Journal of Theoretical and Applied Information Technology
30th November 2017. Vol.95. No 22
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

THE CHALLENGES OF EXTRACT, TRANSFORM AND LOAD (ETL) FOR DATA INTEGRATION IN NEAR REAL-TIME ENVIRONMENT

ADILAH SABTU[*1,2], NURULHUDA FIRDAUS MOHD AZMI[1,2], NILAM NUR AMIR SJARIF[1,2], SAIFUL ADLI ISMAIL[1,2], OTHMAN MOHD YUSOP[1], HASLINA SARKAN[1], SURIAYATI CHUPRAT[1]

[1]Advanced Informatics School (UTM AIS), Universiti Teknologi Malaysia (UTM), Malaysia

[2]Machine Learning for Data Science Interest Group (MLDS), Universiti Teknologi Malaysia (UTM), Malaysia

E-mail: {adilah.sabtu@gmail.com, |huda, nilamnur, saifuladli, haslinams, suriayati.kl|@utm.my}

ABSTRACT

Organization with considerable investment into data warehousing, the influx of various data types and forms require certain ways of prepping data and staging platform that support fast, efficient and volatile data to reach its targeted audiences or users of different business needs. Extract, Transform and Load (ETL) system proved to be a choice standard for managing and sustaining the movement and transactional process of the valued big data assets. However, traditional ETL system can no longer accommodate and effectively handle streaming or near real-time data and stimulating environment which demands high availability, low latency and horizontal scalability features for functionality. This paper identifies the challenges of implementing ETL system for streaming or near real-time data which needs to evolve and streamline itself with the different requirements. Current efforts and solution approaches to address the challenges are presented. The classification of ETL system challenges are prepared based on near real-time environment features and ETL stages to encourage different perspectives for future research.

Keywords: *ETL, Near Real-Time Environment, High Availability, Low Latency, Horizontal Scalability*

1. INTRODUCTION

In the spectrum of big data, per Forrester Research Q1 2016 report, data preparation and data integration technologies are within the gamut of intense survival and growth phases respectively. Ultimately, the goal is to make data flow fast, fresh and seamless across multiple sources in a data warehouse and provide sound bases to accommodate business intelligence or analytics.

Less glamorous than data analytics yet an enabler with equally important stake, data integration involves vetting data across many databases with different business demands to produce a centralized master data management system among other operational systems. Normally, data are integrated before operational and transactional activities. The effectiveness of data integration is basically vindicated by how well the data deliver and realize their intention of use, which in most cases are for analytics. Data warehouse commonly engages extract, transform and load (ETL) system for maneuvering the operational structuring and transaction of the data. Traditional or conventional ETL is comprised of extract, transform and load stages where the steps involved loosely follow the order of the abbreviated term and there is no standard modeling.

A typical ETL system may prove to be effective in managing structured, batch-oriented data which is up to date and within scope for corporate insights and decision making. However, dealing with faster stream, time sensitive or sensor data requires different model and rather massive tweaking to the ETL system [1] where high availability, low latency and horizontal scalability are three key features that need to be addressed in near real-time environment [2]. In this environment, data need to be available fresh, pronto and continuously.

This paper aimed to expand existing research and classify the challenges and measures in developing ETL system for near real-time or streaming data to provide some guidance and focus for research and backroom developers of data analytics. The structure of the paper is as follows: Section 2 provided an overview of the ETL system. Later, Section 3 discussed three (3) key features of a system for near real-time environment, which are high availability, low latency and horizontal scalability. Next, the related works on ETL systems implemented for near real-time data warehousing. is discussed in Section 4. Section 5 discussed about the challenges for ETL in near real-time or streaming data environment based on these features.

2. ETL SYSTEM OVERVIEW

ETL System is a broad presentation of data movement and transactional processes of extracting data from multiple application sources and different communication protocols [3] which are normally very time-consuming to design for efficiency and performance, transforming data into dimensional fact table and conformed format and loading and feeding data efficiently and consistently into data warehouse environment such as data marts, data stores or other target systems [1][4][5]. The process is widely applied in data integration, migration, staging and master data management efforts, also termed as data staging area [3]. Figure 1 illustrates a basic ETL system flow in a data warehouse environment. The flow involved accessing data from source locations, then, ETL processing which included cleaning, integrating and staging the data before transporting and channeling the transformed data to target systems.

