ISSN No. 0976-5697

Volume 8, No. 3, March – April 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Optimisation of ETL Process Using Partitioning and Parallelization Techniques

Kumari Deepika
M.Tech(Computer Science and Technology)
Central University of Punjab,Bhatinda,India

Abstract: The key asset of an organization is organized data in a proper format which helps the decision makers to take critical business decisions. Organizations have multiple business functions that are generating data and it is increasingly important for decision makers to analyze and act on these data. Extract-Transformation-Loading (ETL) is a process in data warehousing which converts the structure of data and enables decision makers as well as other applications to access it. As the volume of data is growing so quickly, ETL processes have to deal with a large amount of data and manage workloads through many different data flows, they consume a significant amount of time in order to move data from source to target system. However in most of the systems the effectiveness of the decisions matters on how quickly the decision has been made. In this work, we have analyzed the behaviour of various operations to extract, transform and load the data. Since ETL processes have to complete their execution within a specified time window in order to meet SLA given by the organization. In this paper, we delve into the optimization methods of ETL processes in order to minimize the execution time of ETL jobs. We identify the bottlenecks which cause the delay in execution of ETL process. We provide the techniques using partition, parallelization, and multi-threading which will help us to optimize the execution of ETL process.

Keywords: ETL; Dataflow Partitioning; Parallelization; Optimization.

I. INTRODUCTION

Data is scattered throughout the different systems within the organizations and proven useless because of not stored in centralized manner in proper schema and format. Extract, Transform, and Load (ETL) is the method used nowadays for transferring data from a source system to a data warehouse. It involves fetching data from many outside sources, transforming it to fit the operational needs (sometimes using intermediate tables), then loading it into the target Database (DB) or data warehouse. In Data extraction process, data is retrieved from distinct data sources for further processing or data storage. Data transformation activities are responsible to convert a set of data values from the data format of a source data system into the data format of a destination data system according to business logic given by client in the form of business requirement document (BRD).[1]

ETL process in data warehousing technologies performs the following tasks for pulling data out of the source systems and placing it into a data warehouse:

- extracting the data from source systems, data from different source systems is converted into one consolidated data warehouse format which is ready for transformation processing.

-Transforming the data may involve the following tasks

- Applying required business rules
- Cleaning columns
- Filtering rows
- Splitting column into multiple columns and vice versa
- Joining data together from multiple sources
- Transposing rows and columns
- Applying any kind of simple or complex data validation
- Loading the data into a data warehouse or data repository other reporting applications.

As shown in figure 1, in view of large data volume and to speed up the process a spate server can be used to perform ETL.

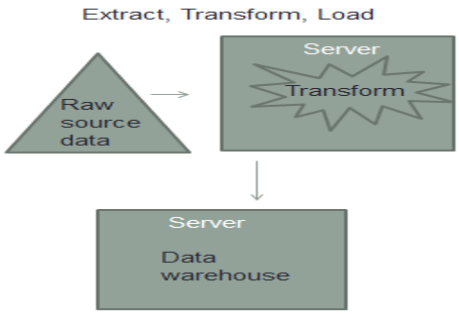


FIGURE 1: ETL PROCESS FLOW

The main objective of ETL process is to provide the right information to the right people at right time. ETL process has to be completed its execution within a specified-time period. Several optimization methods have been proposed to optimize the ETL performance. Partitioning and parallelization method is based on execution of multiple dataflow simultaneously proves useful in optimizing ETL performance by multithreading and optimal resource utilization. However this approach is limited to ETL dataflow itself. Partitioning and parallelization concepts can be extended to source and target side as well improve the ETL process performance. Pushdown logic i.e. part of ETL logic pushed to the database side can also produce the better performance.[10]

Nowadays, with ever-growing volumes of data, data warehousing is facing an enormous pressure to enhance its data processing capabilities for the evidence that ETL processes of many enterprises are taking hours or even days to complete its execution. Since data is not provided to decision makers in a timely manner so this delay might lead to an inaccurate business decision making. Enterprises are willing to

invest in enhancing storage and processing capabilities in order to improve the pace of ETL process execution. ETL developers therefore tend to improve ETL poor performance to make data warehousing systems efficient enough to cater the timely data needs of enterprises. The Performance is assessed either on DB level or ETL levels. The process of achieving good performance improvement in designing new ETL mapping and also fine tune the performance of existing ETL mapping ETL loads needs the performance assessments on all the ETL constructs levels i.e. source and target database levels as well as ETL processing levels.

