Methods to Enhance Transformation in Near Real Time ETL

Mohammed Muddasir N.
Assistant Professor
Dept of IS&E
VVCE, Mysore

Ravi Kumar V.
Professor& HoD
Dept of IS&E
VVCE, Mysore

Prajwal V.
Student
Dept of IS&E
VVCE, Mysore

ABSTRACT
During the transformation phase of near real time ETL there could be some technique applied so that we get better results in terms of speed and accuracy. Transformation phase concentrates on changing the transactional data into semantically suitable format for the data warehouse. We try to bring in some of the solution during transformation phase that could enhance the speed and accuracy of the phase like advanced query optimization techniques, designing a new workflow so that we could reschedule some of the task. E.g. some functions applied on two parallel flows could be applied only once if the flows are converging. Also we look into some of the solutions for stream data how we could merge stream data and stored data, the challenges like speed and memory utilization. We also explore solutions like event based transformation for selected items, and handling of metadata efficiently so that it could add valued to the transformation phase.

Keywords
ETC,CBR, MDB

1. INTRODUCTION
Extract Transform and Load is the process of loading data from transactional database to data warehouse. This process was done in an offline mode in the non productive hours i.e. past mid night or on weekends. Offline mode of operations had the shortcoming of giving stale data for query results. But this has got advantages of not tampering with the production or transactional database and efficiency of transactional database would not be affected by offline ETL. Because of increased amount of data available on the net and in transactional databases businesses are looking for real time analytics. To do real time analytics data warehouses are required to be updated in real time or as soon as possible not waiting for the nightly or weekly scheduled update. This requirement leads to the concept of Near Real Time ETL [1]. There are several challenges to near real time ETL and research community has contributed effectively in overcoming some of the challenges. During transformation phase the extracted data are generated as streams from transaction database sources and data from disk read. Various solutions are proposed on this regard that processes the data in parallel. We try to identify the various solution and algorithms that address some of the issues related to ETL. Our focus in this survey is on the transformation phase of ETL.

2. RELATED WORK
Some of the challenges and possible solutions for near real time ETL are proposed in [2]. They have identified two problems each in extraction, transformation and loading phase. In extraction phase there are problems such as multiple heterogeneous data source integration solved using change data capture with stream processor and data integration tools. Second problem is data source overload solved using update significance and other record change methods. Also special format of CDC [3] logs are considered. In transformation phase they have identified master data overhead and need for immediate server for aggregate data. The solution for the first one is maintaining a cache of master data and database. Solution for the second problem is instead of transform and load we could actually load and transforms that could possibly reduce the time consumed for aggregation. During the loading phase the problems are performance degradation and OLAP internal inconsistencies for which solutions such as stage table, multiple stage table, staging outside data warehouse, snapshot data, real time data cache and layer based view have been proposed. Apart from this work not much work has been done on survey concerning the challenges of near real time ETL to the best of our knowledge.

3. METHODS TO ENHANCE TRANSFORMATION PHASE IN ETL
One of the important features of OLAP is to eliminate duplicates tuples by aggregating the tuples. In [4] the queries based on aggregate functions are enhanced using the semantics of the domain. They work on very basic aggregate functions provided by SQL that are based on equality of the tuples i.e. tuples get aggregated if they satisfy certain equality criteria. Example if employees belong to the same department they form a group [5]. In their work they have added further semantics based grouping of tuples that is based on similarity and not mostly on equality. In similarity bases duplication elimination they have two phase entity identification where tuples based on similarity are grouped and entity reconciliation where they grouped tuples are identified as a same real world entity. Tuples are similar if logic expression evaluated to a value above the set threshold. Various logical expressions are considered depending upon the domain. To the existing SQL group by clause they have added “ON” clause on which the similarity is evaluated.

Having a good query optimization technique is definitely an advantage at the transformation phase. Further it’s also important to design an efficient workflow. That is the focus in [6] they design a work flow based on state space tree. They identify that optimization is the task left for the DBMS to do and hence this could be improved. They take a case of loading data from two sources some of the data needs some transformation like converting dollars to euros etc. What they do is simply try to rework on the way things are executed first transformation then loading or first loading then transformation. They model each transformation task as an activity and process of ETL has several such activities. An activity can provide input to another activity or it could generate data into a file. The solution is to minimize the process or reschedule the process. As said earlier the process is modeled as a state space tree where nodes of the tree are activities or data stores and edges shows the flow of data from

one node to another. The ETL process is represented with a directed acyclic graph. Each of the activities has four components id, input schema, output schema and semantics of data flow in terms of relation algebra. They perform the transition in states by several ways like transforming execution sequence; common flow parallel jobs are transformed to a point where these parallel flows converge and by dividing the tasks of joint flow to clone. Swap transition are applied on unary operations like selection, primary key check, null value check etc. It's used to push certain activities at the end of the tree. Factorize and distribute are applied on binary operations e.g. union, join, difference etc. Two activities having same functionality but on different flow would be replaced after the flows have been converged. This allows certain activities are performed only once. Merge and split of two activities without changing the semantics to reduce the state space tree without losing on any design requirement.

