International Journal of Modern Engineering Research (IJMER)
www.ijmer.com *Pp-116-119* *ISSN: 2249-6645*

Extract, Transform and Load (ETL) Performance Improved by Query Cache

Mr. Udhaya Kumar P[1], Mrs. Selvi R[2]
P.G. Student[1], Assosiate Professor[2]
Department of Computer Science & Engineering, Aarupadai Veedu Institute of Technology, Paiyanoor

Abstract: *Extraction, Transformation, and Loading (ETL) processes are responsible for the operations taking place in the back stage of a data warehouse architecture Extract, transform and load (ETL) is the core process of data integration and is typically associated with data warehousing. ETL tools extract data from a chosen source, transform it into new formats according to business rules, and then load it into target data structure. Managing rules and processes for the increasing diversity of data sources and high volumes of data processed that ETL must accommodate, make management, performance and cost the primary and challenges for users. ETL is a key process to bring all the data together in a standard, homogenous environment. ETL functions reshape the relevant data from the source systems into useful information to be stored in the data warehouse. Without these functions, there would be no strategic information in the data warehouse. If source data taken from various sources is not cleanse, extracted properly, transformed and integrated in the proper way, query process which is the backbone of the data warehouse could not happened In this paper we purpose an ultimate advance approach which will increase the speed of Extract, transform and load in data ware house with the support of query cache. Because the query process is the backbone of the data warehouse It will reduce response time and improve the performance of data ware house.*

I. INRODUCTION

ETL is a data integration function that involves extracting data from outside sources (operational systems), transforming it to fit business needs, and ultimately loading it into a data warehouse [1] To solve the problem, companies use extract, transform and load (ETL) technology, which includes reading data from its source, cleaning it up and formatting it uniformly, and then writing it to the target repository to be exploited . The data used in ETL processes can come from any source: a mainframe application, an ERP application, a CRM tool, a flat file or an Excel spreadsheet. [2]. According to some industry experts approximately 60-80 percent of a data warehousing project effort is spent on this process alone. In today's high volume, client/server environment data acquisition techniques have to coordinate staging operations, filtering, data hygiene routines, data transformation and data load techniques in addition to cooperating with network technology to populate the data warehouse and operational data stores[6]. ETL functions reshape the relevant data from the source systems into useful information to be stored in the data warehouse. Without these functions, there would be no strategic information in the data warehouse. if source data taken from various sources is not cleanse, extracted properly, transformed and integrated in the proper way, query process which is the backbone of the data warehouse could not happened [7]. It's more than just in the data acquisition process. While data acquisition is the predominant process using the ETL tools, the data delivery process and movement of data from the analytical functions to the ODS or operational systems use ETL processing as well. The full-blown set of ETL operations must combine into a cohesive, integrated system. A system that ensures each process will fit into the overall effort efficiently, determines how the tool will be used for each component and synchronizes all ETL events. There should be ETL expert in the organization who ensures that the ETL processes have strength and endurance [6].

Extract, Transform, Load; three database functions that are combined into one tool that automates the process to pull data out of one database and place it into another database. The database functions are described following [9].

Extract -- The extraction step is conceptually the simplest task of all, with the goal of identifying the correct subset of source data that has to be submitted to the ETL workflow for further processing. The process of reading data from a specified source database and extracting a desired subset of data.
Transform -- the process of converting the extracted/ acquired data from its previous form into the form it needs to be in so that it can be placed into another database. Transformation occurs by using rules or lookup tables or by combining with other data.
Load -- the process of writing the data into the target database An alternative approach to information integration is that of mediation: data is extracted from original data sources on demand when a query is posed, with transformation to produce a query result [8].
Workflow Diagram of ETL

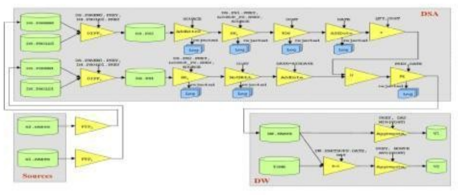

Figure 1: An example ETL workflow

II. ROLE AND RESPONSBILITY OF ETL

The ETL architect must have the following roles and responsibilities. The ETL architect should have a close eye on the needs and requirements of the organization. He/she must understand the overall operational environment and strategic performance requirements of the proposed system. The architect must interact with the source system operational and technical staff, the project database administrator (DBA) and the technical infrastructure architects to develop the most efficient method to extract source data, identify the proper set of indexes for the sources, architect the staging platform, design intermediate databases needed for efficient data transformation and produce the programming infrastructure for a successful ETL operation. An ETL programmer should not only see his or her single threaded set of programs. The architect must see the entire system of programs, He/she must ensure the technical team understands the target database design and its usage so that the transformations which convert the source data into the target data structures are clearly documented and understood.