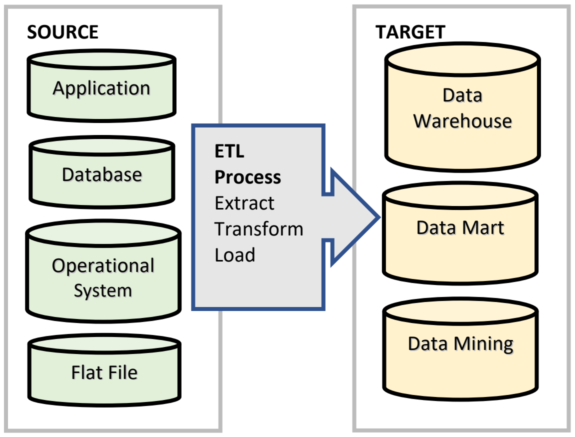

Figure 1: ETL System Flow in a Data Warehouse Environment

Like any living sentinels which in time experience change and evolvement, a living IT system also needs to undergo evolution to ensure its relevance and continuity in the face of environment change. Traditionally, it was already advantageous to streamline and enhance ETL process through efficient automation [6], new data management approaches [7], effective ETL modeling and powerful tools such as Informatica, Oracle Warehouse Builder and Microsoft SQL Server Integration [5]. Nowadays, the management and maintenance part of ETL system further need to have the capacity to address change of directions and to support new big data demand which are becoming more huge volume, endless streaming, wider variety of types and sources, request for real-time user requirements and new analytics, extreme integration, availability of new technologies and much more powerful tools [8].

3. KEY FEATURES OF A SYSTEM FOR NEAR REAL-TIME ENVIRONMENT

System for near real-time environment is mainly featured by its high availability of streaming data [2] described as very ferocious fast, high volume with a propensity to overload [9], low latency of intervals between transactions [2], like some internet of things (IoT) applications which require very low latency support [10]; and horizontal scalability for performance improvement [2] and support at global scale [10]. The features are shown in Figure 2. Failing to address these features would affect or limit functionality of an operational system.

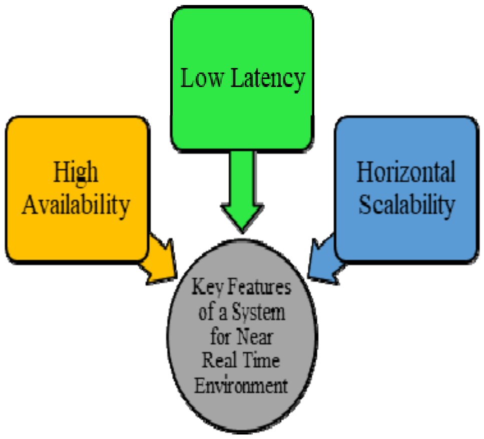

Figure 2: Key Features of a System for Near Real-Time Environment

4. RELATED WORKS ON ETL IN NEAR-REAL TIME ENVIRONMENT

Existing research classified the challenges and solutions in different ways such as ETL stages [11], real-time modeling, real-time mechanism, real-time ETL enabler, OLAP issues [12] and data stream management system [13][14]. This paper explored the related works based on a system three (3) key features in such environment, namely high availability, low latency and horizontal scalability.

4.1 High Availability

Streaming data as the name dictates are always available in endless constant flow within mere seconds. They are sensitive by nature that the slightest disruption would affect operations.

Distribution and replication are important considerations to ensure fluidity of data collection process, loss data can be recovered and always available when called upon, even when pitted against outage, mitigate data loss or receiving overloads of throughput [2]. Unlike batch data which are distributed periodically, change of fresh data is less frequent and the same batch data can be used repeatedly.

Handling heterogeneous data sources might require separate kinds of processors and configurations to ensure that fresh data and the necessary backup are always at hand, such as streaming processor [15] for streaming data and change data capture (CDC) for stored data [11]. Setting up separate server for backup and replication or CDC log-based technique [16] ensured availability that the risk of data loss is mitigated [2].

Other solution applied semantic web technologies toolkits or programs that support real-time streaming in ETL system to leverage connections between disparate data sources [3] [17]. Tested in an automotive environment, the inference methods, iterative process and expanded ontology federated process used over the technologies helped with the identification of relationship among the heterogeneous data sources and were expected to support data integration initiatives [18].