1. DB LEVEL TUNING

To achieve good performance, it is first need to ensure that all the bottlenecks on the DB side are removed which ensures that the sources are in synchronized and hence full utilization of the source system is achieved. The details of how to overcome the DB side bottlenecks and its' effects are mentioned in table-1

TABLE 1.DB LEVEL TUNING

S.No.	*How?*	*Why?*	*Results*
1.	Check for the **INDEXES** on Key columns	Indexes on fields that use in **filter** and **join** operations improve the overall performance while fetching data from the table..	Indexes created on all Key Columns improve the performance of the **ETL** Load significantly.
2.	Prepare an **INDEX** analysis Document	**INDEX** analysis document keeps track of the various indexes on different key columns	Decision is taken on application and use of **INDEX** on needed columns is arrived.
3.	Use **'Explain Plan'** for all the Source Qualifier Queries(The EXPLAIN PLAN statement displays execution plans were chosen by the Oracle optimizer for SELECT, UPDATE, INSERT , and DELETE statements)	Explain Plan executes our query and records the "plan" that Oracle devises to execute our query	Provides a way of writing better Source Qualifier Queries and modifying existing ones.

2. INFORMATICA LEVEL TUNING

Once the DB level tunings are completed and now all the sources are fully tuned, we start looking at the Informatica level tuning. The details of how to overcome the Informatica side bottlenecks and it's effects are mentioned in table-2.

TABLE 2.INFORMATICA LEVEL TUNING

S.No.	*How?*	*Why?*	*Results*
1.	Removal of unwanted fields.	The Source Qualifier Query should have a select statement with only the fields that are required and which get loaded into the Target table.	This will bring down the running time of the SQL queries and this means the data will be fetched quickly from the database.
2.	Avoid constraints in WHERE Clause.	The usage of WHERE..IN..clause in the SQL Query should be avoided since it consumes more time in the query completion.	WHERE …EXISTS…clause which will results in better performance.
3.	LOOK UP & FILTER transformation	The Lookup and transformations combination always work very effectively in order to achieve good performance while handling huge data loads in the ETL process.	This result in faster load operations and hence the performance improves.

It is also required to establish a balance between these two levels by way of shifting the constraints according to the need and adopt a right method for testing the performance.

The paper is organized as follows. In section 2 describes literature review. In section 3 presents the experimental approach, and the optimizations applied to the data flows. In section 4 presents the implementation and the evaluation. In section 5 concludes the paper.

II. LITERATURE REVIEW

To optimize the ETL process various methods have been proposed. One such method focuses on the two way loading process[1] which works by putting source data into staging area and apply all ETL logic in the staging area and then load the data into the target. It lacks the importance of ODS (Operational Data Store). Another method focuses to be kept on rearrangement of logical transformation process [2] in order to improve performance and maintain data accuracy. It lacks the importance of logic change. In[3] author focuses to be kept on correcting design level flaws process, removing unnecessary components so that optimization would be achieved but it lacks dealing with huge data volume. In[4] author focuses to be kept on optimizing join and aggregate operation which are very costly while dealing with large volumes of data because of their blocking nature. In [3] author proposed method is not suited for the ETL workflows which deals with the huge data volume but not having complex data transformation requirements. In today's fast-changing, competitive environment, a complaint frequently heard by DW users is that access to time-critical data is too slow. DWs must be able to consistently generate accurate results. But achieving accuracy and speed with large, diverse sets of data can be challenging. They proposed various operations which can be used to optimize data manipulation and thus accelerate DW processes. They have introduced two such operations: 1. Join and 2. Aggregation – which will play an integral role during pre-processing as well as in manipulating and consolidating data in a DW. Changed Data Capture(CDC) approach demonstrate how we can save hours or even days, when processing large amounts of data for ETL, data warehousing, business intelligence (BI) and other mission critical applications. Typically, in a large organization, many distributed, heterogeneous data sources, applications, and processes have to be integrated to ensure delivery of the best information to the decision makers. Joins are used to combine information from two or more data sources, such as database tables, and place it into a new data source suitable for downstream processing or reports. Joins are particularly powerful because they enable rapidly changing data to be organized into categories for subsequent report preparation through the use of matching keys. Joins are used to pre-

