Computing, Information Systems, Development Informatics & Allied Research Journal
Vol. 7 No. 3, September, 2016 - www.cisdijournal.net

A Hybridized Technique for Optimizing Query Processing in Distributed Database Systems

Ogheneovo, E.E. & Oviebor, E.
Department of Computer Science
University of Port Harcourt
Port Harcourt, Nigeria.
edward_ogheneovo@yahoo.com
emadgr8one@yahoo.com

ABSTRACT

Optimizing query processing in distributed database system is an important research area considering the volume of data and information being processed these days. Many techniques have been proposed for optimizing query processing in distributed databases. In this paper, we proposed a combination of two of the most commonly used techniques for optimizing query: data-shipping and query-shipping techniques. This hybridized technique provides solution for storing and processing data and information for quick retrieval of information in a distributed database when the data to be retrieved are not located within a single computer system. Using this technique, each employee in an organization that is geographically located in different regions can decide to hide information about any of the employee whose data is in the organization's database by preserving their individual query intent. In the client machine, the data-shipping technique is used to help in the local processing and communication cost while the query-shipping technique is used in the server machine to optimize the given data selection. In some cases data management (update) can also be carried out by the database administrator thus the technique combines the features of both the data-shipping and query-shipping techniques. We compared the results obtained with previous results when the techniques were separately used and we discovered that our technique performs better in terms of the time and complexities of the algorithm used.

Keywords: Distributed database, optimization, query processing, join, replica, client, server

CISDI Journal Reference Format
Ogheneovo, E.E. & Oviebor, E. (2016): A Hybridized Technique for Optimizing Query Processing in Distributed Database Systems. Computing, Information Systems, Development Informatics & Allied Research Journal. Vol 7 No 3. Pp 7-14
Available online at www.cisdijournal.net

1. INTRODUCTION

The Optimization of a query [(Berstein et al., 1981; Kossmann, 2000) within distributed database environment involves the process whereby multiple query plans are generated to satisfy that query. The actual updating and retrieval of data is performed through various low- level operations (Reza et al., 2001). The user communicates with the database by submitting requests for data selection (queries) or management (updates) in a distributed system. However, given the existence of exhaustive enumeration algorithms that can find the optimal solution for queries with dozens of relations very quickly (Jake, 1984; Raipurkarn and Bamnote, 2013; Freeman, 1989, and Karwin, 1996), there are few cases where resorting to heuristics or disabling bushy trees should be necessary. For every query that is initialized whether in a centralized or distributed database system, it must be optimized by the optimizer to generate the best and the less expensive query evaluation plan from a set or sequence of equivalent query evaluation plan (Du et al., 1982).

Distributed databases (Hameurlain and Morvan, 2009; Yu and Chang, 1984) are databases that locate their storage devices at various participating sites over a communication network. In the processing of a query, users of the database usually state the information or data that is needed instead of identifying the process to get the needed information or data in the distributed environment. Thus distributed database systems are group of sites connected on common high-bandwidth network (Kulkarni, 1994; Padia et al., 2015). A distributed system must have an assemblage of sites interconnected in a single communication network with high communication capacity (bandwidth). Distributed query works very effectively when it relies on the capacity of the query optimizer to formulate an effective query processing strategies. Once the query, entered by the user, is transformed into a standard relational algebra form; the optimizer searches for an optimal query execution plan (Cattell, 1994; Chung and Iranil, 1986).

A distribute database system allows applications or users to access data from local and remote databases (Su and Ozsoyoglu, 1991; Doshi and Raisinghani, 2011; Alomy et al., 2009). A distributed database is a single logically interrelated database that is spread across computers in multiple sites that are connected by a data communications network (Kim et al. 2009; Kyuseok and Surajit, 1996). It must be noted that a distributed database is truly a database; it is not a loose collection of files as may be perceived. In a distributed database, the network must allow users to share the data available in the databases. For example, a user at node A must be able to access (and perhaps update) data at node B depending on the type of distributed database in use. The distributed database, even though can be accessed (and perhaps updated) at different locations; it is still centrally administered as a corporate resource while providing local flexibility and customization to users in organizations and enterprises (Duncan, 1996). Thus the DBMS keeps track of the locations of the data so they are consistent and that users are not aware of the distributed nature of the database. To ensure that the database is consistent, up-to-date, and current, the DBMS ensures that the data in the distributed database are duplicated and replicated during transactions and update processes (Nick et al., 1995; Qian et al., 1993).

