American-Eurasian Journal of Scientific Research 11 (2): 72-78, 2016
ISSN 1818-6785
© IDOSI Publications, 2016
DOI: 10.5829/idosi.aejsr.2016.11.2.22729

Multi Level Relational Mapping Algorithm Based Dependency Rule Generation for Query Optimization

[1]*Tejy Johnson and* [2]*S.K. Srivatsa*

[1]Department of Computer Applications,
Dr.MGR Educational Research and Institute University, Maduravoyal, Chennai-95
[2]Department of CSC, Prathyusha Institute of Technology and Management (PITAM),
TIRUVALLUR – 602025, Chennai-602 025, India

Abstract: The problem of query optimization has been approached in several methods but suffers with the problem of accuracy. To overcome this issue and to improve the performance of our previous solution, we propose a multi level relational mapping algorithm in this paper. The method first identifies the relational objects and generates relational maps. From the relational maps the method identifies the objects and entity of query. Based on the above the method generates different rules to perform the query and computes the dependency measure for each part of the query. The use of relational map helps to identify the query dependency according to the object and to compute the dependency measure for each of the rule being produced. Finally a subset of dependency rule is produced as a result and the method improve the performance of rule generation and improves the performance of query optimization by scheduling the execution of query parts efficiently.

Key words: Relational Database · Query Optimization · Dependency Rules · Relational Mapping

INTRODUCTION

The modern data base maintains various schema and each has various relations. The information about any product has no limit in size and can be presented in a relational manner. For example, the information about a single human can be split into number of categories like personal, official, financial and so on. The personal information is about the personal detail about the person which has name, parent name, age and sex and so on. Further the personal information can be split into the address and personal information. The address itself can be stored in a relational data base and can be stored as a new entity. Similarly the information about any object can be presented and stored in number of data tables where there is relation between the entities present in different tables of any data base. Such relational entities stored in any data base can be named as relational data base.

The relational database helps organizing the data tuples in more efficient manner where the retrieval also could be performed in more strategic manner. The query produced by any user may access number of relational data tables where the result of any part of the query can become an input to the other part of the query. So in order to execute the query part, some of the other part of the query has to be completed or if two different part of the query access the same relational data table then one has to be wait till the other part of the query has to be finished. This introduces dependency in executing the query. Also, the input query can be split into number of small query parts and the query optimization is performed according to the sequence of query execution. By executing the query parts in different sequence the time complexity will vary and to execute the query in more efficient manner the time complexity has to be minimized.

In general the query optimization is performed by splitting the query into number of small query parts and execute them in different order in such a way to reduce the time complexity. However the query parts has different dependency between them and the earlier methods does not handles this issue to reduce the time complexity and to improve the performance of query optimization. By identifying the query parts and their dependency between them, we can g enerate the dependency rules from which

Corresponding Author: Tejy Johnson, Department of Computer Science,
Ethiraj College for Women, Egmore, Chennai-8, India.

the dependency parts can be identified and can produce a sequence of query execution. There are number of dependency rule generation approaches has been specified earlier but suffers with the efficiency of query optimization.

This paper discusses about the multi level relational mapping to produce the dependency rules. Unlike earlier dependency rule generation, the method identifies the dependency between the query parts in multiple levels and generates rules according to the multiple level dependency and generate sequence of query execution using the relational mapping. Because, the relational table has number of relation between other tables and if a dependency object has other dependency in the relation with other relational object the earlier methods ignores this issue. The proposed approach handles this issue to identify the multi level dependency issue to produce dependency rules.

Related Works: There are number of rule generation approaches has been declared for the problem of dependency rule generation and we discuss some of the methods here in this section.

Spatial Query Optimization Based on Transformation of Constraints [1], describes a problem of spatial query optimization. The processing of such queries is a new area of rapidly developing domain of spatial databases. The main scope of considerations is the impact of constraints type on the speed of execution. Transformation of logical formulas is proposed for some kind of queries as a method of optimization. Proposed decompositions of queries were done according to the logic and set theory. It is experimentally proved that the presented way of optimization is efficient.