process data, to improve the efficiency of queries, and to accelerate changed data capture CDC. CDC method uses MD5 algorithm to combine values of multiple columns into a single generated hash value and compares the hash values of columns from different sources in one pass to capture changes in data. If any change is found then update operation takes place. In [4] author has guided us to provide sorted data to aggregator and joiner in order to enhance their performance. In [5] author focuses to be kept on managing the metadata of ETL process in order to optimize ETL performance but since metadata is heart of an ETL process so developers can't be given access to metadata. He proposed ETL management through metadata management. A metadata management system with good design can highly improve the ETL efficiency. However, the large amount and the wide distribution of metadata in ETL processes cause the mismanagement of metadata. To deal with this problem, they suggest intensively manage ETL by metadata repository. Metadata repository is the data storage which can show DBAs the data about ETL objects in a simple, and centered way, and enables them to understand metadata more easily, therefore metadata management becomes simpler, and more centered. Compared with the common one, ETL processes based on metadata repository have a much better optimization effect. Lacking of effective management for the complex metadata leads to the inefficient ETL processes. To deal with this defect, it is suggested that DW designer can adopt Metadata Repository to manage metadata efficiently for the realization of a standardized and efficient ETL processes. All kinds of ETL modifications can be made conveniently in Metadata Repository such as modification of the source tables or the target data model both can be controlled by Metadata Repository. Once developers have Metadata Repository, they can see the relations between tables or attributes clearly, get a clear view to modify and maintain the ETL processes intensively, and realize the ETL processes optimization. Metadata repository stores execution plans , technical metadata, and business metadata. Business metadata should include dimensionality model and the dependency relationship between dimensional model and data source. Dimensional model mainly involves what kinds of dimensions, the differences among the dimensions, data cube and some rules in data mart. The key of metadata repository construction lies in technical metadata construction which needs the guide of material ETL process and business metadata repository. To Design a framework with reasonable structure and effective operation, and to create a metadata model for the unsystematic metadata through data extract, classifying and modeling, composing a metadata repository with distinct metadata views. Intensive management of metadata through the operation towards metadata repository, optimizing ETL. In [5], author has guided us to enable/disable constraints at the time of loading would help to optimize the performance of ETL process. In [6], author focuses to be kept on using relational algebra as a modeling language that is pushed logic towards the database in case of relational sources. He proposed the application of Relational Algebra as a modeling language of an ETL system as an effort to standardize operations and provide a basis for ETL execution platforms. As the volume of information to be processed periodically increases, an ETL task consumes more time and more resources. An ETL system is comprised of several tasks that gather data from source systems transforms it, and finally loading it into a DW. As already stated previously, the modeling phase of such a system is of its development, and therefore subject of intensive research. Crucial to the success, The application of the RA language as a modeling language of an ETL system in an effort to standardize operations and provide a basis for uncommon ETL execution platforms like grid environments. RA is an efficient tool for modeling ETL systems that uses patterns to model a real scenario of an ETL application. An ETL process is one of the most critical processes of a DWS, having considerable implementation difficulties that put in risk the success of the entire DWS. ETL modeling has gained great importance during the last years, giving excellent means to represent, discuss and analyze conceptually ETL tasks and control mechanisms, providing a "first view" of the entire system – a very useful instrument. So, it is important that we have available a standard notation to use at the ETL conceptual capable to represent the specificities of the process in a very precise way. In [6] author has guided us to use push down logic would improve performance of ETL process. In [7] author focuses to be kept on using ERP and SAP BI solution in order to improve ETL performance. It lacks of a generalized solution. In [8],[9] and [10] authors focus is to be kept on implementing partitioning and parallelization in order to optimize the performance of ETL process. In [8],[9] and [10] the idea of parallelization and partitioning has been kept limited to either enhancing the server capability or implementation within ETL dataflow. In[10] author proposed static partitioning and parallelization at transform level using multithreading and shared cache in ETL data flows. ETL dataflow are widely regarded as complex and expensive operations in terms of time and system resources. In order to minimize the time and the resources required by ETL dataflow, author proposed an optimization framework using partitioning and parallelization. The framework first partitions an ETL dataflow into multiple execution trees according to the characteristics of ETL constructs, then within an execution tree pipelined parallelism and shared cache are used to optimize the partitioned dataflow. Multi-threading is used in component-based optimization. Author proposed a partitioning framework that can partition an ETL dataflow vertically and horizontally according to the component's characteristics. Based on the partitioning, users may apply different optimization techniques to the resulting partitions with different levels of granularity. That is, in a dataflow partitioned by the vertical partitioning, a shared cache is used to transfer data between two connected components, thus, no copying is needed. The framework, then, does a horizontal partitioning on a subset of the vertically partitioned dataflow, and applies pipeline parallelization to process horizontal partitioned data splits. Finally, multi-threading is used to process the data within a component. They choose pipelining and multi-threading as the optimization is for the reason that they both are the effective optimization techniques adopted by ETL many tools. Therefore, using the proposed dataflow partitioning framework, as well as the optimization techniques, we could reduce I/O, and maximize CPU usage for an ETL program. Multiple pipelines instead of single pipeline process operations in parallel mode and optimize the ETL workflow execution. In[10],author lacks in dynamic partitioning. In[10] , author proposed the idea of parallelization and partitioning has been kept limited to either enhancing the server capability or implementation within ETL dataflow. They guided partitioning and parallelization at extract and load level as well

