provenance problem tracing the line age of S objects implies essentially the detection of the guilty agents. Suggested solutions are domain specific, such as lineage tracing for data warehouses and assume some prior knowledge on the way a data view is created out of the data sources. Leakage problem formulation[7] with some objects and sets are more general and simplifying the lineage tracing, since we do not consider any data transformation from Ri sets to S. As far as the data allocation strategies are concerned, main work is mostly relevant to water marking that is used as a means of establishing original ownership of distributed objects. Watermarks were initially used in images, video and audio data whose digital representation includes considerable redundancy.

Recently, and other works have also studied marks insertion to relational data. This approach and water marking are similar in the sense of providing agents with some kind of receiver-identifying information. However, by its very nature, a watermark modifies the item being watermarked. If the object to be watermarked cannot be modified then a water mark cannot be inserted. In such the cases methods that attach watermarks to the distributed data are not applicable. Finally, there are also lots of other works on mechanisms that allow only authorized users to access sensitive data through access control policies. Such approaches prevent in some sense data leakage by sharing information only with trusted parties.

The owner of the data is called as distributor and the supposedly trusted third parties the agents. The goal is to detect[7] when the distributor's sensitive data had leaked by agents, and if possible by identifying the agent that leaked[6] the data. In this paper, a model is developed for accessing the "guilt" of agents. Paper presents algorithms for distributing objects to agents, in away that improves chances of identifying a leaker.

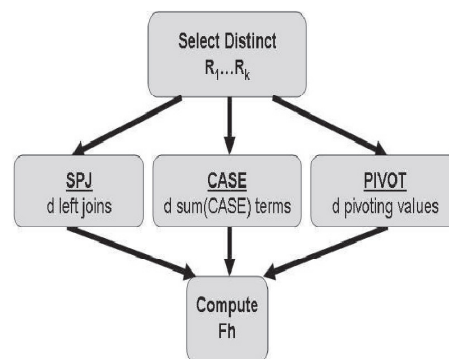
Traditionally, leakage detection is handled by watermarking, e.g., a unique code is embedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, the leaker can be identified. Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermarks can

sometimes be destroyed if the data recipient is malicious[1].E.g. A hospital may give patient records to researchers who will devise new treatments. Similarly, a company may have partnerships with other companies that require sharing customer data. Another enterprise may outsource its data processing, so data must be given to various companies. We call the owner of the data the distributor and the supposedly trusted third parties the agents.

Preparing a dataset for analysis is generally the most time consuming task in a data mining project, requiring many complex SQL queries, joining tables and aggregating columns. Existing SQL aggregations have limitations to prepare datasets because they return one column per aggregated group. In general, a significant manual effort is required to build data sets, where a horizontal layout is required. We propose simple, yet powerful, methods to generate SQL code to return aggregated columns in a horizontal tabular layout, returning a set of numbers instead of one number per row. This new class of functions is called horizontal aggregations. Horizontal aggregations build data sets with a horizontal denormalised layout (e.g. point- dimension, observation-variable, instance-feature), which is the standard layout required by most datamining algorithms. We propose three fundamental methods to evaluate horizontal aggregations: CASE: Exploiting the programming CASE construct; SPJ: Based on standard relational algebra operators (SPJ queries); PIVOT: Using the PIVOT operator, which is offered by some DBMSs. Experiments with large tables compare the proposed query evaluation methods. Our CASE method has similar speed to the PIVOT operator and it is much faster than the SPJ method. In general, the CASE and PIVOT methods exhibit linear scalability, whereas the SPJ method does not. Though there are number of systems designed for the data security by using different encryption algorithms, there is a big issue of the integrity of the users of those systems. It is very hard for any system administrator to trace out the data leaker among the system users. It creates a lot many ethical issues in the working environment of the office. The data leakage detection industry is very heterogeneous as it evolved out of ripe product lines of leading IT security vendors. A broad arsenal of enabling technologies such as firewalls, encryption, access control, identity management, machine learning content/context based detectors and others have already been incorporated to offer protection against various facets of the data leakage threat.