In this paper, we proposed a combination of two of the most commonly used techniques for optimizing query: data-shipping and query-shipping techniques. This hybridized technique provides solution for storing and processing data and information for quick retrieval of information in a distributed database when the data to be retrieved are not located within a single computer system. Using this technique, each employee in an organization that is geographically located in different regions can decide to hide information about any of the employee whose data is in the organization's database by preserving their individual query intent. In the client machine, the data-shipping technique is used to help in the local processing and communication cost while the query-shipping technique is used in the server machine to optimize the given data selection .In some cases, data management (update) can also be carried out by the database administrator thus the technique combines the features of both the data-shipping and query-shipping techniques. We compared the results obtained with previous results when the techniques were separately used and we discovered that our technique performs better in terms of information retrieval.

2. SIGNIFICANCE OF THE STUDY

Optimizing query processing in distributed database system is an important research area considering the volume of data and information being processed these days. Many techniques have been proposed for optimizing query processing in distributed databases but these techniques are not efficient either in terms of cost overhead or memory utilization. Based on this, we proposed a technique that combines two major approaches in query optimization taking advantages of each of the techniques and ensuring their limitations are brought to barest minimum. The technique has the ability to preserve the intension (privacy) of a query in a distributed environment and it allows relational processing of query in the server as well as client machine. The result shows a significant improvement when compared with existing techniques by ensuring that memory usage and cost overhead are brought to a minimal levels.

3. METHODOLOGY

This technique allows the optimizer to make choices of performing certain processing of query at server machine, as well as certain part involving processing at client machine. The technique demands that the client machine has at minimum basic query processing abilities, i.e. the optimizer would simply choose plans permitted through client's query processor. Hybridized shipping technique permits the system to make good use of the client resources. In fact, the hybridized technique can noticeably minimize communication cost if data-inflating processes can be carried out at the client rather than the server. Additionally, the hybridized shipping technique can profit from making use of the client disk properties corresponding to server disk properties. The proposed technique uses randomized optimizer for the user defined function, the execution space will be based on distributed join trees, whose interior nodes have as a minimum of one from every two child nodes as leaf, in our algorithm, i is the primary key in the table which can be replicated and distributed over the set of global relations. The query input for our algorithm are: select, project, and join (SPJ) query input on tables or relations $r_1,...,r_m$.

Figure 1 shows the architecture of our model. As seen in figure 1, when a query is sent to the database, the program checks if it is sent to the client machine or server machine and if it is a client machine, the application uses data shipping technique in order to achieve better result at the lowest cost and time. However, if it query is sent to the server (still through the client machine), it uses the query shipping technique. As soon as the client machine receives the query, it sends it to the server and both the base copy and the replica of the query are scanned and the resultants are joined together to form inner and outer queries. These are further joined together and the result is displayed and a copy or replica is sent to the user as the results of the query at an efficient and faster time. The white color represents client, green represents server 1, and blue represents server 2.

