Fahd Sabry Esmail Ali

A Survey of Real-Time Data Warehouse and ETL

Article Info:

Received 09 July 2014
Accepted 24 August 2014

UDC 004.6

Recommended citation:

Esmail Ali, F.S. (2014). A Survey of Real-Time Data Warehouse and ETL. *International Scientific Journal of Management Information Systems, 9* (3), 03-09.

Summary

Data Warehouses (DW) are typically designed for efficient processing of read only analysis queries over large data, allowing only offline updates at night. The current trends of business globalization and online business activities available 24/7 means DW must support the increasing demands for the latest versions of the data. Real-Time data warehousing aims to meet the increasing demands of Business Intelligence (BI) for the latest versions of the data. Informed decision-making is required for competitive success in the new global marketplace, which is fraught with uncertainty and rapid technology changes. Decision makers must adjust operational processes, corporate strategies, and business models at lightning speed and must be able to leverage business intelligence instantly and take immediate action. Sound decisions are based on data that is analyzed according to well-defined criteria. Such data typically resides in a DW for purposes of performing statistical and analytical processing efficiently. Achieving Real-Time data warehousing is highly dependent on the choice of a process in data warehousing technology known as Extract, Transform, and Load (ETL). This process involves: (1) Extracting data from outside sources; (2) Transforming it to fit operational needs; and (3) Loading it into the end target (database or data warehouse). Not all ETL's are equal when it comes to quality and performance. As such, optimizing the ETL processes for real time decision making is becoming ever increasingly crucial to today's decision-making process. An effective ETL leads to effective business decisions and yields extraordinary decision- making outcomes. This study overviews the theory behind ETL and raises a research vision for its evolution, with the aim of improving the difficult but necessary data management work required for the development of advanced analytics and BI.

Keywords

data warehouse, ETL, load, real-time

1. Introduction

Data warehousing provides architectures and tools for business executives to systematically organize, understand, and use their data to make strategic decisions. With the explosive advent of the Internet, broadband communication, mobile computing, and access to cloud computing, the past couple of decades gave a new meaning to the phrase "information overload". Companies need to consider how to adopt and utilize real-time data and information into the fabric of their decision-making or risk falling behind their competitors.

Indeed, this is an era of unprecedented data copiousness and accumulation. The utter variety of new information available on diverse platforms of electronic data sources has changed the way we live, collaborate, conduct business, undertake research, and make decisions; however, the increased reliance upon networked data has introduced unique data quality challenges. Organizations demand for quick access to new insights has led to predictive analytics for forecasting emerging demands, risks and opportunity. Advanced analytics apply statistical and predictive algorithms to forecasting, correlation, and trend analysis. In contrast, traditional data warehousing and BI have typically been associated with historical analysis and reporting. Advanced statistical and predicative analysis takes advantage of the large data sets (big data) stored within DW to foresee risk, anticipate customer demand, and formulate more successful product and service offerings.

The advanced analytics are highly dependent on access to the most recent, real-time business data, hence, DW must have instantaneous real-time access to business transactions. As the advanced analytics requirements get closer to real-time, the software applications must tolerate some amount of data incompleteness or inaccuracy, as it is not financially or technically feasible to provide 100% of the data within such strict time requirements (Henschen, 2011). According to Inmon (1996), a leading architect in the construction of DW systems, "A DW is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process".

2. Data Warehouse characteristics

The four keywords, subject-oriented, integrated, time-variant, and nonvolatile, distinguish DW from other data repository systems, such as relational database systems, transaction processing systems, and file systems.

A. Subject-oriented

A DW is organized around major subjects, such as customer, supplier, product, and sales. Rather than concentrating on the day-to-day operations and transaction processing of an organization, a DW focuses on the modeling and analysis of data for decision makers. Hence, DW typically provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

B. Integrated

A DW is usually constructed by integrating multiple heterogeneous sources, such as relational databases, flat files, and on-line transaction records.