The competitive benefits of developing a "one-stop-shop", silver bullet data leakage detection suite is mainly in facilitating effective orchestration of the a fore mentioned enabling technologies to provide the highest degree of protection by ensuring an optimal fit of specific data leakage detection technologies with the "threat landscape" they operate in. This landscape is

characterized by types of leakage channels, data states, users, and IT platforms. An existing to preparing a data set for analysis is generally the most time consuming task in a data mining project, requiring many complex SQL queries, joining tables and aggregating columns. Existing SQL aggregations have limitations to prepare data sets because they return one column per aggregated group where it produces the disadvantage of Existing SQL aggregations have limitations to prepare data sets.



To return one column per aggregated group.

A. Data Leakage

In the course of doing business, sometimes sensitive data must be handed over to supposedly trusted third parties. A company may have partnerships with other companies that require sharing customer data. Another enterprise may outsource its data processing, so data must be given to various other companies. The distributor gives data to trusted third party over network. Some of the data is leaked and found in an unauthorized place (e.g., on the web or somebody's laptop). The distributor must assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means.

B. Data Leakage Detection

Data Leakage Detection proposes data allocation strategies (across the agents) that improve the probability of identifying leakages. These methods do not rely on alterations of the released data e.g. watermarks. In some cases distributor can also inject "realistic but fake" data records to further improve chances of detecting leakage and identifying the guilty party[1]. Distributor develops a model for assessing the "guilt" of agents. Project presents algorithms for distributing objects to agents, in a way that improves chances of identifying a leaker. Finally, Distributor also considers the option of adding "fake" objects to the

distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects acts as a type of watermark for the entire set, without modifying any individual members. If it turns out an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.

C. Data Leakage Problem

Goal of this project is to detect when the distributor's sensitive data has been leaked by agents, and if possible to identify the agent that leaked the data. Consider applications where the original sensitive data cannot be anxious. Perturbation is important useful technique where the data are modified and made "low sensitive" before being handed to agents. However, in some cases it is important not to alter the original distributor's data. For example, if an outsourcer is doing our payroll, he must have the exact salary and customer bank account numbers.

The distributor must assess the likelihood that the leaked data came from one or more other agents, as opposed by having independently collected by other means. Propose data allocation strategies that improve the probability of identifying leakages. These methods do not rely on alterations of the released data (e.g., watermarks). In some cases, also injecting the realistic but where the fake where the data records to further needed to improve our chances of detecting leakage or by identifying the guilty parties[8]. Goal is to detect when the distributor's sensitive data has been leaked by agents, and if possible to identify the agent that leaked the data. And this involves by investigating of the existing system, which is on time by consuming with the user and also it's insufficient depth. This includes the collection of data and study of detailed information and literature regarding the complete existing procedure.

The detail initial study documented and the failing and problem is noted separately. The system is properly designed and proper outline of the proposed computerized system is prepared. The proposed design is brought against all the known facts and further proposal are made. Various resources including the software, hardware and manpower requirements are decided and are mentioned. Our goal is to detect when the distributor's sensitive data has been leaked by agents, and if possible to identify the agent that leaked the data. Perturbation is a very useful technique where the data is modified and made "less sensitive" before being handed to agents. We develop unobtrusive techniques for detecting leakage of a set of objects or records. In this section we develop a model for assessing the "guilt" of agents. We also present algorithms for distributing objects to agents, in a way that improves our chances of identifying a leaker. Finally, we also consider the option of adding "fake"

objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects acts as a type of watermark for the entire set, without modifying any individual members. If it turns out an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.

3. Proposed System

We propose a new class of aggregate functions that aggregate numeric expressions and transpose results to produce a dataset with a horizontal layout. Functions belonging to this class are called horizontal aggregations. Horizontal aggregations represent an extended form of traditional SQL aggregations, which return a set of values in a horizontal layout (some what similar to a multidimensional vector), instead of a single value per row. This paper explains how to evaluate and optimize horizontal aggregations generating the standard SQL code.

Our proposed horizontal aggregations provide several unique features and advantages.