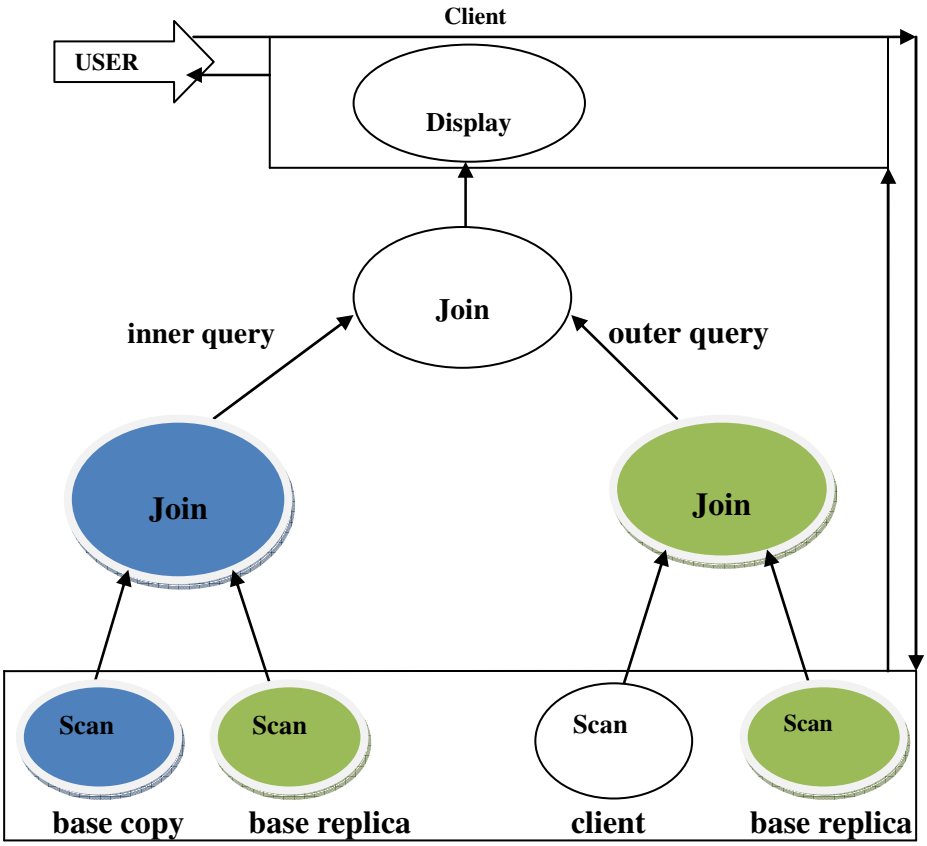

Fig. 1: Architecture of hybridized shipping technique

To ensure minimum resource consumption and minimum response time, the scan operation executes relational operations at various sites of the distributed database so as to reduce the number of relations that contains the query to be executed through reduction process. The results from the inner and outer queries are then joined in the distributed database having different sites and then transmitted to a designated site where they query is executed locally at this site. Figure 2 shows how these queries from different sites in a distributed database are joined using the join operator.

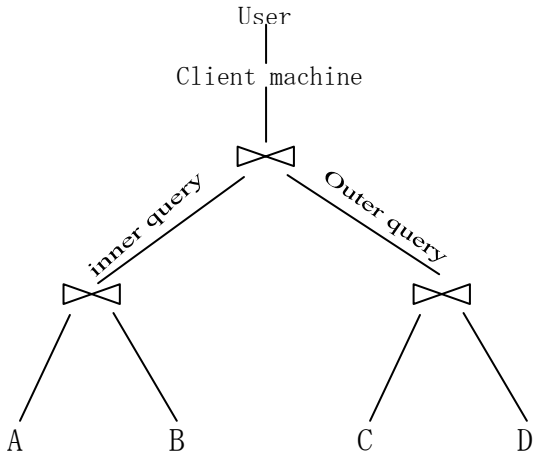

Fig. 2: Queries from different sites in a distributed database

The process works as follows: consider a distributed database system with different sites A, B, C, and D. Data are scanned and collected from sites A and B are joined; also, data are collected from sites C and D and joined to form inner and outer queries respectively by reduction process. The collected data are further joined to a local site where it is processed as shown in figure 2. The final result after execution is then transmitted to the client machine and then displayed for the user. Table 1 shows the query operators used for the hybridized technique.

Table 1: Query operators used for the proposed technique.

Query Operator	**Query Shipping Technique**	**Data Shipping Technique**	**Hybridized Shipping Technique**
Display	Client	Client	Client
Join	Outer or Inner Relation	Consumer (i.e., Client)	Client, Outer, or Inner Relation
Select	Producer	Consumer (i.e., Client)	Producer or consumer
Scan	Primary Copy	Client	Primary copy or client

Algorithm 1 is used for optimizing query processing in a distributed database having many relations as a result of many sites involved.

Algorithm 1: Algorithm for optimizing a query processing

```
1: Input: SPJ Query Q on relations R1,…,Rm
2: Output: A query plan
3: begin
4:   for (i) = 1 to m do {
5:       bestplan({R1}) = accessplans(Ri)
6:     trimplans(bestplan({Ri}))
7:   }
8:     for (i) := 2 to m do {
9:       for every S ⊆ {R1, . . . ,Rm} s.t.||S||=i do {
10:         bestplan (S) = a replica plan with unlimitrd cost
11:              for every O ⊂ S do {
12:                 bestplan(S) = bestplan(S) U joinplans(bestplan(O), bestplan(S-O))
13:                trimplans(bestplan(S))
14:              }
15:          optplan:=bestplan:
16:         }
17:     }
18:  return optplan({R1,…,Rm}
19: end.
```

Fig. 2: Algorithm for optimizing a query processing

The algorithm in figure 2 shows that different query plans can be formulated but the algorithm helps to trim the plans in order to get the best plan obtained from the optimization process.