OLAP allowed different kinds of multidimensional analytics. Having no mechanism from preventing analysis and update happening concurrently, the result might not be accurate. A dynamically generated layer-view combined with lock row where new layer is automatically generated for every new transactional occurrence or new analysis helped to ensure that the latest data is provided [12]. Other plausible solutions are staging data for data analysis separate from the update session of the data warehouse, taking snapshots of the data warehouse table or using Real-Time Data Cache (RTDC) [14][11]. Fang et al. [19] used Real-Time Operational Database (RODB) in the middleware layer to effectively manage massive sensor data and devices and storage. Dynamic mirror technology solution was proposed by Li and Mao [20] to manage OLAP queries and Online Transaction Processing (OLTP) updates from blocking each other functionality.

4.2 Low Latency

The requirement for speed in delivering fresh data to meet business needs is called data latency. Compared to periodical requirement of batch data, the time between the events where data arrive or are conceived and data made available to user for near real-time data is almost instant and low latency [2], [13].

Traditional ETL is designed to accommodate batch processing and deploying efficient bulk-loading techniques [13] where data refreshment can be done during off peak hours or offline, of which operational activities can be temporarily suspended [11]. Therefore, it could not produce accurate result for near real-time analysis where stream data flow continuously.

In their research, [13] introduced an extension technique called Semi-Streaming Index Join (SSIJ) which is an index-based algorithm to join streaming relation with stored data disk-resident relation of an active data warehouse. The analysis demonstrated a sustainable throughput of the join for these data and produced high rate output stream.

Multiple data source integration could also consider separate kinds of processors and configurations to ensure streaming fresh data such as streaming processor and change data capture (CDC) integrated using data integration tools [16] [11]. CDC techniques can adequately contain data source overload using special format log [21] although there are still latency between the original and captured data [22]. Problem when using CDC Trigger technique such as master data overhead was overcome by maintaining separate queues, placing master data in a cache and real-time data in a database queue [16][11].

Narendra et.al [23] proposed a form of ELT approach called data warehouse-based data model for analytics which need a fraction of streaming data only at a time. This approach filtered data required for analysis and extracted them into a staging database of server for aggregation. Then, the data were loaded into the warehouse for transformation. This solution overcome the problem of reduced speed if a large amount of data was processed and stored.

Jain et.al [16] compared many CDC techniques which support real-time data. Log-based technique was considered a better technique based on the minimal impact on source database. Its log files retention minimized data loss, reduced performance degradation and sped up analysis process. In addition, two (2) critical data identifying measures, update significance and record changed methods were used to prevent data source overload. Further study by Jain et.al [16] suggested staging temporary tables following an exact replica of the data warehouse destination tables for receiving and storing new data. Other study by Valencio et.al [24] used a technique termed as Real Time Delta Extraction based on Triggers and Multi-stage Trickle and flip technique by Zuters [25] basically adapted and combined conventional and real-time ETL techniques aimed for zero latency when processing constant small data loads.

Tiwari's research on Advanced ETL (AETL) [6] through integrated PERL subroutine and scripting method proposed a technique of segmenting information by time to reduce time lag caused during processing and enhanced data loading. The research avoided data duplication and quickened its enquiry operations.

4.3 Horizontal Scalability

Horizontal scaling is a sustainable approach for improving performance and mitigating the risk of interruption of seamless data flow [2]. It adds separate independent servers to a cluster of servers handling different tasks and requires only low coordination between systems. Arora et.al [10] mentioned that modern day data intensive and variety processing needs required multitude of data processing engines and benefitted in improving performance by isolating the workloads.

This type of scaling offers higher potential of expansion and less risky measures compared to vertical scaling where boosting its performance is limited within the capacity of a single machine or workflow. If the machine flopped, data flow is interrupted and the system could lose pertinent data. However, this does not justify crossing out vertical scaling from suitability for near real-time environment; merely an implication of risk and compromised performance.

In their research, [10] also discussed the disadvantages of multi-engine architecture which could cause bottleneck in the ETL pipelines because of frequent transferring of data and excessive workloads, if not continuously maintained could end up with inaccurate analytic results due to time lag between data ingestion and real-time results. Hence, the research introduced a multi-representation based data processing architecture which store data in different representation, yet perform various updates in a unified manner and configurable to aid specialized analytic operations.