In [7] the author has focused on the ETL workflow optimization which includes data segmentation, feedback, acquiring the required data, controlling the flow of the data etc. The system also includes an intelligent control module, which automatically judges the allocation of job. Data segmentation is nothing but dividing the big chunk of data in to smaller size which could be handled more easily and run these smaller jobs in parallel .While performing the data segmentation they consider the following aspects. 1) Amount of data: As the size of the data increases the implementation complexity also increases. 2) Disjoint sets: After the data segmentation each of the smaller sized sets should be disjoint in order to produce storage efficient and consistent results. 3) Criteria: The data segmentation should be simple and should have natural division standards to make the implementation easier. Data partition is done horizontally focusing on number of segments formed ,which follows all the above mentioned aspects and this process of segmentation is done is parallel fashion however parallelism is done only when it is suitable. Intelligent control module: The main functions of an ICM are monitoring the data acquisition, selecting an ETL strategy and adjusting the old strategy to the new strategy.

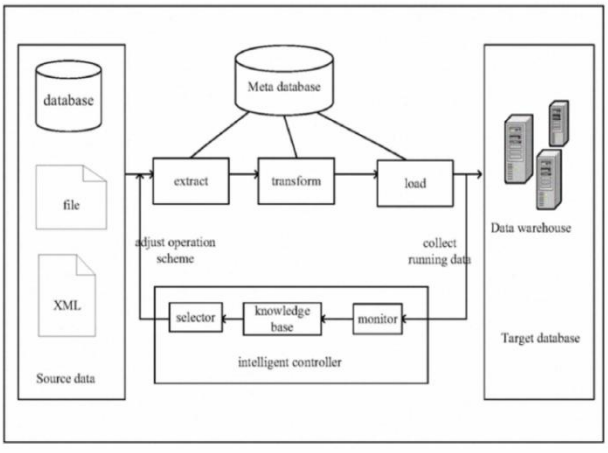

Fig1: Intelligent ETL tool architecture

It has a (i) Monitor, which will be used to acquire the data about the CPU utilization, I/O utilization and the cache.(ii) Knowledge base is a collection of interrelated information which is based on some representation .It also stores ,organize and manage these information.(iii) selector is used for the output of results as shown in fig1. Results are selected from the ETL strategy which has conditions like: 1) whether the job requires parallelism or not. 2) The reason for which the job cannot be a multi-machine parallel. 3) The job is multi-machine parallel and completes the task. The concept of the selector is based on the decision tree which helps in selecting the strategy. Decision trees are highly efficient and can be easily used to classify.

In [8] they have designed a novel approach for master data management dealing in transferring and transforming data in the ETL layer. Usually the ETL process happen in batch process and does offline operations so, there is a need for the data to be available immediately. In this paper the author has introduced an event-based near real-time.ETL layer for transferring and transforming data from the operational database to the data warehouse. This architecture uses the database queue and the push technology to achieve the content enrichment

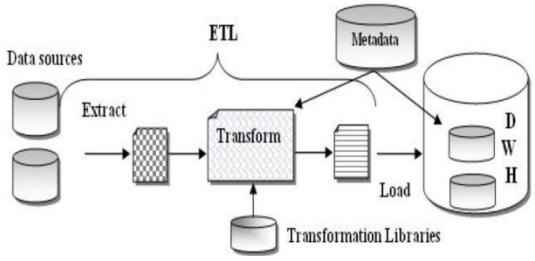

Fig2: Traditional data warehouse architecture

The traditional data warehouse as shown in fig2 uses the batch processing technique and the pull methodology and loads the data from the source to the data warehouse which will not provide the recent data to the decision makers. In the near-real time, the event processing time is faster and the data will be available soon. Content enrichment is nothing but adding some extra information to the current data. To achieve this data will be divided in to master data, which does not change and the transactional data, which will be updated often. Here, the master data referred by the transactional data should be available. In the batch processing method for each time the transaction data is loaded the corresponding master data will be extracted it is time consuming so this new architecture as shown in fig3 is proposed.