C. Time-variant

Data are stored to provide information from a historical perspective (e.g., the past 5-10 years). Every key structure in the DW contains, either implicitly or explicitly, an element of time.

D. Nonvolatile

A DW is always a physically separate store of data transformed from the application data found in the operational environment. Due to this separation, a DW does not require transaction processing, recovery, and concurrency control mechanisms.

3. Stars, snowflakes, and fact constellations

A DW, however, requires a concise, subject-oriented schema that facilitates on-line data analysis. The most popular data model for a DW is a multi-dimensional model. Such a model can exist in the form of a star schema, a snowflake schema, or a fact constellation schema.

A. Star Schema

The most common modeling paradigm is the star schema, in which the DW contains a large central table (fact table) containing the bulk of the data, with no redundancy, and a set of smaller attendant tables (dimension tables),

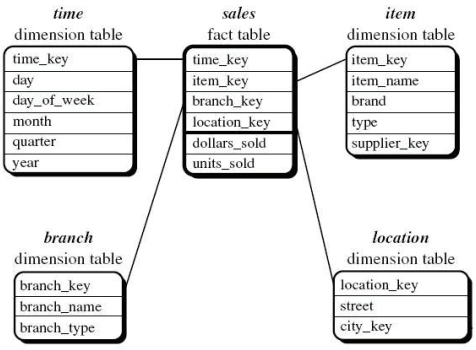

Figure 2 A simple star schema

B. Snowflake Schema

The snowflake schema is a variant of the star schema model, where some dimension tables are normalized, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.

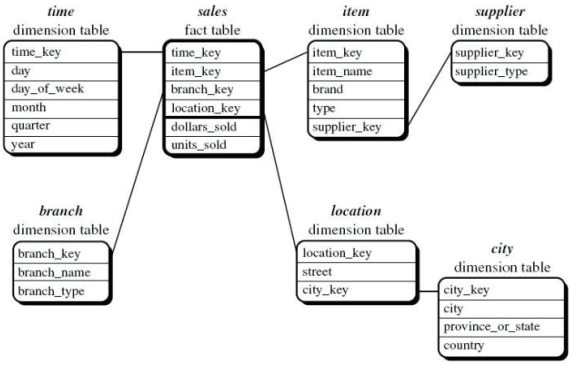

Figure 3 A simple snowflake schema

C. Galaxy Schema

Sophisticated applications may require multiple fact tables to share dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation.

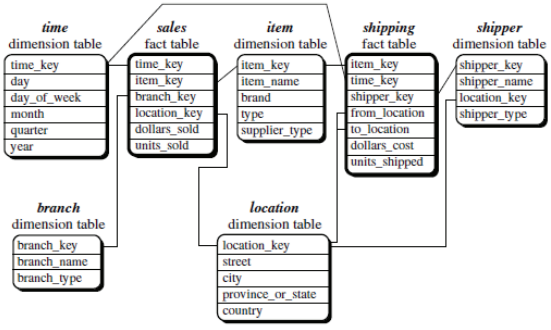

Figure 4 Galaxy schema of a DW for sales and shipping

4. Data Warehouse architecture

DW often adopt a three-tier architecture.

A. The Bottom Tier

The bottom tier is a warehouse database server that is almost always a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external. These tools and utilities perform data extraction, cleaning, and transformation.

B. The Middle Tier

The middle tier is an OLAP server that is typically implemented using either a relational OLAP (ROLAP) model, that is, an extended relational DBMS that maps operations on multidimensional data to standard relational operations; or a multidimensional OLAP (MOLAP) model, that is, a special- purpose server that directly implements multidimensional data and operations.

C. The Top Tier

The top tier is a front-end client layer, which contains query and reporting tools, analysis tools, and/or data mining tools , extraction, cleaning, and transformation.