An adaptive range-query optimization technique with distributed replicas [4], present a compile-time/run-time approach for minimizing the response time of 2-dimensional range when a distributed replica of a dataset exists. The aim is to partition the query payload (and its range) into subsets and distribute those to the replica nodes in a way that minimizes a client's response time. However, since query size and distribution characteristics of data (data dense/sparse regions) in varying ranges are not known a priori, performing efficient load balancing and parallel processing over the unpredictable workload is difficult. A technique based on the creation and manipulation of dynamic spatial indexes for query payload estimation in distributed queries was proposed. The effectiveness of this technique was demonstrated on queries for analysis of archived earthquake-generated seismic data records.

Optimizing Aggregate Query Processing in Cloud Data Warehouses [6], study and optimize the aggregate query processing in a highly distributed Cloud Data Warehouse, where each database stores a subset of relational data in a star-schema. Existing aggregate query processing algorithms focus on optimizing various query operations but give less importance to communication cost overhead (Two-phase algorithm). However, in cloud architectures, the communication cost overhead is an important factor in query processing. Thus, we consider communication overhead to improve the distributed query processing in such cloud data warehouses. We then design query-processing algorithms by analyzing aggregate operation and eliminating most of the sort and group-by operations with the help of integrity constraints and our proposed storage structures, PK-map and Tuple-index-map. Extensive experiments on PlanetLab cloud machines validate the effectiveness of our proposed framework in improving the response time, reducing node-to-node interdependency, minimizing communication overhead and reducing database table access required for aggregate query.

Implementing Semantic Query Optimization in Relational Databases [15], generates an alternate query using the learning framework and the algorithm. The alternate query should be semantically equivalent to original query. The semantically equivalent query generated should be less expensive than the original query. These can be implemented in SQL using the SQL hints. These hints allow user to implement the desired plan for the query.

Optimization of Linear Recursive Queries in SQL [16], present SQL 16438 Mrs.Tejy Johnson and Dr.S.K.Srivatsa implementations for two fundamental algorithms: Seminaive and direct. Five query optimizations are studied: 1) storage and indexing; 2) early selection; 3) early evaluation of nonrecursive joins; 4) pushing duplicate elimination; and 5) pushing aggregation. Experiments compare both evaluation algorithms and systematically evaluate the impact of optimizations with large input tables. Optimizations are evaluated on four types of graphs: binary trees, lists, cyclic graphs and complete graphs, going from the best to worst case. In general, Seminaive is faster than direct, except for complete graphs. Storing and indexing rows by vertex and pushing aggregation work well on trees, lists and cyclic graphs. Pushing duplicate elimination is essential for complete graphs, but slows computation for acyclic graphs. Early selection with equality predicates significantly accelerates computation for all types of graphs.

Semantic based Efficient Cache Mechanism for Database Query Optimization [17], provides semantic based cache mechanism techniques for optimizing the user queries. Here the frame work for optimization is analyzed which supports data and computation reuse, query scheduling and cache efficient utilization algorithm is presented in order to improve the evaluation process and minimize the overall response time. Further the case study is analyzed to test the performance and extended the same for multi-queries to achieve parallelism.

Enhancing Information Retrieval Efficiency Using Semantic-based Combined-Similarity-Measure [19], per we have proposed a new semantic based similarity measure in which each term can be a phrase or a single word and the weight assigned to each term is based on its semantic importance considering each sentence. We have used this semantic similarity measure along with other standard similarity measure as Jaccard and cosine to form the semantic-based-combined-similarity measure. Standard genetic algorithm has been used to optimize the weight given for each similarity measure.

All the above discussed approaches has the problem of identifying exact sequence of query execution and has not provide any efficient solution to reduce the query execution time.