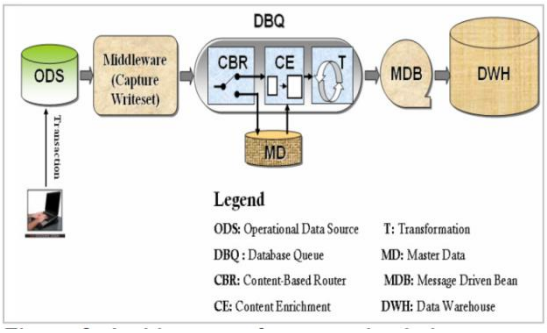

Fig3: Architecture of proposed solution

It has 3 main components: 1) Middleware, which is software used to capture the changed data in the form of a write set. 2) Database queue, it is a queue which stores the database tables. This will allow to process soon, many message driven bean can work on same queue and there is no need for writing separate code for create (), queue (), delete () etc. The content-based router (CBR) is used to distinguish the master and the transactional data on the basis of table and field names and sends it to the appropriate repository. 3) MDB, which checks

the DB queue for transformed updates if it is available it extracts and loads in to data warehouse. Content enrichment for the data is done using the INLJ (index nested loop join) on the master table. The lookup is created using the foreign keys of transactional data. By doing this it avoids the replication of master data for each transaction

In [9] the focus is on the speed of arrival of stream data and the I/O access speed of the disk. The streams arrive at a faster rate than the access speed of I/O on the disk. They represent input stream as S and look up table which is needed for transformation as R. only after the data from S is joined with R in the transformation phase it is loaded into the warehouse. One more parameter is the memory of the system having the lookup table. With the high arrival stream rate and limited memory the algorithm tries to achieve a high service rate keeping one of the parameters fixed. Either the streaming rate is fixed and memory utilized is considered for high service rate or memory is fixed and varying the arrival rate to maximize the amount of data that is transformed and stored in the warehouse. In their solution I/O cost of accessing disk table are shared among the tuples in the stream data. The main operation in transformation phase is join operation and they developed a solution known as MESH join that is faster compared to other joins. To understand MESH join the following tables are considered fig4 in these sources represents the arrival of stream data and lookup is the data on the disk. While performing join the two inputs to join are the source and lookup table these have different access speeds. Also there is a limited amount of memory to buffer and join the stream data and I/O data.

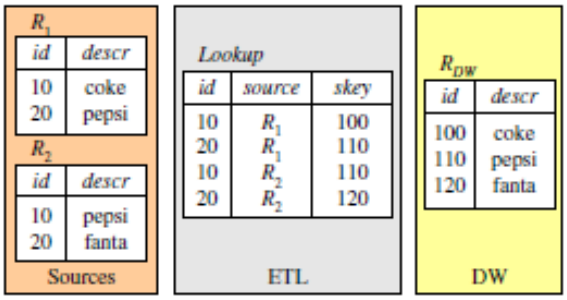

Fig4: Surrogate key generation

The mesh join operation is demonstrated in the diagram below fig5. At each time interval pages in the memory are joined with stream data and with new stream arrival old stream data are discarded as they have already being joined.

