Real-Time Data Warehousing For Business Intelligence

Farrah Farooq
Punjab University College of Information Technology
(PUCIT) University of the Punjab
Allama Iqbal (Old) Campus, Anarkali, Lahore,
Pakistan
+92-(0)42-111-923-923
farrah.farooq@pucit.edu.pk

Syed Mansoor Sarwar
Punjab University College of Information Technology
(PUCIT) University of the Punjab
Allama Iqbal (Old) Campus, Anarkali, Lahore,
Pakistan
+92-(0)42-111-923-923
syed.sarwar@pucit.edu.pk

ABSTRACT

Real-time ETL and data warehouse multidimensional modeling (DMM) of business operational data has become an important research issue in the area of real-time data warehousing (RTDW). In this study, some of the recently proposed real-time ETL technologies from the perspectives of data volumes, frequency, latency, and mode have been discussed. In addition, we highlight several advantages of using semi-structured DMM (i.e. XML) in RTDW instead of traditional structured DMM (i.e., relational). We compare the two DMMs on the basis of four characteristics: heterogeneous data integration, types of measures supported, aggregate query processing, and incremental maintenance. We implemented the RTDW framework for an example telecommunication organization. Our experimental analysis shows that if the delay comes from the incremental maintenance of DMM, no ETL technology (full-reloading or incremental-loading) can help in real-time business intelligence.

Categories and Subject Descriptors

A.1 [**General Literature**]: Introductory and Survey.

H.2.4 [**Database Management**]: Systems – *query processing, relational databases.*

H.2.7 [**Database Management**]: Database Administration – *data warehouse and repository*.

General Terms

Languages, Management, Performance.

Keywords

Real-time data warehousing, multidimensional modeling business intelligence, real-time ETL, operational analysis, XML, star-schema, and incremental maintenance.

1. INTRODUCTION

Data warehouses (DW) have been increasingly used since the last decade for improving business intelligence in an organization. Traditional DWs usually provide one-day old historical data for analysis, while current transactional data remains in operational systems. Nowadays, business users require up-to-date or real-time data for the purpose of analysis, which necessitates the building of a real-time data warehouse (RTDW) [1].

RTDW brings two challenges into the data warehousing framework. First is related to the real-time data integration, i.e., extract, transform, and load (ETL) process and second is real-time incremental maintenance of DW multi-dimensional model (DMM). The real-time ETL tools help to identify, capture, transform, and load data-updates instantly to the data warehouse repository. The static, historical, and aggregated data warehouse model then needs to be either recomputed or incrementally maintained. The incremental maintenance means that only data-updates are propagated to the DMM. However, the choice of DMM (i.e., structured or semi-structured) greatly influences the speed of incremental maintenance. No real-time ETL technology is valuable if latency comes from the modeling of data-updates. Sensitive applications that are based on sensing geographic data for a cyclone, detecting real-time credit card, or telecommunications fraud do not tolerate this data-update latency.

Researchers and market vendors have focused more on the development of real-time data integration tools. Today many real-time ETL technologies exist for RTDW. Unfortunately, the issues related to real-time incremental maintenance of DMM have not been studied well in the past. Therefore, the main aim of this paper is to highlight the importance of the fact that the semi-structured DMM (i.e., XML DWs) has more power to incorporate real-time data updates quickly, rather than structured DMM (relational data cubes, MOLAP). The main contributions of this paper include:

- State-of-the-art review of real-time ETL technologies from the perspectives of data volumes, frequency, latency, and mode.
- Comprehensive literature survey on the two classes of DMM techniques, structured and semi-structured DMMs.
- With the help of an example of telecommunication organization data warehouse, which is nowadays one of the most competitive business arenas in today’s market, we explain two DMM techniques, XML and relational.

- Implementation of real-time data warehouse framework. Our experiments indicate that for RTDW, XML DMM suits well and has several advantages over relational DMM.

The paper is organized as follows. Section 2 discusses the state-of-the-art real-time ETL technologies proposed in literature. Section 3 presents a detail discussion on two classes of DMM and discuss the advantages of semi-structured DMM over structured DMM. The RTDW framework that we implemented for the analysis of DMM techniques and real-time business analysis is presented in section 4. We discuss the experimental setup and results in section 5. Finally we conclude our paper.