Narendra et.al [23] showed the effectiveness of scalability in a system architecture built on different modules, servers and process processors to perform each own separate goals, processes, functions and scheduling when feeding the many requirements of near real-time data. Another solution successfully staged OLAP outside the data warehouse update period [19][11].

Another research [26] introduced CR-OLAP a cloud-based real-time OLAP system utilized cloud infrastructure consisting multi-core processors which increased database size and maintain performance.

A comparison of near real-time warehouse technologies for Twitter between Apache Cassandra and NOSQL databases conducted by Murazza and Nurwidyantoro [27] showed how the distribution system of Cassandra, built on a set of database nodes which separated write and read operations on different nodes with more scalable, fault tolerance and flexible modeling performed better at handling Twitter streams than a column oriented NOSQL database.

A research by Mehmood and Naeem [9] demonstrated how a parallelized algorithm called P-CACHEJOIN improved its service rate significantly over a sequential execution of

CACHEJOIN algorithm by exploiting its memory and CPU resources optimally.

5. CHALLENGES FOR ETL IN NEAR REAL-TIME ENVIRONMENT

The environment for near real-time data lean more on the extreme side; incessant flow of fresh data in voluminous amount and real-time reporting, with minimal impact on source data and at the same time realizing cost-effectiveness in its implementation [16] despite the additional workload and trade-off.

Although there is a lot of powerful ETL tools introduced in the industry to overcome ETL process shortcomings [5], traditional ETL system still need major works and remodeling to support the requirement. Mesiti et.al [28] in their research made a more complex attempt involving artificial intelligence techniques, which in their research, the design of ETL operations encompassed hardware and software components, having active human links that made handling of stream data easier, relevant data were accurately captured with reduced data flow and processing delay. Similarly, Tiwari [6] described automated-intelligence based ETL tools programmed and prepared to manage data volumes access, implicitly backed by strong client requirement interface, enhanced data handling quality and having logging insights and data blunder foresights.

5.1 Extraction

The process of extraction involves profiling data dimensional attributes from separate databases, capturing change data and getting data from the source locations. Streaming data such as network traffics, click streams and sensors are fleeting, constantly changing and continuous. The frequency of data change is extreme and the volume of fresh data is colossal too.

It challenges the way change data are captured for constant update and loaded without becoming overloaded and disrupting normal operational activities [16][11], research in [7] concluded that existing ETL was lacking at covering context and semantics when processing Entity Resolution (ER) for stream data, and there was the task to handle multi-joins relation that produced high rate output streams to meet processing deadline [13]. It also compels remodeling ETL to become source independent for more efficiency, such as including user defined transformation in the modeling approach [5].

Extraction stage manages multiple as well as heterogenous data sources, and not exclusively for near real-time data source. Stream data change constantly while static data are less volatile hence can be used repeatedly. These are some methods extended in ETL, researched and discussed in previous section for effective extraction process of the different types of data and resources:

- Implementation of stream processor for near real-time data and different method for stored data, such as Change Data Capture (CDC)
- Semantic measures to identify and manage different types of data sources
- Index-based join relation algorithm for joining stored and stream data
- Entity Resolution for stream data
- Artificial Intelligence and Machine Learning
- Parallelized algorithm for exploiting memory and CPU resources
- Multi-representation based on data processing architecture
- Multiple data source integration by combining Change Data Capture (CDC), stream processing and data integration tools

Change Data Capture (CDC) Log-based technique is a mechanism applied in near real-time environment which identifies and extracts changes from source data while having minimal impact on its source database [16]. Despite that, the constant reading of continuous streams of data whilst performing other transactions will create data source overload. To overcome them, extensions of CDC techniques have been considered, one such was implementing technique that filtered critical data based on its update significance and number of records change which set priority criteria for the extraction and prevented overload. Nowadays, more ETL tools with CDC and other techniques are developed and available for near real-time data warehouse.

In the event of shutdown, options of implementing reliable backup and recovery [8] for the volatile stream data are needed. There are also issues concerning latency and mismatch between actual transactions and the changes captured. Extensions of log-based CDC mechanisms are widely applied as measures, besides accommodating servers for replication or using separate processor and configurations with ETL integration tools.

For best practice and to avoid further degradation in performance, design of the extraction process need to consider system architecture based on different models, allocation of servers to perform separate goals, distributed system modeling, time segmentation and remodeling source independence.