The ETL process is much more than code written to move data. The ETL architect also serves as the central point for understanding the various technical standards that need to be developed if they don't already exist. These might include limits on file size when transmitting data over the company intranet, requirements for passing data through firewalls that exist between internal and external environments, data design standards, standards for usage of logical and physical design tools and configuration management of source code, executables and documentation.

The ETL architect must also ensure that the ETL design process is repeatable, documented and put under proper change control.

A key consideration for the ETL architect is to recognize the significant differences that the design and implementation methods for a business intelligence system have from an online transaction processing (OLTP) system approach.

The role of the ETL architect also extends to that of consultant to the programming effort. The architect works closely with the programmers to answer questions and plays a key role in problem resolution. Depending on the size of the programming effort and the project organization, the ETL architect may also supervise the development of the programming specifications. In any case, the ETL architect plays a key role as a reviewer and approver during the peer review process.

One last role for the ETL architect must be to ensure that the various software tools needed to perform the different types of data processing are properly selected ETL is one of the most important sets of processes for the sustenance and maintenance of Business Intelligence architecture and strategy.

III. PROBLEMS OF ETL

There are numerous problems to implementing efficient and reliable ETL processes. [10] Technical challenges moving, integrating, and transforming data from disparate environments Short load windows, long load times Inconsistent, difficult to maintain business rules Lack of exposure of business rules to end users Source systems missing certain critical data Poor query performance ETL functions reshape the relevant data from the source systems into useful information to be stored in the data warehouse. Without these functions, there would be no strategic information in the data warehouse. [4] If source data taken from various sources is not cleanse, extracted properly, transformed and integrated in the proper way, query process which is the backbone of the data warehouse could not happened.

IV. QUERY CACHE

For the fast access the database we use the query cache. Query cache will store all record of executed query. Query cache will keep record of newly executed queries. The major goal of the query cache is to reduce the response time of query. It will increase the brainpower of data ware house so that system will memorize the latest work it has performed. This memory will be used afterward for answer the result of queries which has been earlier performed by the users.

The cache will maintain two state valid and invalid state. When any query submitted by the user, the cache memory is first examined to check whether requested query is already store in the cache. If the query is stored, then check the state is valid or invalid. If state is valid then data can be access and if state is invalid then data can’t be access. but If user send a query of insert, update, delete and drop then data will be alter in database and state of related query will be invalid. Now invalid state data and query can’t be access by user. This can save important time and improve data warehouse performance by not reevaluating the queries which are already stored in the cache.

One of analyst place a query to show me the employee of a company, who working under the manager_id is 100,101,201. The query will look like as follows: -

SELECT *emp_id, name, salary, manager_id* FROM *employees* WHERE *manager_id IN (100, 101, 201);*

When the query is submitted, query cache will be examined to check whether this query is available or not and state is valid or invalid. If it is not available, query will be evaluated and result will be store in the query cache. The results of the query are shown in the table1

TABLE 1 – OUTPUT OF THE ABOVE QUERY

Emp_id	Name	Salary	Manager_id
202	Mukesh	6000	201

200	Mohan	4400	101
205	Sohan	12000	101
101	Rohit	17000	100
102	Sanjay	14000	100

If any other user submitted the same query the result will be retrieved from query cache because that query is already stored in the cache. We will call this Query1.
Let suppose another user wants to the employee of a company, whose salary greater than equal to 10000 **AND** manager_id is 100,101. The query will look like as follows:

SELECT *emp_id, name, salary, manager_id* FROM *employees*

WHERE *manager_id IN (100, 101) and salary>=10000;*

When the query is submitted, cache memory is examined. Same query is stored in the cache memory and state is valid then we can get the result of Query 2 as shown in Table 2.

TABLE 2 – OUTPUT OF THE ABOVE QUERY

Emp_id	**Name**	**Salary**	**Manager_id**
205	Sohan	12000	101
101	Rohit	17000	100
102	Sanjay	14000	100

Now result of Query 2 will be generated from the Query 1 result set instead of going through from all the data stored in the data warehouse. This process will save lot of time and effort required to go through all the records. The SELECT query, the cache first partitions the query to access all the partitioned tables. Then it checks whether each partitioned query is previously cached. If not, then it proceeds to see if any previously cached query can provide a partial answer.

If such a query exists and state is valid, then it sends a remainder query to the database. When the database returns the result for the remainder query, it merges the result with the partial result obtained from the cache. Finally, it merges the results from all the partitioned queries and returns the merged result back to the client.