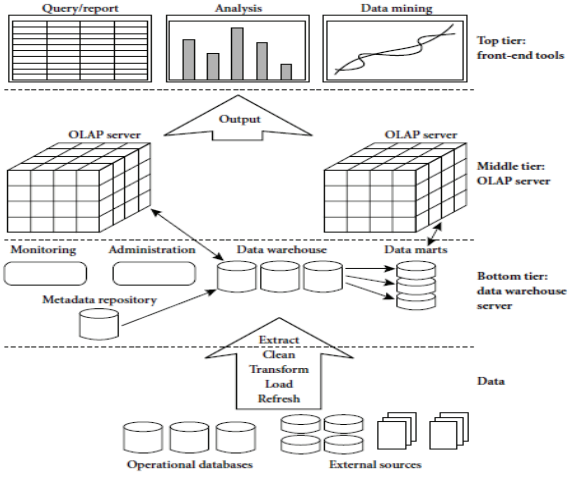

Figure 5 A three-tier data warehousing architecture

Four different views regarding the design of a DW must be considered: the top-down view, the data source view, the DW view, and the business query view.

A. Top-Down View

The top-down view allows the selection of the relevant information necessary for the DW for matching the current and future business needs.

B. Data Source View

The data source view exposes the information being captured, stored, and managed by operational systems. This information may be documented at various levels of detail and accuracy, from individual data source tables to integrated data source tables. Data sources are often modeled by traditional data modeling techniques, such as the entity-relationship model or CASE (computer-aided software engineering) tools.

C. Data Warehouse View

The DW view includes fact tables and dimension tables. It represents the information that is stored inside the DW, including pre-calculated totals and counts, as well as information regarding the source, date, and time of origin, added to provide historical context.

D. Business Query View

The business query view is the perspective of data in the DW from the viewpoint of the end user.

5. Data Warehouse design

A DW can be built using a top-down approach, a bottom-up approach, or a combination of both. The top-down approach starts with the overall design and planning. It is useful in cases where the technology is mature and well known, and where the business problems that must be solved are clear and well understood. The bottom-up approach starts with experiments and prototypes. This is useful in the early stage of business modeling and technology development. It allows an organization to move forward at considerably less expense and to evaluate the benefits of the technology before making significant commitments. In the combined approach, an organization can exploit the planned and strategic nature of the top-down approach while retaining the rapid implementation and opportunistic application of the bottom- up approach. In general, the warehouse design process consists of the following steps:

A. Choose a Business Process to Model.
B. Choose the Grain of the Business Process.
C. Choose the Dimensions.
D. Choose the Measures.

6. Data Warehouse & ETL

The typical ETL-based DW uses staging, integration, and access layers to house its key functions. The staging layer or staging database stores raw data extracted from each of the disparate source data systems. The integration

layer integrates the disparate data sets by transforming the data from the staging layer often storing this transformed data in an operational data store (ODS) database.

The integrated data is then moved to yet another database, often called the DW database, where the data is arranged into hierarchal groups often called dimensions and into facts and aggregate facts. ETL plays an important role in DW architecture since these ETL processes move the data from transactional or sources systems to data staging areas and from staging areas into the DW. Demands for real-time data warehousing result from the recent trends of business globalization, 24x7 operations, ever increasing data volumes, competitive pressures, demanding customers and increased demand by decision makers for real-time data (Ankorion, 2005).

These current trends require business to have access to the most updated data for analysis and statistical purposes, which necessitates a requirement for building real-time data warehousing and ETL. Techniques to achieve real-time data warehousing include the Change Data Capture (CDC) technique and the integration of change data capture with existing ETL processes to maximize the performance of ETL and achieve real-time ETL). The CDC integration with existing ETL tools provides an integrated approach to reduce the amount of information transferred while minimizing resource requirements and maximizing speed and efficiency. In contrast, migrating the data into DW using conventional ETL tools has a latency problem with the large volumes of data sets because ETL processes consume substantial CPU resources and time for large data sets.

7. ETL operations

A. Data Extraction

Taking out the data from a variety of disparate source systems correctly is often the most challenging aspect of ETL. Objective: Convert the data into a single format which is appropriate for transformation processing.