5.2 Transformation

Transformation is the process where data are cleaned and conformed into fact table or predetermined format which can be shared across different business platforms and needs. The challenges posed in this process are namely on managing master data overhead and minimizing movement of data within the environment.

For near real-time data, transactional data are constantly refreshed so the frequency of having to update master data too is higher than batch data. However, the master data which are set in dimensional table should not be compromised when joined with transactional data fact table [11]. One technique is to separately place master data in a cache while near real-time transactional data are placed in a database queue, both implemented outside the data warehouse. This enabled the joining of master data and transactional data to become more manageable during refreshment period and achieve near real-time capabilities. To obtain optimum performance or faster ETL process, other techniques might be implemented together. For instance, Change Data Capture (CDC) log-based technique can be used for maintaining log files detailing the transactions in a database at minimal impact on the source database [16].

Another challenge in this stage is that the amount of data carried into the warehouse after transformation is smaller and also constant that made the transformation process quite inefficient processing smaller amount at more frequent rate [11] and later having to refresh them incessantly. Recent development considered Extract, Load and Transform (ELT) methods beside ETL as fitting practice for streaming data where business required more instantaneous response time. In this case, the technique is analogous to an express check-out counter of a large grocery store. The counter is setup specifically to speed up transactions for small unit purchases. For ELT technique, the frequent flow of small chunks of unprocessed data are extracted, normally through specific applied filters and server aggregation [23], then loaded faster into the data warehouse which has staging area or table setup. These raw data are ready for transformation on demand [21]. The ELT pipeline for this implementation could be separated from other ETL processes with less pressing real-time requirement.

5.3 Loading

In loading stage, the transformed data or metadata are delivered into the data warehouse target dimensional models or tables which are accessed by business users and applications. The challenges are to maintain optimum performance during OLAP or online analytical processing analysis when overlap occurred while loading [11][25] and to mitigate impact from OLAP internal inconsistency [12][29].

If tasks overlapped, they would cause performance to degrade. Furthermore, the query results would be inaccurate due to time lag [6][10]. For solutions, trickle and flip techniques and variations are applied [25]. Basically, staging tables which duplicated the target dimensional models or tables are temporarily setup for loading, to receive and store data in real-time. Periodically, the target dimensional models perform updates from these staging tables. Hence, enabling ETL and analytics tasks to be performed separately with more efficiency, accuracy and preventing from the degradation while maintaining ETL process off limit to the users.

OLAP is originally designed to cater spreadsheet environment and financial applications which used historical data with proprietary format. There is no specific operant to manage the continual data change while OLAP is in progress which again would affect result accuracy. To safeguard the integrity of data in the warehouse, OLAP is performed in temporary staging areas, replicated tables or snapshots of target dimensional tables, or applying an external cache, all of which separated OLAP tasks from the frequent update tasks [24] [25] .

For the external cache method, RTDC (Real-Time Data Cache) is applied. Real-time data are stored in the external cache and used to perform OLAP analytics. The results are derived and temporarily stored in the cache. During update period, these data are refreshed and stored into the data warehouse. The approach is quite a contrast from the staging tables and snapshots methods.

On the other hand, a layer-based view method will generate new layer-view from the data warehouse tables whenever there are data change so that latest data are made available for OLAP [12]. This method still leaves the data warehouse tables separate from OLAP process.

The identified ETL challenges and solution approaches are summarized and presented in Table 1, placed after the list of References. Further details of these solutions with the existing studies related to the identified key features of near real-time environment were discussed in previous Section.

6. CONCLUSION

Data integration of Extract, Transform, and Load (ETL) technologies have been used to accomplish this in traditional data warehouse environments. However, as we enter the Brontobyte Age, traditional approaches of data integration such as ETL begin to show their limits. Thus, traditional ETL needs to be enhanced to support streaming of data processes with federation across multiple sources. The role of ETL needs to be evolving to handle newer data management environments. Traditional or conventional ETL system is facing challenges in keeping up with the evolving requirements of big data in data warehousing.

This paper reviewed existing literatures and classified the challenges in ETL system for streaming or near real-time data based on the key features of a system purposely for these environments which are high availability, low latency and horizontal scalability. The paper offered a different angle for other future research to deliberate, verisimilarly, understanding the system features for the environments would facilitate better planning of near real-time projects in a preemptive way.