Multi Level Relational Mapping Based Dependency Rule Generation: The multi level relational mapping approach reads the input query and parse them to identify the relational objects first. Then the method splits the input query into number of subset queries. Then for each input query the method performs the relational mapping to identify the number of relations a single query has. Finally the method performs the multi level relational mapping to produce the dependency rule. The entire process has been presented in four stages namely Preprocessing, Relational Map Generation, Dependency Measure Computation, Multi Level Relational Mapping, dependency rule generation. We explain each of the stage in detail in this section.

The Figure 1, shows the architecture of multi level relational mapping approach for dependency rule generation and its functional components.

Preprocessing: Preprocessing is the process of preparing the input query to generate the dependency rule generation. First the input query is read and the method reads the schema and identifies the set of all relational objects. Once the relational objects are identified then the method splits the input query into number of query partition based on the occurrence of key words from the input query. Identified relational objects and the query set will be given as a result.

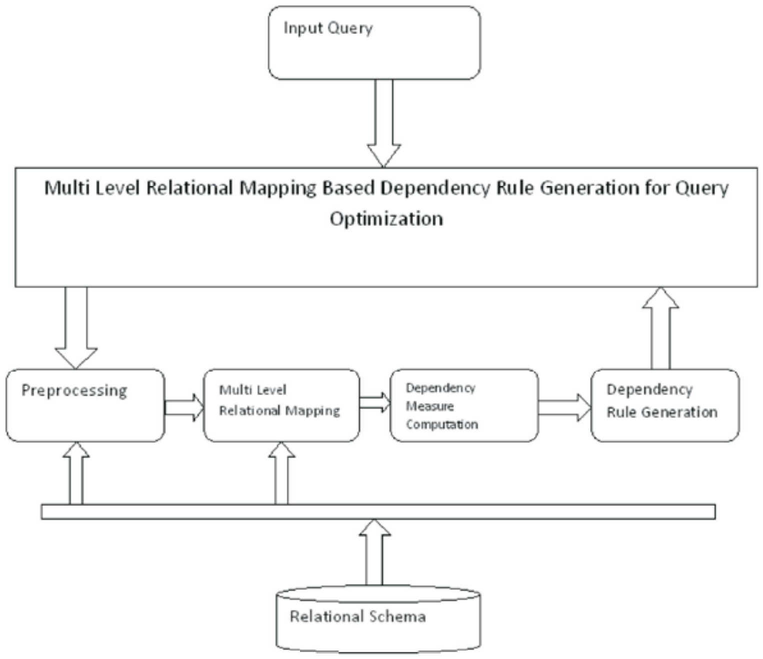
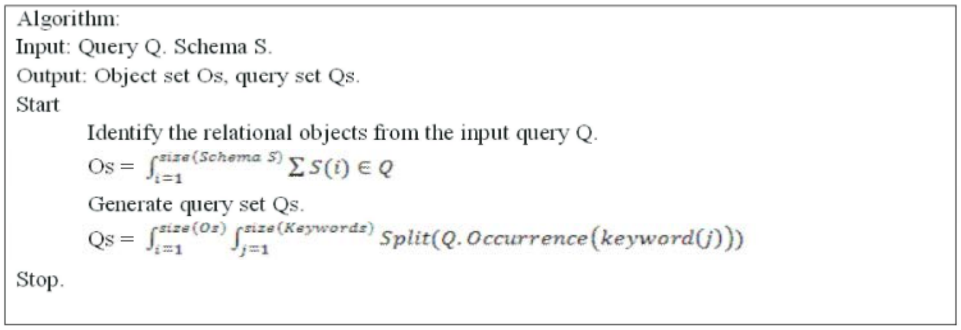

Input Query
Multi Level Relational Mapping Based Dependency Rule Generation for Query Optimization
Preprocessing
Multi Level Relational Mapping
Dependency Measure Computation
Dependency Rule Generation
Relational Schema