can make the process more efficient and better performance could be achieved.

III. METHODOLOGY

In this work, we are taking two data sets: employee data set and fact sales data set.

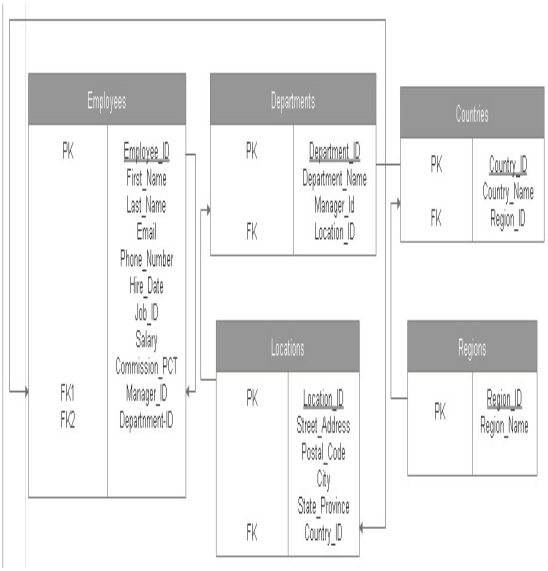
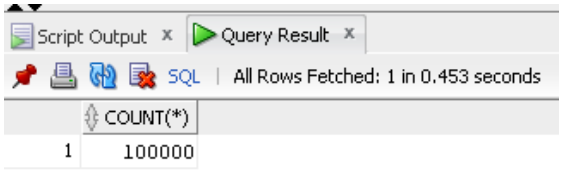


FIGURE 2: EMPLOYEE DATA SET



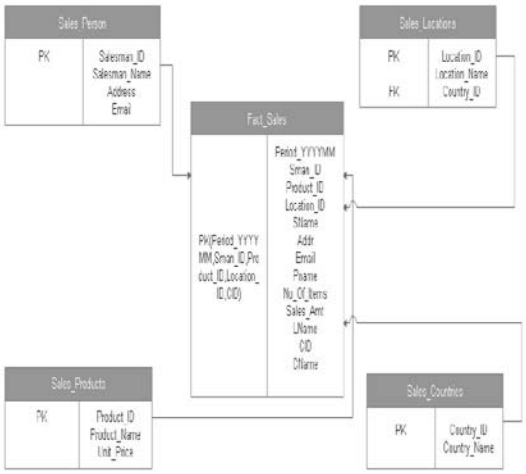


FIGURE 3:FACT SALES DATA SET

In this work, we will perform the activities of 6 cases as mentioned below:

1. w/o and with ETL source partition

An ETL job running with a single dataflow takes t time units to complete its execution. If multiple data flows are created instead of single data flow and data transformation task is handled in a parallel mode simultaneously in multiple flows then execution time will be reduced significantly by n times.

2.w/o and with pushdown logic in ETL mapping

Operations performed on the database side are faster than operations performed on ETL side. So pushdown logic will optimize the overall performance of ETL process. Operations on Database side are compared with the operations on ETL side on the basis of execution time.

3.w/o and with disable/enable Primary Key constraints

When load happens in a table integrity of data is to be checked because of constraints defined on a table and it makes the load operations slow. If constraints are disabled before loading these integrity checks won't be performed at the time of loading and load operations will be faster.