4. RESULTS AND DISCUSSION

Table 2 is shows the results of the application. As seen in the table, when a query is sent to the database, the program checks if it is to the client machine or server machine and if it is a client machine, the application uses data shipping technique in order to achieve better result at the lowest cost and time. Table 2 contains standard values for the experimental setup results in the various tables below. Every data entry is accounted for during execution. The time and disk space taken for the processing and optimization of each data entry had their S. I. unit in second (s) and kilobyte (kb) respectively.

Table 2: Experimental setup

Number of data entry	CPU time for optimizing data entry (s)	Disk space for a data entry (kb)
50	2.7	500
70	3.7	700
95	5.0	950
105	5.6	1050

Table 3: Queries Optimized at the client machine

Number of entries	Percentage due to query %	Data-shipping Technique		Query-shipping Technique	
		Disk space (kb)	Time (s)	Disk space (kb)	Time (s)
300	15	450	2.4	3000	16
280	13	364	1.9	2800	14.9
250	5	125	0.7	2500	13.3
190	9	171	0.9	1900	10.1
150	10	150	0.8	1500	8.0
130	17	221	1.2	1300	6.9
120	7	84	0.4	1200	6.4

Table 3 illustrated how, the number of entries in a relation can be worked on by the use of data-shipping and query-shipping technique. The percentage due to query, is gotten from a particular structured query language (SQL), stating some conditions that involves the simplest expression (mapping) in a relation, it is only possible when data-shipping technique is used in the client machine and query-shipping technique in the server machine. We observed that that disk space and time spent during query processing and optimization is considerably smaller using data-shipping technique in the system than query-shipping technique. Data-shipping technique only account for (i.e., disk space and time) data that matches the mapping which gives the percentage from the structured query language, while query-shipping accounts for virtually all the data based on the number of entries. In Figure 4 we indicate Y as the number of entries in a relation, X1 is data-shipping technique which performances optimally at the client environment and finally, X2 is the query-shipping technique that uses more disk space and time. In Figure 4 we plotted the value of time spent on data entries by both X1, and X2 against Y. Both graphs showed that data-shipping technique is optimal in the client side.

y	x1	x2
300	450	3000
280	364	2800
250	125	2500
190	171	1900
150	150	1500
130	221	1300
120	84	1200