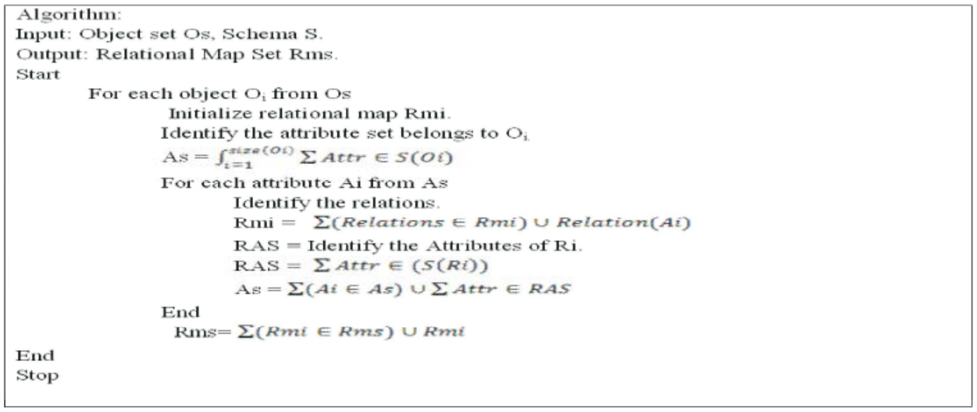
Fig. 1: Architecture of multi level relational mapping.



```
Algorithm:
Input: Query Q. Schema S.
Output: Object set Os, query set Qs.
Start
    Identify the relational objects from the input query Q.
    Os = $\int_{i=1}^{size(Schema\ S)} \sum S(i) \in Q$
    Generate query set Qs.
    Qs = $\int_{i=1}^{size(Os)} \int_{j=1}^{size(Keywords)} Split(Q.Occurrence(keyword(j)))$
Stop.
```

The above discussed algorithm identifies the set of all relational objects from the input query and identifies the query set.

Multi Level Relational Mapping: At this stage, the method reads the set of all relational objects being identified. For each object identified, the method reads the data base schema and identifies the relations a single object has with others. For each object there may be number of relations present in the data base schema, the method identifies such schemas. The method does not stop at a single level, buts identifies the relations iteratively in order to find the other objects relational mapping. Identified relational mapping will be used in computing the dependency measure.

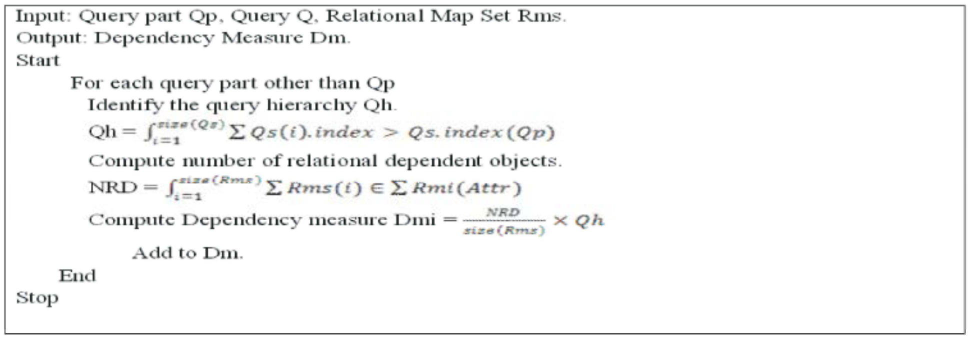


```
Algorithm:
Input: Object set Os, Schema S.
Output: Relational Map Set Rms.
Start
    For each object $O_i$ from Os
        Initialize relational map Rmi.
        Identify the attribute set belongs to $O_i$
        As = $\int_{i=1}^{size(Oi)} \sum Attr \in S(Oi)$
        For each attribute Ai from As
            Identify the relations.
            Rmi = $\sum(Relations \in Rmi) \cup Relation(Ai)$
            RAS = Identify the Attributes of Ri.
            RAS = $\sum Attr \in (S(Ri))$
            As = $\sum(Ai \in As) \cup \sum Attr \in RAS$
        End
        Rms= $\sum(Rmi \in Rms) \cup Rmi$
End
Stop
```

The above discussed algorithm performs the relational mapping in multiple levels and

The above discussed algorithm identifies the relations present in the each part of the query and adds to the relational map set.