1. They represent a template to generate SQL code from a datamining tool. Such SQL code automates writing SQL queries, Optimizing them, and testing them for correctness. This SQL code reduces manual work in the data preparation phase in a data mining project.

2. Since SQL code is automatically generated it is likely to be more efficient than SQL code written by an end user. For instance, a person who does not know SQL well or someone who is not familiar with the database schema (e.g., a data mining practitioner). Therefore, data sets can be created in less time.

3. The dataset can be created entirely inside the DBMS. In modern database environments, it is common to export denormalized datasets to be further cleaned and transformed outside a DBMS in external tools (e.g., statistical packages). Unfortunately, exporting large tables outside a DBMS is slow, creates in consistent copies of the same data and compromises database security.

Advantage of guilt technique is the SQL code reduces manual work in the data preparation phase in a data mining project. The SQL code is automatically generated it is likely to be more efficient than SQL code written by an end user. The data sets can be created in less time. The data set can be created entirely inside the DBMS.

Module Description:

1. Admin Module
2. User Module
3. View Module
4. Download Module

Module 1 : Admin Module

Admin will upload new connection form based on regulations in various states. Admin will be able upload various details regarding user bills like a new connection to a new user, amount paid or payable by user. In case of payment various details regarding payment will be entered and separate username and password will be provided to users in large.

Module 2 : User Module

User will be able to view his bill details on any date may be after a month or after months or years and also he can view the our bill details in a various ways for instance, The year wise bills, Month wise bills, totally paid to bill in EB. This will reduce the cost of the transaction. If user thinks that his password is insecure, he has option to change it. He also can view the registration details and allowed to change or edit and save it.

Module 3 : View Module

Admin has three ways to view the user bill details, the 3 ways are

1. SPJ
2. PIVOT
3. CASE

SPJ : While using SPJ the viewing and processing time of user bills is reduced.

PIVOT : This is used to draw the user details in a customized table. This table will elaborate us on the various bill details regarding the user on monthly basis.

CASE : using CASE query we can customize the present table and column based on the conditions. This will help us to reduce enormous amount of space used by various user bill details. It can be viewed in two difference ways namely Horizontal and Vertical.

In case of vertical the number of rows will be reduced to such an extent it is needed and column will remain the same on other hand the Horizontal will reduce rows as same as vertical and will also increase the columnar format

Module 4 : Download Module

User will be able to download the various details regarding bills. If he/she is a new user, he/she can download the new connection form, subscription details etc. then he/she can download his /her previous bill details in hands so as to ensure it.

New Module Description for Views:

SPJ Method:

The SPJ method is interesting from a theoretical point of view because it is ased on relational operators only. The basic idea is to create one table with a vertical aggregation for each result column, and then join all those tables to produce F_H .

We aggregate from F into projected tables with d

Select-Project-Join-Aggregation queries (selection, projection, join, aggregation). Each table F_i corresponds to one subgrouping combination and has $fL_1; \dots; L_j$ g as primary key and an aggregation on A as the only nonkey column. It is necessary to introduce an additional table F_0 , which will be outer, joined with projected tables to get a complete result set. We propose two basic substrategies to compute F_H . The first one directly aggregates from F . The second one computes the equivalent vertical aggregation in a temporary table F_V grouping by $L_1; \dots; L_j; R_1; \dots; R_k$.

Then horizontal aggregations can be instead computed from F_V , which is a compressed version of F , since standard aggregations are distributive.

We now introduce the indirect aggregation based on the intermediate table F_V that will be used for both the SPJ and the CASE method. Let F_V be a table containing the vertical aggregation, based on $L_1; \dots; L_j; R_1; \dots; R_k$. Let $V()$ represent the corresponding vertical aggregation for H δ P. The statement to compute F_V gets a cube:

```
INSERT INTO FV
SELECT L1; ... ; Lj; R1; ... ; Rk, V(A)
FROM F
GROUP BY L1; ... ; Lj; R1; ... ; Rk;
```