Extraction types:

Full extraction: In this type of extraction data from the source system is completely extracted.

Incremental extraction: In this type of extraction only the changes made to the source systems. Example, Change Data Capture (CDC) is mechanism that uses incremental extraction.

B. Data Transformation

This converts data from legacy or host format to warehouse format using similar steps:

- Selecting only certain columns to load;
- Translating coded values;
- Encoding free-form values;
- Deriving a new calculated value;
- Sorting;
- Joining data from multiple sources;
- Generating surrogate-key values;
- Splitting a column into multiple columns;
- Aggregation;
- Applying forms of simple, complex data validation.

C. Data Load

Loads the data into the end target, usually the DW. Some DW may overwrite existing information with cumulative information; frequently updates with extracted data are performed on hourly, daily, weekly, or monthly basis.

Mechanisms to load a DW include:

- SQL loader: generally used to load flat files into DW.
- External tables: this mechanism enables external data to be used as a virtual table which can be queried and joined before loading into the target system.
- Oracle Call Interface (OCI) and direct path Application Programming Interface (API): are methods used when the transformation process is done outside the database.
- Export/Import: this mechanism is used if there are no complex transformations and data can be loaded into target DW.

8. Evolution of ETL

With the evolution of BI, ETL tools have undergone advances and there are three distinct generations of ETL tools.

The First-generation ETL tools were written in the native code of the operating system platform and would only execute on the native operating system. The most commonly generated code was COBOL code because the first generation data was stored on mainframes. These tools made the data integration process easy since the native code performance was good but there was a maintenance problem.

Second generation ETL tools have proprietary ETL engines to execute the transformation

processes. Second generation tools have simplified the job of developers because they only need to know only one programming language i.e. ETL programming. Data coming from different heterogeneous sources should pass through the ETL engine row by row and be stored on the target system. This was a slow process and this generation of ETL programs suffered from a high performance overload.

Third Generation ETL tools have a distributed architecture with the ability to generate native SQL. This eliminates the hub server between the source and the target systems. The distributed architecture of third generation tools reduces the network traffic to improve the performance, distributes the load among database engines to improve the scalability, and supports all types of data sources. Third Generation ETL uses relational DBMS for data transformations. In this generation the transformation phase does processing of data rather than row by row as in second generation ETL tools. "In the ETL architecture, all database engines can potentially participate in a transformation—thus running each part of the process where it is the most optimized. Any RDBMS can be an engine, and it may make sense to distribute the SQL code among different sources and targets to achieve the best performance. For example, a join between two large tables may be done on the source" (De Montcheuil, 2005). RDBMS have power for data integration; ETL tools are taking the advantage of this feature of the RDBMS to improve their performance.

9. Real-Time DW and ETL techniques

Increasingly there is a need to support and make business decisions in near real-time based on the operational data itself.

Typical ETL architectures are batch update oriented and cause a significant lag in the currency of information at the DW.

We question the performance effectiveness of typical batch ETL architectures and near real-time based updates on operational data and raise the questions to address instant-time research in order to address the timely business-decision making process. We raise the issue that the current ETL process needs to move away from periodic refreshes to continuous updates; however online updating of DW gives rise to challenges of view synchronization and resource allocations. "To cope with real-time requirements, the DW must be able to enable continuous data integration, in order to deal with the most recent business data" (Santos & Bernardino, 2009).

View synchronization problems arise when views are composed of data derived from multiple data sources being updated indiscriminately. Resource challenges result when there are conflicting resource demands of long-term analysis queries in the presence of concurrent updates. In traditional ETL tools, loading is done periodically during the downtime and during this time no one can access the data in DW. The separation between querying and updating clearly simplifies several aspects of the DW implementation, but has a major disadvantage that the DW is not continuously updated.

Traditional ETL tools are not capable enough to handle such continuous inserts or updates with no DW down time. In real time DW loading is done continuously as opposed to a periodic basis in traditional approaches. One approach to the general architecture of a near real time DW consisting of the following elements:

(a) Data Sources hosting the data production systems that populate the DW, (b) an intermediate Data Processing Area (DPA) where the cleaning and transformation of the data takes place and (c) the DW (Vassiliadis & Simitsis, 2008).