Dependency Measure Computation: The dependency measure computation is performed using the relational maps being generated at the previous stage and the query hierarchy and the number of relational objects being shared by two different queries. The dependency measure represents that how depth the part of query is depend on another part of the query.



```
Input: Query part Qp, Query Q, Relational Map Set Rms.
Output: Dependency Measure Dm.
Start
    For each query part other than Qp
        Identify the query hierarchy Qh.
        Qh = $\int_{i=1}^{size(Qs)} \sum Qs(i).index > Qs.index(Qp)$
        Compute number of relational dependent objects.
        NRD = $\int_{i=1}^{size(Rms)} \sum Rms(i) \in \sum Rmi(Attr)$
        Compute Dependency measure Dmi = $\frac{NRD}{size(Rms)} \times Qh$
            Add to Dm.
    End
Stop
```

The above discussed algorithm computes the dependency measure for each part of the query and based on the value of the measure the dependency rule is generated.

Dependency Rule Generation: The dependency rule generation is performed with the support of all the above mentioned processes. For each part of the query the method generates the dependency rule using the other part of the query and the computed dependency measure. For each part of the query, the method computes the dependency measure and based on the value of all the measures the method generates the dependency rule which can be used to execute the query submitted.

```
Algorithm:
Input: Query Set Qs
Output: Dependency Rule DR.
Start
        Generate Rule DR.
        For each query part Qp from Qs.
                    Compute dependency measure Dm.
                    Dm = compute dependency measure(Qp).
                      Compute cumulative dependency measure CDM.
                    CDMi = $\frac{\sum Dm(i)}{size(Dm)}$
              DR = {CDM} ∪ $CDM$i
          End
        Sort the DR according to CDM value.
        Execute the query.
Stop.
```

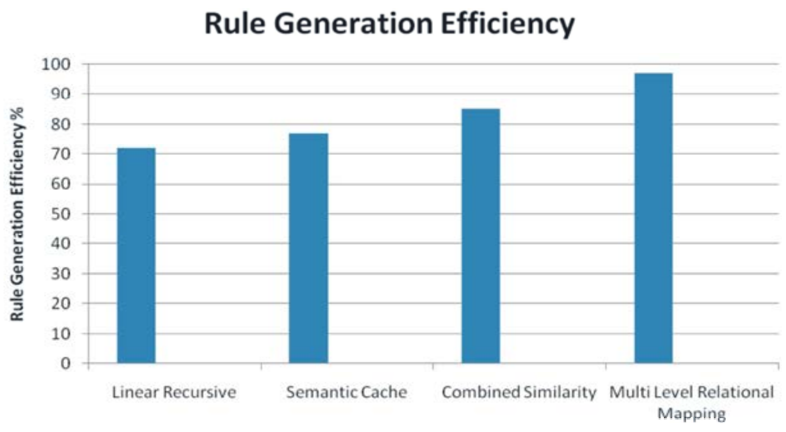
The above discussed algorithm computes the dependency rule and based on the value of cumulative dependency measure a final sequence will be selected to perform the query execution.