UPDATE SHOPS
SET district = ZIP LOCATION MAPPING.district WHERE ISNULL (district) AND zip = ZIP LOCATION MAPPING.zip

When an update, or delete query is received, it is forwarded to the database. When the response from the database is received, the cache invalidates all cache entries dependent on either the affected tables or the affected columns. These references are then deleted from the database schema data structure.

A. Query cache state

Invalid –if query is not stored in query cache then state will be invalid. If data is updated by user by any these query insert, update delete the state will be invalid.
Valid – if query is stored in query cache and not updated in database from any these query insert, update delete the state will be valid.

Our problem is that we have a query and query result stored in the cache. But if the warehouse is updated with the new data the cache query result will reflect to old data. We will create a mechanism of state;

Query 1 is submitted by the user and his result is stored in the query cache. When next user submit the same query on updated data warehouse the query cache will check the state if state is invalid, it means the data warehouse is updated with new data. Now the query doesn't have to go through from all of the records. It will get the last index of the query result stored in the query cache. Then it will start searching the records which meet the query criteria from onward to that index. This can save lot of time and effort required to search the large amount of data.

B. Experiment

In the experiments, we consider a the client emulator, Apache and Tomcat are co-located on one machine. The query cache is running on a second machine and the MySQL database on a third machine. The same methodology is used for measuring the performance of both cache versions (the C-JDBC cache with and without the semantic optimizations). In each experiment the first half of the run is used to warm-up the cache and is excluded from the measurements. Each experiment also starts with an identical database. Differences between repeated runs of the same experiment were minimal. To select the load for the experiments, we are driving the server without the cache with increasing the number of clients, until performance peaks.

Then, we use the same number of clients to drive the server with the cache enabled, for both the basic C-JDBC cache and the semantically enhanced cache. We measured the performance in terms of throughput and response time of our cache versus the C-JDBC cache.

V. CONCLUSION

ETL functions needed to be carried out by a competent and trained ETL team. The ETL team is headed by an ETL expert. It's the responsibility of ETL architect to devise a comprehensive and effective ETL process to load the data warehouse.

The ETL architect/expert ensures that the ETL processes have strength and endurance. The ETL architect works in close coordination with the business users and identifies which data and at what level of detail is required. View materialization is a strategy used to provide fast answers to user queries.

In this paper, we have introduced a method to improve the performance and speed of ETL in data warehouse by minimizing the response time significantly. The primary goal of this technique is to store queries and their corresponding results. If similar query is submitted by any other user the result will be obtained using cache memory. Query Cache technique is to store queries and their corresponding results. If similar query is submitted by any other user the result will be obtained using cache memory. The information from previous queries is used to generate results. Finally, the results are efficiently merged for high performance. This active method greatly improves the performance and speed of ETL in Data ware house.

REFERENCES

[1]. R. Fagin, P. G. Kolaitis, and L. Popa. Data exchange: getting to the core. *ACM Trans. Database Syst.*, 30(1):174{210, 2005.

[2]. L. M. Haas, M. A. Hern¶andez, H. Ho, L. Popa, and M. Roth. Clio grows up: from research prototype to industrial tool. In *SIGMOD Conference*, pages 805 {810, 2005.

[3]. Fundamentals of database systems. 4 Edition. Persons international and Addison Wesley. Ramez Elmasri and Shamkant B. Navathe

[4]. Ideal Strategy to Improve Datawarehouse Performance by Fahad Sultan & Dr. Abdul Aziz. (IJCSE) International Journal on Computer Science and Engineering Vol.02, No. 02, 2010, 409-415

[5]. Efficient incremental view maintenance in data warehouses. Ki Yong Lee, Jin HyunSon, Myoung Ho Kim. Korea Advanced Institute of Science and Technology.

[6]. Strategy to make superior Data ware house by Vishal Gour in International Conference on advance computing and creating entrepreneurs Feb2010.

[7]. Shim J.; Scheuermann, P.; Vingralek, R.: Dynamic Caching of Query Results for Decision Support Systems, in: Proceedings of the 11th International Conference on Scientific and Statistical Database Management (SSDBM'99, Cleveland, Ohio, USA, July 28-30).

[8]. Nutt, W.; Sagiv, Y.; Shurin, S.: Deciding Equivalence among Aggregate Queries, in: 17th Symposium on Principles of Database Systems (PODS'98, Seattle,Washington, USA, June 1-3), 1998.

[9]. Building an ETL Tool by Ahimanikya Satapathy SOA/Business Integration, Sun Microsystems Data Warehousing Intermediate & Advanced Topics Common Problems, Uncommon Solutions by Eric Mellum