ACKNOWLEDGEMENT

The authors would like to thank Advanced Informatics School (AIS), Universiti Teknologi Malaysia (UTM) for the support of the resources. This work is currently funded by GUP Tier 2 from UTM (Vot Number:14H08).

REFERENCES:

[1] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, 2011.

[2] B. Ellis, *Real-Time Analytics: Techniques to Analyze and Visualize Streaming Data*. John Wiley & Sons, 2014.

[3] N. Anand and M. Kumar, “Modeling and optimization of extraction-transformation-loading (ETL) processes in data warehouse: An overview,” in *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, 2013, pp. 1–5.

[4] S. K. Bansal and S. Kagemann, “Integrating Big Data: A Semantic Extract-Transform-Load Framework,” *Computer*, vol. 48, no. 3, pp. 42–50, Mar. 2015.

[5] S. Sharma and R. Jain, “Modeling ETL Process for Data Warehouse: An Exploratory Study,” in *2014 Fourth International Conference on Advanced Computing Communication Technologies*, 2014, pp. 271–276.

[6] P. Tiwari, “Advanced ETL (AETL) by integration of PERL and scripting method,” in *2016 International Conference on Inventive Computation Technologies (ICICT)*, 2016, vol. 3, pp. 1–5.

[7] P. A. Priya, S. Prabhakar, and S. Vasavi, “Entity resolution for high velocity streams using semantic measures,” in *2015 IEEE International Advance Computing Conference (IACC)*, 2015, pp. 35–40.

[8] R. Kimball and M. Ross, *The Kimball Group Reader: Relentlessly Practical Tools for Data Warehousing and Business Intelligence*. Wiley Publishing, 2010.

[9] E. Mehmood and M. A. Naeem, “Optimization of cache-based semi-stream joins,” in *2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, 2017, pp. 76–81.

[10] V. Arora, F. Nawab, D. Agrawal, and A. E. Abbadi, “Multi-representation Based Data Processing Architecture for IoT Applications,” in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2234–2239.

[11] A. Wibowo, “Problems and available solutions on the stage of Extract, Transform, and Loading in near real-time data

warehousing (a literature study),” in *2015 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, 2015, pp. 345–350.

[12] Z. Lin, Y. Lai, C. Lin, Y. Xie, and Q. Zou, “Maintaining Internal Consistency of Report for Real-Time OLAP with Layer-Based View,” in *Web Technologies and Applications*, 2011, pp. 143–154.

[13] M. A. Bornea, A. Deligiannakis, Y. Kotidis, and V. Vassalos, “Semi-Streamed Index Join for near-real time execution of ETL transformations,” in *2011 IEEE 27th International Conference on Data Engineering*, 2011, pp. 159–170.

[14] R. S, S. B. B, and N. K. Karthikeyan, “From Data Warehouses to Streaming Warehouses: A Survey on the Challenges for Real-Time Data Warehousing and Available Solutions,” *Int. J. Comput. Appl.*, vol. 81, no. 2, pp. 15–18, Nov. 2013.

[15] F. Majeed, M. S. Mahmood, and M. Iqbal, “Efficient data streams processing in the real time data warehouse,” in *2010 3rd International Conference on Computer Science and Information Technology*, 2010, vol. 5, pp. 57–61.

[16] T. Jain, R. S, and S. Saluja, “Refreshing Datawarehouse in Near Real-Time,” *Int. J. Comput. Appl.*, vol. 46, no. 18, pp. 24–29, May 2012.

[17] R. P. Deb Nath, K. Hose, T. B. Pedersen, and O. Romero, “SETL: A programmable semantic extract-transform-load framework for semantic data warehouses,” *Inf. Syst.*

[18] D. Ostrowski, N. Rychtyckyj, P. MacNeille, and M. Kim, “Integration of Big Data Using Semantic Web Technologies,” in *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, 2016, pp. 382–385.

[19] S. Fang *et al.*, “An Integrated System for Regional Environmental Monitoring and Management Based on Internet of Things,” *IEEE Trans. Ind. Inform.*, vol. 10, no. 2, pp. 1596–1605, May 2014.

[20] X. Li and Y. Mao, “Real-Time data ETL framework for big real-time data analysis,” in *2015 IEEE International Conference on Information and Automation*, 2015, pp. 1289–1294.

[21] R. Mukherjee and P. Kar, “A Comparative Review of Data Warehousing ETL Tools with New Trends and Industry Insight,” in *2017 IEEE 7th International Advance Computing Conference (IACC)*, 2017, pp. 943–948.