Table F_0 defines the number of result rows, and builds the primary key. F_0 is populated so that it contains every existing combination of $L_1; \dots; L_j$. Table F_0 has $fL_1; \dots; L_j$ g as primary key and it does not have any nonkey column.

```
INSERT INTO F0
SELECT DISTINCT L1; ... ; Lj
FROM fF jFVg;
```

In the following discussion I 2 fl; ... ; dg: we use **hto** make writing clear, mainly to define Boolean expressions. We need to get all distinct combinations of subgrouping columns $R_1; \dots; R_k$, to create the name of dimension columns, to get d , the number of dimensions, and to generate the boolean expressions for WHERE clauses. Each WHERE clause consists of a conjunction of k equalities based on $R_1; \dots; R_k$.

```
SELECT DISTINCT R1; ... ; Rk
FROM fF jFVg;
```

Tables $F_1; \dots; F_d$ contain individual aggregations for each combination of $R_1; \dots; R_k$. The primary key of table F_i is $fL_1; \dots; L_j$ g.

```
INSERT INTO Fi
SELECT L1; ... ; Lj; V (A)
FROM fF jFVg
WHERE R11/4 v1I AND .. AND Rk1/4 vkI
GROUP BY L1; ... ; Lj;
```

Then each table F_i aggregates only those rows that correspond to the i th unique combination of $R_1; \dots$

$\dots; R_k$, given by the WHERE clause. A possible optimization is synchronizing table scans to compute the d tables in one pass. Finally, to get F_H we need d left outer joins with the $d-1$ tables so that all individual aggregations are properly assembled as a set of d dimensions for each group. Outer joins set result columns to null for missing combinations for the given group. In general, nulls should be the default value for groups with missing combinations. We believe it would be incorrect to set the result to zero or some other number by default if there are no qualifying rows. Such approach should be considered on a per-case basis.

INSERT INTO F_H

SELECT

$F_0:L_1; F_0:L_2; \dots; F_0:L_j;$

$F_1:A; F_2:A; \dots; F_d:A$

FROM F_0

LEFT OUTER JOIN F_1

ON $F_0:L_1 \text{ } \frac{1}{4} \text{ } F_1:L_1 \text{ and } \dots \text{ and } F_0:L_j \text{ } \frac{1}{4} \text{ } F_1:L_j$

LEFT OUTER JOIN F_2

ON $F_0:L_1 \text{ } \frac{1}{4} \text{ } F_2:L_1 \text{ and } \dots \text{ and } F_0:L_j \text{ } \frac{1}{4} \text{ } F_2:L_j$

$\dots$

LEFT OUTER JOIN F_d

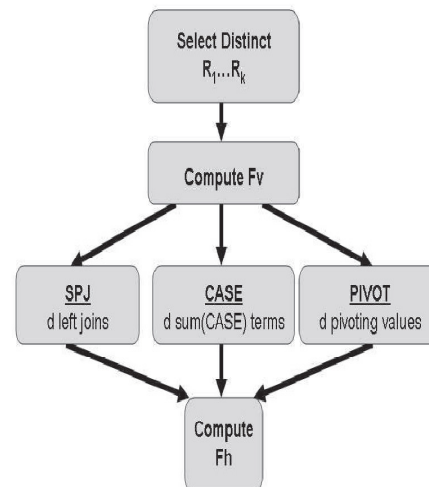
ON $F_0:L_1 \text{ } \frac{1}{4} \text{ } F_d:L_1 \text{ and } \dots \text{ and } F_0:L_j \text{ } \frac{1}{4} \text{ } F_d:L_j;$

This statement may look complex, but it is easy to see that each left outer join is based on the same columns $L_1; \dots; L_j$. To avoid ambiguity in column references, $L_1; \dots; L_j$ are qualified with F_0 . Result column I is qualified with table F_I . Since F_0 has n rows each left outer join produces a partial table with n rows and one additional column.

Then at the end, F_H will have n rows and d aggregation columns. The statement above is equivalent to an update-based strategy.

Table F_H can be initialized inserting n rows with key $L_1; \dots; L_j$ and nulls on the d dimension aggregation columns.