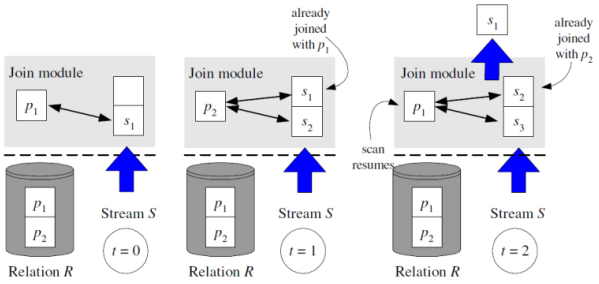

Fig5: Operations of Mesh Join

In [10] ETL includes the joining of the incoming updates and the metadata, the author has proposed an algorithm called Semi-Streaming Index Join (SSIJ) which improves the throughput of the join and is dynamic in nature i.e., it will manage both the memory and the incoming stream .Data may be of different types and come from different origins so, there is a need for cleansing, transforming and reconciliation before the data is saved in data warehouse. ETL process is done to update the data warehouse and it involves the job of joining the incoming updates and the data which is already present in the data warehouse. The join methods like conforming are expensive in terms of the operation done on the data because it involves refreshment of the data and tuples often and check for the redundancy etc. Where as in active data warehouses the data will be updated in real time so the joining of the updates with the data has to be done quickly and in an efficient way so there are joins like Nested-loop join, Mesh-join etc. but these joins have their own drawbacks. So the Semi-Streaming Index Join (SSIJ) as shown in fig6 is introduced to join the update with the metadata because it has the following characteristics: It exploits available memory resources to cache F.A.P (frequently accessed pages) of the relation and helps in faster join, accesses to pages are always batched to minimize the access costs, blocks of the relation that are not located in memory are read only, it dynamically adapts memory allocation to cope up with different characteristics of the streamed data, it supports equality, conditions and works well for arbitrary join relations to produce the exact join result, it is non blocking and produces a high rate output stream and it adapts its execution in order to meet processing deadlines for the incoming stream tuples.

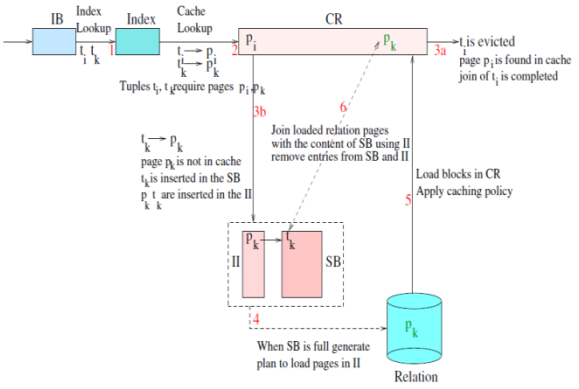

Fig6: Semi Streamed Index Join Overview

SSIJ uses B+ tree for indexing which helps in reducing the indexing overhead by decreasing the height of the tree by one level and makes best use of the memory. The algorithm performs a sort operation based on the indexes.

Algorithm: The memory of size M is partitioned in to 5 disjoint sets of variable sizes (will be dynamically adjusted by the algorithm) .It has 3 phases: Pending phase: Here, the algorithm waits for the minimum input buffer's unprocessed tuples to accumulate. Online phase: Here, the tuples are looked up using an index and joined with the appropriate relations. Join phase: This is the last phase where it reads the disk continuously and records will be brought to the cache. The tuples which are not joined with their relations in the previous phase will be searched and Joined using "matching".

Another work [11] develops a new method for partitioning the data they call it QoX-driven design framework that's based on certain quality metrics. Quality matrices are used for developing a tradeoff between the various competing objectives like performance, fault tolerance, reliability, recoverability, freshness etc. They try to optimize workflow design and execution of ETL with their partitioning approach.

The idea is to partition the flow keeping in mind flow throughput, memory, load on each partition, response time and synchronization. To do the partitioning they use two operators one is the router operator and other is the merger operator. Router operator performance the partitioning of the flow and merger operator does the merger. There are several aspects to be considered before the flow gets partitioned and also before the flow gets merged.

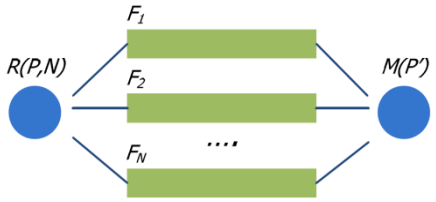

Fig7: Abstract partition of flow F into N flows

The function R is the router function which does the partitioning of the flow F in to several partitions f1, f2, f3,… fn and M is the merger function which combines the flow as shown in fig7. Ideally F after partition and merger should not change but its ok in this experiment because if we wanted to keep F unchanged it would be expensive and hence small delta is allowed keeping in mind however the semantics does not change. They complement the various partitioning policies available with commercial ETL tools like the round robin(RR), hash based (HB), and range(RG) with a novel partitioning policy called QoX-based (QB). If we consider RR then each operator i have the same processing cost cj for all the partition uniformly throughout leaving less scope for optimization. QB is based on the principal for taking the semantics on data in all phases of ETL e.g. prioritizing data based on SLA, origin of data etc so the partitioning varies and we could optimize the processing of data.