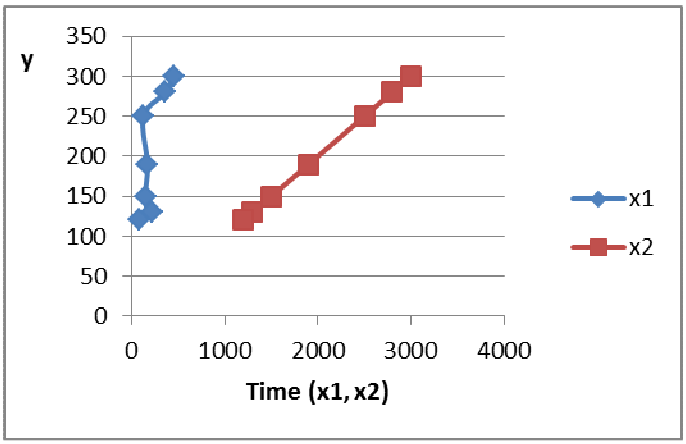

Fig 4: The disk space(s) used by x1, and x2 against y

Y	X1	X2
300	2.4	16
280	1.9	14.9
250	0.7	13.1
190	0.9	10.1
150	0.8	8
130	1.2	6.9
120	0.4	6.4

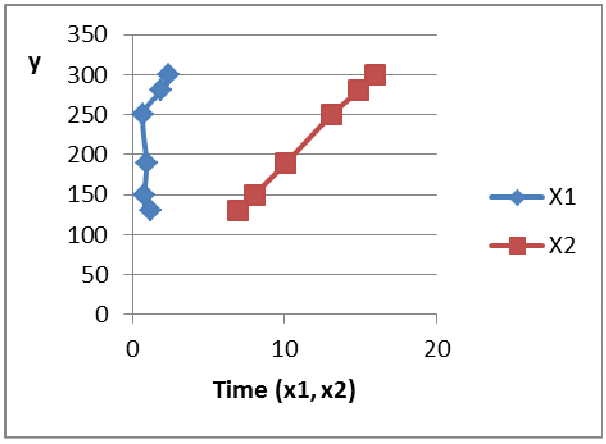

Fig. 5: The time(s) spent by x1, and x2 against y

Table 4 shows queries that are processed and optimized at the server machine, here query-shipping technique performed optimally when compared to data-shipping technique. Lesser disk space and time were spent on data. Figure 5 indicates that query shipping technique uses less disk space than data-shipping technique at the server environment. Figure 6 depicts time spent on data entries when we made use of both techniques, here query-shipping technique uses fewer than data-shipping technique. The same process is performed to get the data in figure 7

Table 4: Queries Optimized at the Sever Machine

Number of entries	Percentage due to query %	Query-shipping Technique		Data-shipping Technique	
		Disk space (kb)	Time (s)	Disk space (kb)	Time (s)
300	15	450	2.4	3000	16
280	13	364	1.9	2800	14.9
250	5	125	0.7	2500	13.3
190	9	171	0.9	1900	10.1
150	10	150	0.8	1500	8.0
130	17	221	1.2	1300	6.9
120	7	84	0.4	1200	6.4

Y	X1	X2
300	3000	450
280	2800	364
250	2500	125
190	1900	171
150	1500	150
130	1300	221
120	1200	84

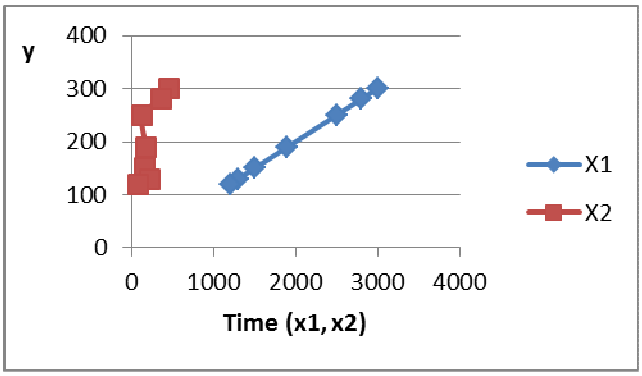

Fig 6: The disk space(s) used by x1, and x2 against

Y	X1	X2
300	16	2.4
280	14.9	1.9
250	13.1	0.7
190	10.1	0.9
150	8	0.8
130	6.9	1.2
120	6.4	0.4

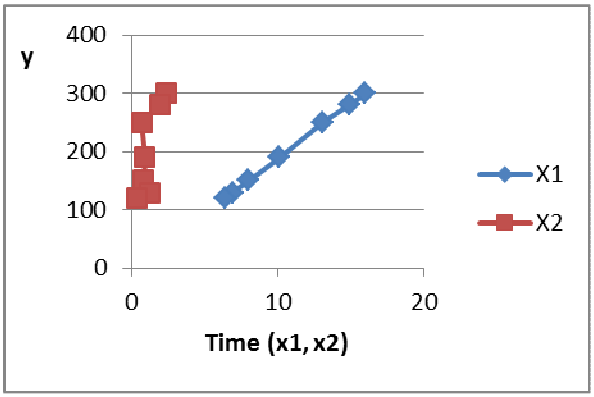

Fig 7: The time(s) spent by x1, and x2 against

5. CONCLUSION

This paper proposed a hybridized that combines the data-shipping and the query-shipping techniques for processing queries for easy and faster retrieval from databases that are geographically distributed. The aim and objectives of developing this application has been achieved with the administrator managing the server while the employee is at the client machine. We designed this application in such a way that when a user either at the client machine or server machine logs in, the ID and user name of the person is displayed to show us the particular user. Both query and data shipping techniques were applied to attain the requirement of this research work. We compared the results obtained with previous results when the techniques were separately used and we discovered that our technique performs better in terms of the time and complexities of the algorithm used. The major models for server-client query evaluation are query and data shipping techniques.

Initially, we describe the course of actions relation to the limits they put on operator node selection throughout the optimization of a query in distributed environment**.** This technique for Optimizing Query processing in a distributed database system is a computerized solution for storing the details of employees in an organization and task assigned to an employee by an organization. As long as useful optimizer could be designed for deciding the suitable option, the technique has the capacity, at all times, to accomplish at minimum as good as any of the two pure shipping techniques, and most times, considerably superior. Therefore, we conclude that the application has been developed using the recommended techniques in order to reduce cost and harness both human and natural resources.