[22] “Addressing BI Transactional Flows in the Real-Time Enterprise Using GoldenGate TDM - (Industrial Paper) - Semantic Scholar.” [Online]. Available: /paper/Addressing-BI-Transactional-Flows-in-the-Real-Time-Pareek/10c635702e00476d81a6a4d3fb29c336659d7f10. [Accessed: 31-Mar-2017].

[23] N. Narendra, K. Ponnalagu, S. Tamilselvam, and A. Ghose, “Goal-driven context-aware data filtering in IoT-based systems,” *Fac. Eng. Inf. Sci. - Pap. Part A*, pp. 2172–2179, Jan. 2015.

[24] C. R. Valencio, M. H. Marioto, G. F. D. Zafalon, J. M. Machado, and J. C. Momente, “Real Time Delta Extraction Based on Triggers to Support Data Warehousing,” *DeepDyve*, Dec. 2013.

[25] J. Zuters, “Near Real-Time Data Warehousing with Multi-stage Trickle and Flip,” in *Perspectives in Business Informatics Research*, 2011, pp. 73–82.

[26] F. Dehne, Q. Kong, A. Rau-Chaplin, H. Zaboli, and R. Zhou, “A distributed tree data structure for real-time OLAP on cloud architectures,” in *2013 IEEE International Conference on Big Data*, 2013, pp. 499–505.

[27] M. R. Murazza and A. Nurwidyantoro, “Cassandra and SQL database comparison for near real-time Twitter data warehouse,” in *2016 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, 2016, pp. 195–200.

[28] M. Mesiti, S. Valtolina, L. Ferrari, M. S. Dao, and K. Zettsu, “An editable live ETL system for Ambient Intelligence environments,” in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015, pp. 393–394.

[29] “ETL Evolution for Real-Time Data Warehousing,” *TechRepublic*. [Online]. Available: http://www.techrepublic.com/resource-library/whitepapers/etl-evolution-for-real-time-data-warehousing/. [Accessed: 30-Mar-2017].

Journal of Theoretical and Applied Information Technology
30th November 2017. Vol.95. No 22
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Challenges for ETL in Near Real-Time Environment and Solution Approach from Related Works

Feature			Extract, Transform and Load (ETL)			Challenge	Solution Approach
High Availability	Low Latency	Horizontal Scalability	E	T	L		
o			o			- Heterogenous Data Source	- Stream Processor - Semantic Web Technologies Toolkits - Join Relation Algorithm, e.g. SSIJ - Entity Resolution (ER) Processing for stream data
o			o			- Heterogenous Data Source	- Artificial Intelligence with human interface
o	o	o	o			- Heterogenous Data Source	- Parallelized Algorithm, e.g. P-CACHEJOIN
o	o	o	o			- Heterogenous Data Source	- Multi-representation based data processing architecture
o			o			- Backup Data	- Server for Replication - Log-Based CDC - Separate processor & configurations with integration tools
o			o		o	- OLAP Inconsistency	- Snapshot - Real-Time Data Cache (RTDC) - Layer-Based View - Real-Time Operational Database
o					o	- OLAP Inconsistency	- Dynamic Mirror
		o			o	- OLAP Inconsistency	- Staging OLAP outside data warehouse update period - Cloud-based Real-Time OLAP (CR-OLAP)
	o		o			- Multiple Data Source Integration	- Combine change data capture - Stream processing and data integration tools
	o		o			- Multiple Data Source Integration	- Staging Temporary Tables for receiving and storing
	o		o			- Data Source Overload	- Update significance and record changed method - CDC Log Based Techniques - Special format for CDC
	o			o		- Master Data Overhead	- Separate Master Data Cache and Data Base Queue - CDC Log-Based Techniques
	o			o		- Intermediate Server Aggregation	- Extract Load Transform (ELT)
	o			o		- Separate Server for Aggregation	- Extract Load Transform (ELT)
	o				o	- Performance Degradation	- Staging Table - Multi Stage Trickle and Flip
	o	o	o			- Performance Degradation	- System architecture based on different modules, servers to perform separate goals
	o	o	o			- Performance Degradation	- Distributed System Modeling
o	o		o			- Performance Degradation	- Segmentation by time - Advanced ETL
o	o		o			- Performance Degradation	- Remodeling Source independence

o : relevance