In [12] they deal with challenge that is related to the mismanagement of metadata. It uses ETL on metadata repository to overcome the above challenge. Metadata is nothing but the information that is related to the data, which helps the data warehouse users in locating and using it easily. ETL process has 3 parts namely Extraction phase, where the data from the source is read. In transformation phase, the above read data will undergo some changes depending on some rules and deals with problems related to the data such as redundancy, ambiguity, incompleteness etc. Loading phase, where the above transformed data will be loaded in to the data warehouse. The problem is that the general ETL processes will not store this metadata in a proper way often it will be undermined and stored using excel and other simple tool which is ineffective on the larger data sets. To make this effective the metadata will be stored in a centralized server, which is also a metadata repository and is independent of other tools. Metadata repository will have the unambiguous and the correct data which can be directly used by the data warehouse operators to join the tables, modifying the tables etc. Metadata repository consists of business metadata and technical metadata. Business metadata has dimensionality model and the dependency relationship between dimensionality model and data source. Technical metadata should include this kind of metadata that can be applied in developing, administrating and maintaining data warehouse.

Framework:

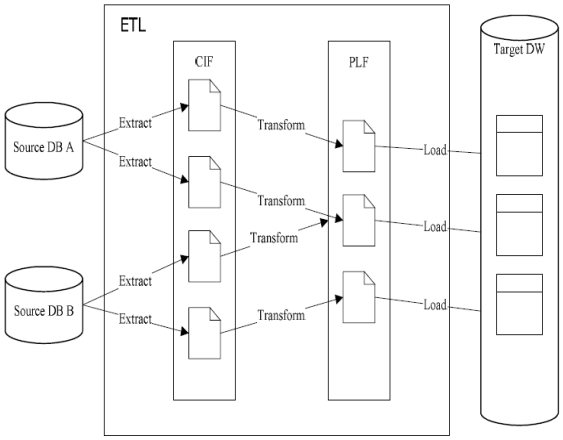

Fig8: ETL logical framework

Flat files are chosen as interface for each process, flat files contain the data records with no structured relationship. After the extraction it will create CIF files, these are then transformed in to PLF files, finally it is loaded in to data warehouse.CIF files can be obtained after the data cleansing which helps in maintaining the source data in the same format and eliminate the differences between source files. PLF are the pre-loaded files which are the obtained after transforming one or more CIF files. Choosing the flat files will give the advantage of running the ETL process faster as shown in fig8.

4. CONCLUSION AND FUTURE WORK

Throughout this paper the transformation phase of ETL was studied and methods required to improve this phase were discussed. We could say that depending on the type of data various different enhancements could be applied to transformation phase. Example if the data are only from stored sources we could enhance transformation with query optimization techniques. If the data are from streams and stored sources parallel flow algorithms, new work flow design, and joins based on some part of stream data and some part of stored data could be considered. Further our focus will be on the extraction and loading phase. We would like to study methods to enhance extraction and loading phase. Also entire ETL optimization based on some hypothetical data and considering different cases would be our next work.

5. REFERENCES

[1] P. Vassiliadis and A. Simitsis, "Near Real Time ETL," springer, vol. 3, 2008.

[2] A. Wibowo, "Problems and Available Solutions On The Stage of Extract, Transform, and Loading In Near Real-Time Data Warehousing," IEEE, p. 345, 2015.

[3] C. K. Bhensdadia, D. M. Tank, A. Ganatra and Y. P. Kosta, "Speeding ETL Processing in Data Warehouses Using High-Performance Joins For Changed Data Capture (CDC)," IEEE, 2010.

[4] E. Schallehn, K.-U. Sattler and G. Saake, "Advanced Grouping and Aggregation," in CIKM '01 Proceedings of the tenth international conference on Information and knowledge management, New York, 2001.

[5] R. Elmasri and S. Navathe, Fundamentals of Database Systems, Addison-Wesley Pubs, 2000.

[6] A. Simitsis, P. Vassiliadis and T. Sellis, "State-Space Optimization of ETL Workflows," IEEE

TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, vol. 17, no. 10, 2005.

[7] Y. Tu and C. Guo, "An Intelligent ETL Workflow Framework based on data Partition," IEEE, 2010.

[8] M. A. Naeem, G. Dobbie and G. Weber, "An Event-Based Near Real-Time Data Integration Architecture," IEEE, 2008. [9] N. Polyzotis, S. Skiadopoulos and P. Vassiliadis, "Supporting Streaming Updates in an Active Data Warehouse," IEEE, 2007.

[9] M. A. Bornea, A. Deligiannakis, Y. Kotidis and V. Vassalos, "Semi-Streamed Index Join for Near-Real Time Execution of ETL Transformations," IEEE, 2011.

[10] A. Simitsis, C. Gupta, S. Wang and U. Dayal, "Partitioning Real-Time ETL Workflows," IEEE, 2010.

[11] L. Li, "A Framework Study of ETL Processes Optimization Based on Metadata Repository," IEEE, 2010.