4.Join operation for unsorted and sorted data

More number of iterations is required to perform join operations if sorted data is not coming from the downstream sources and operations will be slower. If sorted data will be passed from downstream sources join operations will be faster.

5.Aggregate operation for unsorted and sorted data

To perform aggregate operations data is grouped on the basis of GROUP BY columns and if sorted data is not coming from downstream sources the full table scan is performed to group data. If sorted data will be passed from downstream sources only partial scan will be performed and aggregate operations will be faster.

6.w/o and with ETL target partition

If parallelization and partition techniques are applied on source and target systems as well then fetching and loading (Insert as well as Delete) operations of an ETL job will be tuned well and there is a significant performance gain.

IV. RESULTS AND DISCUSSION

TABLE 3.PERFORMANCE EVALUATION IN ETL TOOL

Performance Evaluation Chart		
	Performance Analysis (Time in Sec)	Data Analysis (Y/N)
For Employee Data Set and # records = 100000.		
w/o source Partition logic	35	Y
With source Partition logic	09	Y
w/o Pushdown logic	39	Y
With Pushdown logic	33	Y
w/o Target Partition Logic	748	Y
With Target Partition Logic	133	Y

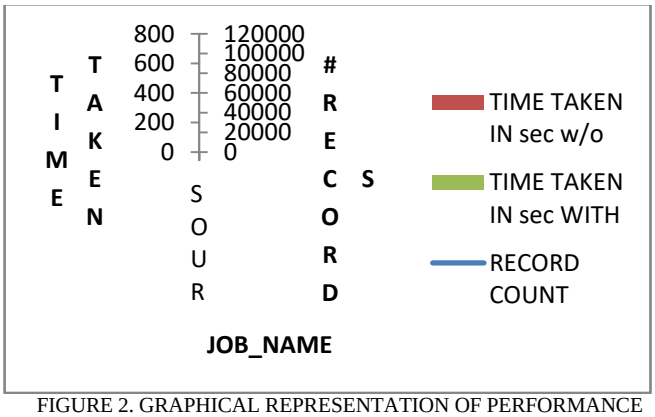


FIGURE 2. GRAPHICAL REPRESENTATION OF PERFORMANCE EVALUATION OF ETL JOBS

Analysis:

In [10], author takes into account the characteristics of ETL components, and based on the characteristics to do partitioning and optimizations. Using of the shared caches of transferring data that can't only reduce memory usage but also minimize I/O usage. Pipelining and multi threading both are used as well in order to optimize the performance of ETL flows. This approach has shown 4.7 times faster than usual approach. By analyzing this approach we have got the idea of implementation of partitioning at source as well as target side in order to improve the insert, update, and delete operation. In the approach of partitioning source table and implementing multiple ETL flows has shown 4 times faster than usual approach . In the approach of partitioning target table and implementing insert operation through bulk load and instead of deleting one row at a time , bulk delete operation is performed by dropping a particular partition operation has shown 5.6 times faster than usual approach.

In [6], author takes into account the application of the RA language as a modeling language of an ETL system in an effort to standardize operations and provide a basis for uncommon ETL execution platforms. Data transformations usually appear in any regular ETL system may be implemented using procedural representation for the description and validation of data based operations. By analyzing this approach we have got the idea of pushing ETL logic towards database. Some blocking and complex transformations consume a significant amount of memory and other resources, block the data and use separate space in memory to do their processing. These transformation logic can be pushed to the database side (either source or target), the optimizer of the database prepares the optimal plan and processes the logic so that execution time is reduced significantly. In the approach of implementation of push down logic has shown 1.2 times faster than the usual approach. If some query optimization technique is applied along with pushdown logic might produce the more optimal solution.

TABLE 4 .PERFORMANCE EVALUATION IN DB PROCEDURE

Approach	Disable/Enable Primary Key Constraint		Aggregate Operation on unsorted and sorted Operation		Join Operation on unsorted and sorted operation	
# records	Time taken w/o approach (in sec)	Time taken with approach(in sec)	Time taken for unsorted data(in sec)	Time taken for sorted data(in sec)	Time taken for unsorted data(in sec)	Time taken for sorted data (in sec)
1100000	64	26	9	3	67	52
2200000	124	54	17	9	142	120
3300000	167	104	25	12	209	180
4400000	290	110	35	17	284	254
5500000	301	119	45	24	349	307
6600000	387	162	54	29	401	378
7700000	463	240	68	31	578	465