The role of the data processing area (DPA) is to: a) cleanse and transform the data in the format required by the DW; b) act as the regulator for the DW (in case the warehouse cannot handle the online traffic generated by the source); and c) perform various tasks such as check pointing, summary preparation, and quality of service management.

"A Warehouse Flow Regulator (WFlowR) orchestrates the propagation of data from the DPA to the warehouse based on the current workload from end users posing queries and the requirements for data freshness, ETL throughput and query response time." (Vassiliadis & Simitsis, 2008). Third generation ETL tools are using techniques to achieve real time data warehousing without causing downtime. Some of the real time ETL techniques are found in the research of J. Langseth (Langseth, 2008) that include:

1) Near real-time ETL: The cost effective solution for applications that do not have a high demand for real time data is to just increase the frequency of loading, for ex: from daily to twice a day.

2) Direct Trickle feed: In this approach true real time data can be achieved by continuously moving the changed data from the source systems by inserting or updating them to the fact tables. There is a scalability problem with this approach because complex queries don't perform well with continuous updates. Constant updates on tables, which are being queried by reporting or OLAP tools leads to degrading the query performance of the DW.

3) Trickle and flip: In this approach, data is inserted or updated into staging tables which are in the same format as target tables. The real time data is stored in staging tables, which have same format as historical data in target tables. The DW can access fresh data instantly by getting a copy from the staging tables into the fact tables, the time window for refreshing the DW can vary from hours to minutes.

4) External real-time data cache: In this approach real-time data is stored outside DW in an external real time data cache (RTDC). The function of RTDC is to load the real time data into database from source systems. It resolves the query contention and scalability problem by directing the queries to RTDC which access real time data. With this approach, there is no additional load on the DW as the real time data lies on separate cache data base. It provides up-to-the-second data and users don't wait for queries to get executed because they are so quick (Langseth, 2008).

The physical data residing in OLAP is in its de-normalized form for query processing while relational online analytical processing (ROLAP) needs data to be in 3rd Normal form because it uses the relational queries for processing the data. Multidimensional analytical processing (MOLAP) can be used because data is built from a data cube, which is separate from transactional data.

10. Real-time data BI techniques

One proposal for real-time BI architecture requires that the data delivery from the operational data stores to the DW must occur in real-time in the format referred to data streams of events (Agrawal, 2009). The usage of real-time data event streams eliminates the reliance on batched or offline updating of the DW.

This architecture also introduces the middleware technology component, referred to as the stream analysis engine. This stream analysis engine performs a detailed analysis of the incoming data before it can be integrated into the DW to identify possible outliers and interesting patterns. The goal of the stream analysis process is to "extract crucial information in real-time and then have it delivered to appropriate action points which could be either tactical or strategic" (Agrawal, 2009). Complex Event Processing Engines (CEP engines) such as Stream-base enable business users to specify the patterns or temporal trends that they wish to detect over streaming operational data known as events. Decision makers can then take appropriate actions when specific patterns occur.