REFERENCES

1. P. A. Bernstein, N. Goodman, E. Wang, C. L. Reeves, and J. B. Rothnie (1981). Query Processing in a System for Distributed Databases (SDD-1), ACM Transactions on Database Systems (TODS), pp. 602-625.
2. D. Kossmann (2000). The State of the Art in Distributed Query Processing,ACM Computing Surveys, Vol. 32, No. 4, pp. 422-469.
3. S. Reza, Z. Carlo, Z. Amir and A. Jafar (2001). *Optimization of Sequence Queries in Database Systems*. In Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems.
4. M. Jarke (1984). Query Optimization in database Systems, ACM Computing Surveys, Vol. 16, No. 2, pp. 111-152.
5. A. Raipurkar and G. R. Bamnote (2013). Query Processing in Distributed Database Through Data Distribution, Int'l Journal of Advanced Research in computer and Communication Engineering, Vol. 2, Issue 2, pp. 1134-1139.
6. M. Freeman (1989). Hybrid data distribution strategy Database, Programming and Design, April, Vol. 2, No. 4.
7. Karwin, B. (1996). InterBase Server Configuration and Optimization. Borland Developer's Conference.
8. Du, W., Krishnamurthy R., and Shan M.C. (1992). Query Optimization in Heterogeneous DBMS. In Proceedings of the 18th Int'l Conference on Very Large Databases, Vancouver.
9. A. Hameurlain and F. Morvan (2009). Evolution of Query Optimization Methods, Trans. on Large Scale Data and Knowledge Cent. Syst.I.
10. C. T. Yu and C. C. Chang (1984). Distributed Query Processing, Computing Surveys, Vol. 16, No. 94, pp. 399-433
11. K. Kulkarni (1994). Object-Oriented Extensions in SQL3: A Status Report. *ACM SIGMOD Conf.*, Minneapolis.
12. S. Padia, S. Khulge, A. Gupta and P. Khadilikar (2015). Query Optimization Strategies in Distributed Databases, Int'l Journal of Computer Science and Information Technologies, Vol. 6, No. 5, 2015, pp. 4228-4234.
13. R. G. G. Cattell (1994). *Object Database Standard*. Morgan-Kaufmann Publishers, San Mateo.
14. C.-W. Chung and K. B. Irani (1986). An Optimization of Queries in Distributed Database Systems, Journal of Parallel and Distributed Computing, Vol.3 3, pp. 137-157.
15. T. Su and G. Ozsoyoglu (1991). Controlling FD and MVD Inference in Multilevel Relational Database Systems. *IEEE Transactions on Knowledge and Data Engineering*.
16. P. Doshi and V. Raisinghani (2011). Review of Dynamic Query Optimization Strategies in Distributed Database, Electronics Computer Technology (ICECT), In Proceedings of the 3rd IEEE Int'l Conference, Vol. 6, pp. 145-149, 8-10 April, 2011, Kanyarkumari. Doi: 10.1109/ICECTECH.2011.5942069.
17. B. M. M. Alomy, F. A. Henskens and M. Hannaford (2009). Query Processing and Optimization in Distributed Database System, Int'l Journal of Computer Science and Network Security, Vol. 0, No. 9, pp. 143-152.
18. J. Kim (1986). A Method for Limiting Disclosure of Microdata Based on Randomize and Transformation *Proceedings of the Section on Survey Research Methods of the American Statistical Association*.
19. S. Kyuseok and C. Surajit (1996). Optimization of Queries with User-Defined Predicates. In Proc. Int'l Conf. on VLDB, Mumbai(Bombay), India.
20. G. Duncan (1991). Enhancing Access to Data while Protecting Confidentiality: Prospects for the Future. *Statistical Science*.
21. R. Nick, M. C. Chungmin, K. Stephen, D. Alexios and P. Yannis (1995). The ADMS Project: Views "R" Us. Bulletin of the Technical Committee on Data Engineering, June.
22. X. Qian, M. Stickel, P. Karp, T. Lunt and T. Garvey (1993). Detection and Elimination of Inference Channels in Multilevel Relational Database Systems. In *Proc. of the IEEE Symposium on Research in Security and Privacy*.