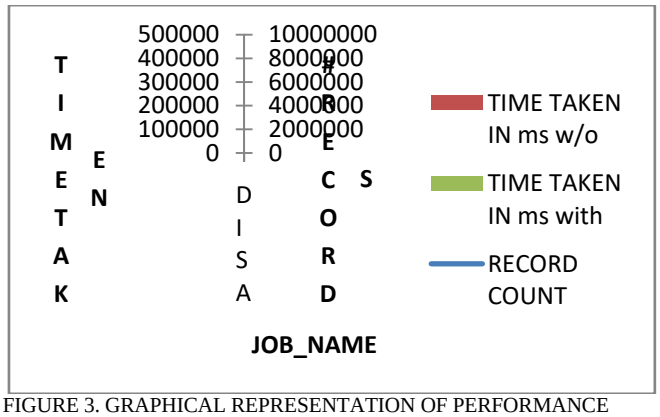


FIGURE 3. GRAPHICAL REPRESENTATION OF PERFORMANCE EVALUATION OF DISABLE/ENABLE PRIMARY_KEY CONSTRAINT

In [5], author takes into account the methods of managing the metadata in ETL process intensively by using the ETL metadata repository, properly analyzed metadata would speed up the pace of ETL. By analyzing this approach we have got an idea of disabling and enabling constraints by modifying the metadata of the target table before and after loading that table. We have performed the experiment on seven distinct set of records (1100000-7700000) and observed that the performance of these experiments have ranged from 1.6 to 2.6 times(2.25 times on average) faster than the usual approach.

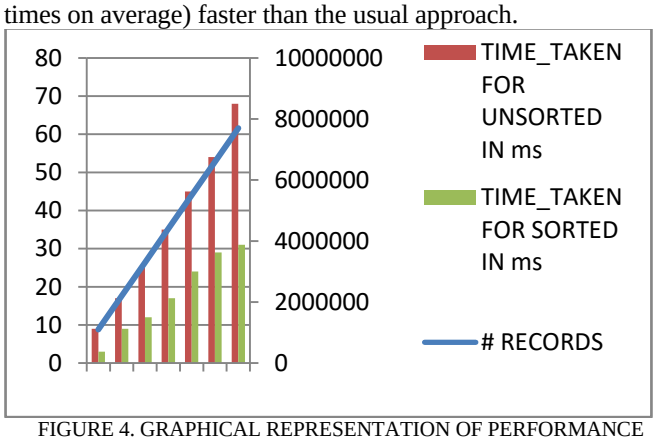


FIGURE 4. GRAPHICAL REPRESENTATION OF PERFORMANCE EVALUATION OF AGGREGATE OPERATION FOR UNSORTED AND SORTED DATA

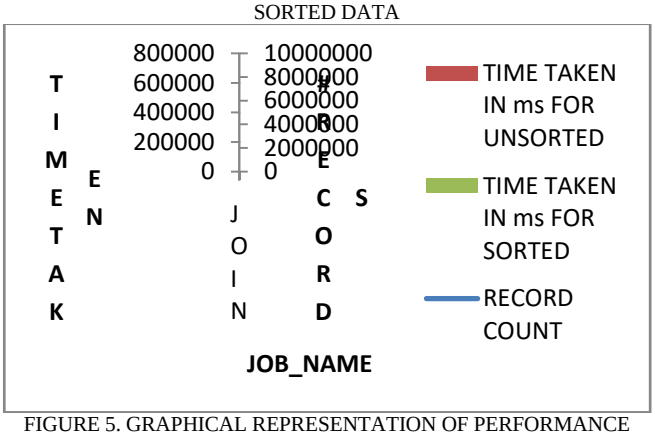


FIGURE 5. GRAPHICAL REPRESENTATION OF PERFORMANCE EVALUATION OF JOIN OPERATION FOR UNSORTED AND SORTED DATA

In [4], author takes into account that aggregation and join are very common operation in ETL processing of every industrial

sector where large volumes of data is processed. To optimize aggregate and join operation speeds up the pace of ETL process. By analyzing this approach we have got an idea of passing sorted data to aggregate and join operations. Sorted data will reduce the number of comparisons in joins and the waiting time of aggregate operation. We have performed the experiment of aggregate operation on seven distinct set of records (1100000-7700000) and observed that the performance of these experiments have ranged from 1.9 to 3 times (2.13 times on average) faster than the usual approach. We have again performed the experiment of join operation on seven distinct set of records (1100000-7700000) and observed that the performance of these experiments have ranged from 1.06 to 1.28 times (1.16 times on average) faster than the usual approach.