RESULTS AND DISCUSSION

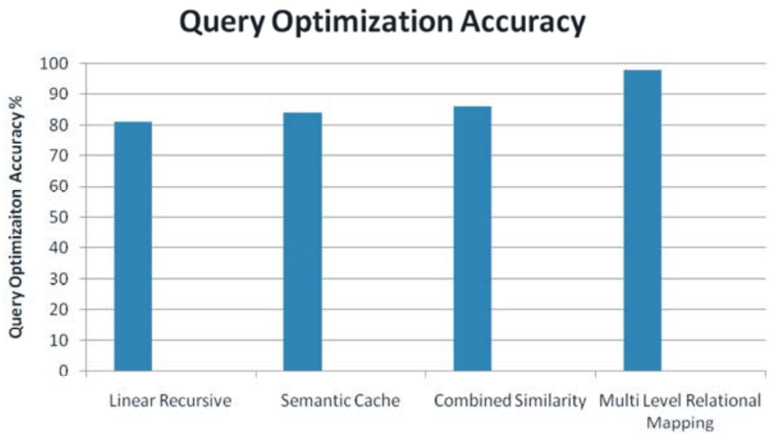
The proposed multi level relational mapping based dependency rule generation approach for query optimization has been implemented and evaluated for its efficiency using SQL data base. The method has been evaluated using various setup and has been produced efficient results on query optimization and dependency rule generation.

The Table 1, shows the details of evaluation parameter being used to evaluate the proposed method and the method has been evaluated for various parameters of query optimization.

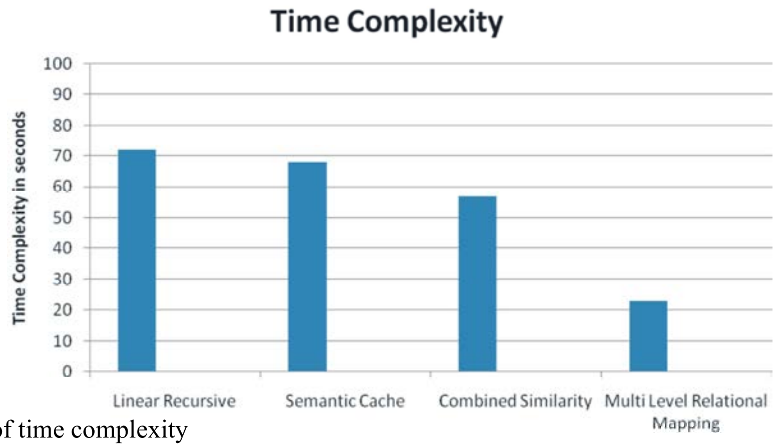
The Graph 1, shows the comparison of rule generation efficiency produced by different methods and it shows clearly that the proposed method has produced more rule generation efficiency than other methods.



Graph 1: Comparison of rule generation efficiency



Graph 2: Comparison of query optimization accuracy



Graph 3: Comparison of time complexity

Table 1: Details of evaluation parameter

Parameter	Value`
Tool Used	SQL
Number of relational objects used	200
Size of query used	75
Amount of Tuples	2 lakhs

The Graph 2, shows the comparison of query optimization accuracy produced by different methods and it shows clearly that the proposed method has produced higher accuracy than other methods.

The Graph 3, shows the comparison on time complexity produced by different methods and it shows clearly that the proposed method has produced less time complexity than other methods.

CONCLUSION

In this paper, the author described an efficient multi level relational mapping algorithm for dependency rule generation to improve the query optimization. The method starts with preprocessing the input query which identifies the relational objects and splits the input query into number of small queries. At the second stage, the method generates the multi level relational mapping which identifies the relational objects shared by the various parts of the query. The dependency rule generation approach computes the rule with the help of dependency measure computation algorithm which computes the measure based on the number of relations being shared by the query part and number of query parts is present in the top order. Based on the dependency measure and the rule being generated the method sorts the query part and based on the sorted order the query parts are executed to produce efficient results and reduce the time complexity also.

REFERENCES

1. Lupa Micha³ and Adam Piórkowski, 2014. Spatial Query Optimization Based on Transformation of Constraints, Springer, Advances in Intelligent Systems and Computing, 242: 621-629.