Then F_H is iteratively updated from F_I joining on $L_1; \dots; L_j$. This strategy basically incurs twice I/O doing updates instead of insertion. Reordering the d projected tables to join cannot accelerate processing because each partial table has n rows. Another claim is that it is not possible to correctly compute horizontal aggregations without using outer joins. In other words, natural joins would produce an incomplete result set.



We provide a more efficient, better integrated and more secure solution compared to external datamining tools. Horizontal aggregations just require a small syntax extension to aggregate functions called in a SELECT statement. Alternatively, horizontal aggregations can be used to generate SQL code from a data mining tool to build datasets for data mining analysis.

We introduce a new class of aggregations that have similar behavior to SQL standard aggregations, but which produce tables with a horizontal layout. In contrast, we call standard SQL aggregations vertical aggregations since they produce tables with a vertical layout. Horizontal aggregations just require a small syntax extension to aggregate functions called in a SELECT statement. Alternatively, horizontal aggregations can be used to generate SQL code from a data mining tool to build datasets for data mining analysis. We start by explaining how to automatically generate SQL code. We introduced a new class of extended aggregate functions, called horizontal aggregations with help preparing data sets for datamining and OLAP cube exploration. Specifically, horizontal aggregations are useful to create data sets with a horizontal layout, as commonly required by datamining algorithms and OLAP cross-tabulation. Basically, a horizontal aggregation returns a set of numbers instead of a single number for each group, resembling a multi-dimensional vector.

4. Conclusion

Thus we have been introduced a new class of extended aggregate functions which is called as the horizontal aggregations which are help by preparing the datasets for OLAP cube exploration and data mining. Specifically, horizontal aggregations are useful to create data sets with a horizontal layout, as commonly required by datamining algorithms and OLAP cross-

tabulation Existing SQL aggregations have limitations to prepare data sets because they return one column per aggregated group, extension to SQL standard aggregate functions to compute horizontal aggregations which just required specifying the subgrouping columns inside the aggregation function call. From a query optimization perspective, we proposed three query evaluation methods. The first one is SPJ which relies the standard relational operators and the second is CASE which relies the SQL CASE construct and where the third one is PIVOT which uses a built-in operator in a commercial DBMS but where that is not widely available. The first method where SPJ method is important from a theoretical point of view where it is based on select, project and join(SPJ) queries. The CASE method is also the most important contribution in the cloud where it is efficient evaluation method and it has been used widely since it can be programmed combining GROUP- BY and CASE statements.

5.Further Scope of the Project

Every application has its own merits and demerits. This work has covered almost all the requirements. Further requirements and improvements can easily be done since the coding is mainly structured or modular in nature. Changing the existing modules or adding new modules can append improvements. Further enhancements can be made to the application, so that the system will be immediately blocked while attacks take place. In future all transaction will be processed in a secure manner and can find the intruders activity by getting all relevant details.