V. CONCLUSION

An ETL job running with a single data flow takes t time units to complete its execution. If multiple data flows are created instead of single data flow and data transformation tasks are handled in a parallel mode simultaneously in multiple flows then execution time will be reduced significantly. A huge volume of data has to be loaded into the target table and Data load rate is slow because of constraints on the target table. To disable constraints before load and to enable them after load can improve the ETL job performance significantly. In ETL scenarios, Join and aggregate operations cause degradation in performance. Passing sorted data to joiner and aggregator can improve the performance of join and aggregate operations. Since ETL applies the transformations step by step and can be time-consuming in some scenarios like joining many tables. Such logic can be pushed to database side to make the ETL job's execution faster. The reasons of performance degradation may be found in ETL transformation operations as well as fetching and loading operations. In some scenarios, ETL jobs have to perform Insert as well as delete and update operations. Instead of performing delete and update operation on row level, these operations can be performed on the whole set of data in one go by using table partition and temporary table concepts. Deleting data using drop a partition of a table is faster and it will produce a significant performance gain.

VI. ACKNOWLEDGMENT

I would like to thank Er. Surinder Singh Khurana (Assistant Professor) for continuously providing me support and guide me to do my work and also to the academic sources and environment provided to the students in the Central University of Punjab.

VII. REFERENCES

[1]. M.Kumar, N. A. (4-6 July 2013). Modeling and optimization of extraction-transformation-loading (ETL) processes in data warehouse: An overview. *Computing, Communications and Networking Technologies (ICCCNT),2013 Fourth International Conference on* (pp. 1 - 5). Tiruchengode: IEEE.

[2]. Vassiliadis, A. S., & Sellis, T. (5-8 April 2005). Optimizing ETL processes in data warehouses. *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on* (pp. 564 - 575). Athens, Greece : IEEE.

[3]. A. Simitsis, P. V., & Sellis, T. (Oct. 2005). State-space optimization of ETL workflows. *IEEE Transactions on Knowledge and Data Engineering (Volume:17 , Issue: 10)* (pp. 1404 - 1419). Greece: IEEE.

[4]. Tank, D. M., Ganatra, A., Kosta, Y. P., & Bhensdadia, C. K. (16-17 Oct. 2010). Speeding ETL Processing in Data Warehouses Using High-Performance Joins for Changed Data Capture (CDC). *Advances in Recent Technologies in Communication and Computing (ARTCom), 2010 International Conference on* (pp. 365 - 368). Kottayam: IEEE.

[5].cLi, L. (16-18 April 2010). A framework study of ETL processes optimization based on metadata repository. *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on (Volume:6)* (pp. V6-125 - V6-129). Chengdu: IEEE

[6]. Belo, V. S. (26-28 Oct. 2015). Using Relational Algebra on the Specification of Real World ETL Processes. *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on* (pp. 861 - 866). Liverpool: IEEE.

[7]. R. Wijaya, B. P. (27-29 May 2015). An overview and implementation of extraction-transformation-loading (ETL) process in data warehouse (Case study: Department of agriculture). *Information and Communication Technology (ICoICT), 2015 3rd International Conference on* (pp. 70 - 74). Nusa Dua: IEEE

[8]. Yingying Tu, C. G. (29-31 Oct. 2010). An intelligent ETL workflow framework based on data partition. *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on (Volume:3)* (pp. 358 - 363). Xiamen: IEEE.

[9]. Gupta, C., Wang, S., & U. Dayal, A. S. (1-6 March 2010). Partitioning real-time ETL workflows. *Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on* (pp. 159 - 162). Long Beach, CA: IEEE.

[10]. Iftikhar, X. L. (2015-04-13). An ETL optimization framework using partitioning and parallelization. *SAC '15 Proceedings of the 30th Annual ACM Symposium on Applied Computing* (pp. 1015-1022). USA: ACM New York, NY, USA ©2015.

[11]. J Anitha, K. S. (2012). Data Warehousing concepts using ETL process for Informatica Mapping Designer. *Advances in communication , navigation and computer networks.* hyderabad.