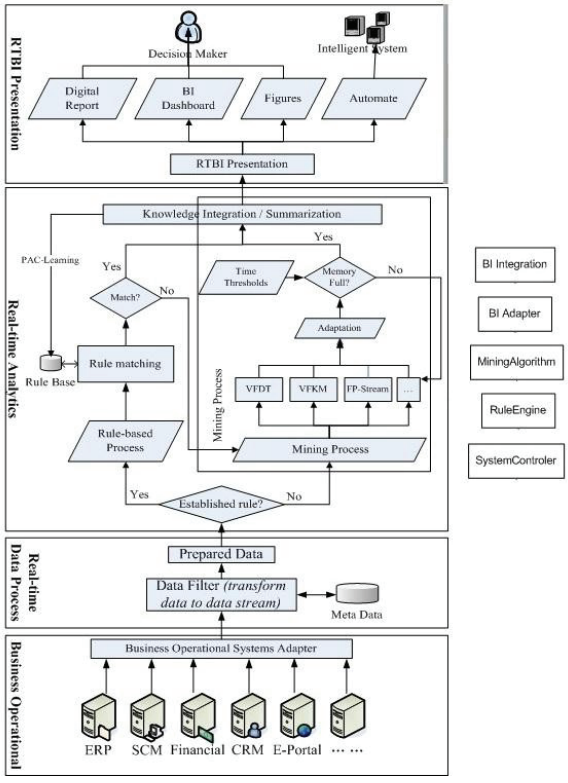

Figure 6 Real-time BI architecture

The origins or CEP engines were in the financial domain where they were applied to algorithmic stock trading. More recently they are being applied to make decisions in real-time such as the click stream analysis of manufacturing process monitoring (Chaudhuri, Dayal, & Narasayya, 2011).

The arrival of events from the input streams trigger the query processing, and the queries are performed continuously as long as events arrive in the input stream. One major technical challenge is that the continuous running queries may reference data in the data base and impact near real-time requirements. A major challenge is that algorithms which require multiple passes over the data are no longer feasible for streaming data.

11. Conclusions

As the role of enterprises becomes increasing real-time such as real-time BI will be increasing important to such companies. In the traditional ETL approach, the most current information is not available. With the increase demands by businesses for real-time BI and Predictive Analytics, there is a need to build ETL tools, which provide real-time data into DW. Not every analysis task warrants real-time analysis. The trade-off between the overhead of providing real-time BI and DW, and the need for such an analysis calls for serious research and consideration. Otherwise, or the resulting system may have prohibited costs associated with it (Agrawal, 2009). The underlying technology components and custom solutions are prohibitively expensive. The importance, complexity and criticality of such an environment make real-time BI and DW a significant topic of research and practice; therefore, these issues need to be addressed in the future by both the industry and the academia (Vassiliadis, Simitsis, 2008).

Acknowledgment

My deepest gratitude to Dr. Ali El Bastawissy for his kind guidance, valuable advice and continuous encouragement during the progress of the "Advanced Data-Bases" Course.

References

Agrawal, D. (2009). The Reality of Real-Time Business Intelligence. In M. Castellanos, U. Dayal, & T. Sellis, *Proceedings of the 2nd International Workshop on Business Intelligence for the Real Time Enterprise (BIRTE 2008)* (pp. 75-88). Heidelberg: Springer.

Ankorion, I. (2005). Change data capture: Efficient ETL for Real- Time BI. *Information Management, 15* (1), 36-36.

Chaudhuri, S., Dayal, U., & Narasayya, V. (2011). An overview of Business Intelligence Technology. *Communications of the ACM , 54* (8), 88-98.

De Montcheuil, Y. (2005). *LESSON - Third-Generation ETL: Delivering the Best Performance.* Retrieved May 15, 2012 from TDWI: http://tdwi.org/articles/2005/10/13/thirdgeneration-etl-delivering-the-best-performance.aspx

Henschen, D. (2011). *BI and Information Management Trends.* Retrieved June 29, 2012 from Information Week: http://reports.informationweek.com

Inmon, W. H. (1996). *Building the Data Warehouse, 1st edition.* Indiana: Wiley Publishing Inc.

Santos, R.J., Bernardino, J. (2009). Optimizing data warehouse loading procedures for enabling useful-time data warehousing. In *Proceedings of the 2009 International Database Engineering & Applications Symposium* (pp. 292-299). New York: ACM

Vassiliadis, P., Simitsis, A. (2008).A method for the mapping of conceptual designs to logical blueprints for ETL processes. *Decision Support Systems 45*(1), 22–40.

Fahd Sabry Esmail Ali
Modern Academy for Computer Science and Information Technology
Department of Management Information Systems
Cairo
Egypt
Email: fahdsabry985@gmail.com