References

- [1] P. Buneman, S. Khanna, and W.C.Tan, "Why and Where: A Characterization of Data Provenance," Proc. Eighth Int'l Conf. Database Theory (ICDT '01), J.V. den Bussche and V. Vianu, eds., pp. 316-330, Jan. 2001.
- [2] P. Buneman and W.-C. Tan, "Provenance in Databases," Proc. ACM SIGMOD, pp. 1171-1173, 2007.
- [3] S. Czerwinski, R. Fromm, and T. Hodes, "Digital Music Distribution and Audio Watermarking," 2007.
- [4] S. Jajodia, P. Samarati, M.L. Sapino, and V.S. Subrahmanian, "Flexible Support for Multiple Access Control Policies," ACM Trans. Database Systems, vol. 26, no. 2, pp. 214-260, 2001
- [5] P. Papadimitriou and H. Garcia-Molina, "Data Leakage Detection," IEEE Trans. on Knowledge And Data Engineering, Vol. 23, No. 1, Jan. 2011.
- [6] J.J.K.O. Ruanaidh, W.J. Dowling, and F.M. Boland, "Watermarking Digital Images for Copyright Protection," IEE Proc. Vision, Signal and Image Processing, vol. 143, no. 4, pp. 250-256, 1996..
- [7] IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL., NO. 2010 Data Leakage Detection Panagiotis Papadimitriou, Member, IEEE, Hector Garcia-Molina, Member, IEEE
- [8] J.Gray,A. Bosworth, A.Layman, and H. Pirahesh. Datacube:A relational aggregation operator generalizing group- by, cross-tab and subtotal. InICDE Conference.
- [9] J.Hanand M.Kamber.Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco, 1st edition,2001.
- [10] Sandip A.Kale Prof.Kulkarni S.V. Dr.B.A.M.University, Aurangabad(M.S), India, Data Leakage Detection: A Survey, (IOSR Journal of Computer Engineering (IOSRJCE)ISSN :2278-0661 Volume 1, Issue6(July-Aug 2012),PP32-35 www.iosrjournals.org
- [11] IEEE Transactions On Knowledge And Data Engineering, Vol. 22, No. 3, March2011DataLeakage Detection Panagiotis Papadimitriou, Member, IEEE, Hector Garcia-Molina, Member, IEEE P.P (2,4-5)
- [12] G. Bhargava, P. Goel, and B.R. Iyer, "Hypergraph Based Reorderings of Outer Join Queries with Complex Predicates," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '95), pp. 304-315, 1995
- [13] Rudragouda G Patil Dept Of CSE, The Oxford College Of Engg, Bangalore. International Journal Of Computer Applications In Engineering Sciences [VOL I,ISSUEII, JUNE 2011] [ISSN:22314946]P.P(1,4) Development Of Data Leakage Detection Using Data Allocation Strategies
- [14] C. Ordonez, "Integrating K-Means Clustering with a Relational DBMS Using SQL," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 2, pp. 188-201, Feb. 2006.
- [15] Shabtai, a.Gershman, M. Kopeetsky, y.EloviciDeutsche Telekom Laboratories at Ben-Gurion University, Israel. Technical Report TR-BGU-2409-

- [16] 201024Sept.20101A Survey of Data Leakage Detection and Prevention Solutions P.P(1-5, 24-25)
- [17] Panagiotis Papadimitriou¹, Hector Garcia-Molina² Stanford University 353 Serra Street, Stanford, CA 94305, USA P.P(1,4-5) A Model for Data Leakage Detection
- [18] Web-based Data Leakage Prevention Sachiko Yoshihama¹, Takuya Mishina¹, and Tsutomu Matsumoto² ¹ IBM Research - Tokyo, Yamato, Kanagawa, Japan fsachiko@
- [19] S. Sarawagi, S. Thomas, and R. Agrawal, "Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '98), pp. 343-354, 1998.
- [20] H. Wang, C. Zaniolo, and C.R. Luo, "ATLAS: A Small But Complete SQL Extension for Data Mining and Data Streams," Proc. 29th Int'l Conf. Very Large Data Bases (VLDB '03), pp. 1113-1116, 2003
- [21] H. Garcia-Molina, J.D. Ullman, and J. Widom, Database Systems: The Complete Book, first ed. Prentice Hall, 2001.
- [22] Archie Alimagno California Department of Insurance P.P (27), The Who, What, When & Why of Data Leakage Prevention/Protection
- [23] An ISACA White Paper Data Leak Prevention P.P(3-7) [14] Mr. V. Malsoru, Naresh Bollam/REVIEW ON DATA LEAKAGE DETECTION, International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com Vol.1, Issue3, pp.1088-1091 1088
- [24] SHUBHANSHU GUPTA, S. KOLANGIAMMAL, T. PADMAPRIYA, "Smart Curtain Using Internet Of Things" International Innovative Research Journal of Engineering and Technology, Vol. 2, Special Issue, pp. 4-8.
- [25] S.V. Manikanthan and K. Srividhya "An Android based secure access control using ARM and cloud computing", Published in: Electronics and Communication Systems (ICECS), 2015 2nd International Conference on 26-27 Feb. 2015, Publisher: IEEE, DOI: 10.1109/ECS.2015.7124833.