2. Lupa, M. and A.Piórkowski, 2012. Rule-based query optimizations in spatial databases. Studia, 33(2): 105–115.
3. Roumelis, G. and M.Vassilakopoulos, Corral, A.: Performance comparison of xBR-trees and R*-trees for single dataset spatial queries. In: Eder, J., Bielikova, M., Tjoa, A.M. (eds.) ADBIS 2011. LNCS, 6909: 228–242
4. Ahmet Sayar, Pierce Marlon and C. Geoffrey Fox, 2014. An adaptive range-query optimization technique with distributed replicas, Journal of Central South University,21(1):190-198.
5. Mauroux, C.P., E. WU and S. Madden, 2010.TrajStore: An adaptive storage system for very large trajectory data sets [C]// IEEE 26th International Conference on Data Engineering (ICDE). Long Beach, CA: IEEE Press, pp: 109-120.
6. Kurunji Swathi, Tingjian Ge, Xinwen Fu, Benyuan Liu, Amrith Kumar and Cindy X. Chen, 2014. Optimizing Aggregate Query Processing in Cloud Data Warehouses, Springer, Data Management in Cloud, Grid and P2P Systems Lecture Notes in Computer science, 8648: 1-12.
7. Kemper, A. and T. Neumann, 2011. Hyper: A hybrid OLTP and OLAP main memory database system based on virtual memory snapshots. In: IEEE 27th International Conference on Data Engineering (ICDE), Hannover, Germany, 195–206.
8. Vlachou, A., C. Doulkeridis, K.Norvag and Y. Kotidis, 2012. Peer-to-Peer Query Processing over Multidimensional Data. Springer, 16444 Mrs.Tejy Johnson and Dr.S.K.Srivatsa.
9. Curino, C., P.C.J. Evan, A.P. Raluca, N. Malviya, E. Wu, S. Madden and H. Balakrishnan, 2011. N.: Relational Cloud: A Database Service for the cloud. In: 5th Biennial Conference on Innovative Data Systems Research (CIDR), California, USA, 235-240.
10. Graefe, G., 2012. New algorithms for join and grouping operations. Journal Computer Science - Research and Development, 27(1): 3-27.
11. Kurunji, S., T. Ge, B. Liu and C.X. Chen, 2012. Communication Cost Optimization for Cloud Data Warehouse Queries,In: 4th IEEE International Conference on Cloud Computing Technology and Science Proceedings (CloudCom), Taipei, Taiwan, 512–519.
12. Srikanth, B., H.Li, J. Unmesh, Y. Zhu, L Vince and C.Thierry, 2013. Adaptive and Big Data Scale Parallel Execution in Oracle, International Journal on Very Large Data Bases (VLDB), 6(11): 1102–1113.
13. Cao, Y., R. Bramandia, C.Y. Chan and K.L. Tan, 2012. Sort-Sharing-Aware Query Processing. International Journal on Very Large Data Bases (VLDB), 21(3): 411–436.
14. Melita L., 2012. Genetic algorithms: An inevitable solution for query optimization in web mining — A review, Springer, Conference on science and education.
15. Jyoti Mor, Indu Kashyap and R K Rathy, 2012. Article: Implementing Semantic Query Optimization in Relational Databases. International Journal of Computer Applications, 52(9):41-46.
16. Carlos Ordonez, 2010. Optimization of Linear Recursive Queries in SQL; IEEE Transactions on Knowledge and Data Engineering, 22(2): 264-277.
17. Mohan Kumar P. and J. Vaideeswaran, 2012. Semantic based Efficient Cache Mechanism for Database Query Optimization, International Journal of Computer Applications, 43(23): 14-18.
18. Panus Jan and Josef Pirkl, 2010.Testing of Oracle database utilization, International Journal of Applied Mathematics and Informatics, 3(4): 58-65.
19. Mayank Saini, Sharma Dharmendar and P. K. Gupta, 2011.Enhancing Information Retrieval Efficiency Using Semantic-based-Combined-Similarity-Measure, International Conference on Image Information Processing (ICIIP 2011), IEEE Computer Society.