2. REAL-TIME ETL TECHNOLOGIES

Several ETL variants for incorporating real-time data are proposed in literature. We classify the existing ETL tools four categories: Traditional ETL, near real-time ETL (NR-ETL), ETL with direct data feed (ETL-DDF), and ETL with real-time data caches (ETL-RDC). This classification is according to the distinguishing characteristics of ETL tools. The *mode* describes whether data loading from operational sources is done in batch-mode or incremental-mode. The *frequency* indicates the scheduling of ETL process i.e., ETL is performed monthly, weekly, or daily. The *data volumes* and *latency* properties are inversely proportional to the frequency of ETL. The more frequent the ETL process is, less is the data volume and data latency. The NR-ETL tools simply increase the frequency of ETL without changing the load process, data models and reporting applications. In literature, most of the work has been done on this approach. Tools proposed in [2-7] are built on this approach. ETL-DDF provides a RTDW solution which is based on the DW schema change for directly inserting real-time data into the DW schema [8]. Constantly updating the same schema that is being used by OLAP applications can cause the data warehouse query performance to degrade. This technique requires building of a unified schema that can fulfill OLAP and OLTP requirements. Some ETL-RDC tools have been proposed in [9, 10]. As the name implies, this technique requires external high speed data caches for real-time data processing. It improves the query performance and data loading process, but as the real-time and historical data are separately stored it take considerably large time for a single query to answer both types of data. Table 1 summarizes the four characteristics of real-time ETL technologies belonging to different classes.

3. DATA WAREHOUSE MULTIDIMENSIONAL MODELING (DMM)

Generally, a multidimensional model is based on two concepts: business dimensions and facts. A *business dimension* is a concept that is used as constraints for analysis (e.g., customer, time, and product). Whereas, a *fact* describes a business measure, e.g., revenue, price, and discount. In literature, several DMMs have been proposed. We classify these DMMs into two categories: structured (i.e., relational and array-based) and semi-structured (i.e., XML). DMMs proposed in [11-16] belong to the relational category, most popular being the relational star-schema [12] and data cubes [15]. These models use either tables or multidimensional arrays to store dimensions and fact data. Usually fact table is centered between all the dimension tables and measure values in the fact table corresponding to the Cartesian-product of all the dimension tables. The second category of DMMs is semi-structured XML-DMMs [17-20]. Most of these techniques focus on web-data. In the approach presented in [17], the XML DMM design is carried out directly by the *XML documents*. These documents are modeled as *ordered trees* and the relationship among the business dimensions and facts are described by the sub-elements in the XML schema. An *XML schema* gives the representation of the XML document structure and consists of type definitions and element declarations.

Table 1. Characteristics of real-time ETL technologies

ETL Types	*Mode*	*Frequency*	*Data Volumes*	*Latency*
Traditional ETL	Batch	Monthly, weekly, daily	High	Hours
NR-ETL	Incremental	Daily, twice a day, hourly, minutes	Low	Minutes
ETL-DDF	Incremental	Minutes, seconds	Low	Minutes, seconds
ETL-RDC	Incremental	Minutes, seconds	Low	Minutes, seconds

3.1 Advantages of XML-DMM over Relational-DMM

Although, for the last many years relational-DMMs have been used to build data warehouses, but there are several reasons that describe the importance and need of XML-DMM in today's business organizations. Firstly, XML has become the data storage and exchange standard on the Internet and nowadays most of the organizations incorporate web data for analysis purpose. In [21], authors explain how XML-DMMs can be efficiently utilized for scalable and optimized *data integration of heterogeneous data* (i.e., data available in different storage engines and accessible through multiple protocols/interfaces). XML-DMM has implicit capability of storing free-structure, free-format, and variable size data. Apart from the Internet data, some scientific data with unknown format and telecommunication data may be incorporated into a data warehouse repository and analyzed from different aspects. Relational-DMM has a strict structure and uniform formats, and is incapable of storing such complex data [22].

Secondly, in relational-DMM, the fact table stores only numeric measures aggregated along different granularity levels, but in XML-DMM, the XML-cube can store any type of measures and provide flexibility in real-time analysis in data warehouse.

The third characteristic on which we evaluate the efficiency of any DMM is *aggregate query processing*, analytical queries which include *group-by* clause and set of *aggregation operations* like max, min, sum, avg, count etc. In the early stages of data warehousing research, considerable efforts [23, 24] was put in on aggregate query processing in relational-DMM. These operations are well-supported by SQL in relational DBMS [25, 26]. To accelerate query processing, one approach is to materialize, pre-compute and physically store multiple aggregated fact tables so that expensive join operations can be eliminated. But in operational environment, incrementally maintaining multiple aggregated fact tables is not a realistic approach. Recently, research on indexing and aggregate query processing in XML-DMM has been described in [27, 28, 29]. An XML query with some selection condition can be presented by a tree structure and is called a *twig pattern*. The main operation in XML query processing is to find all the matching occurrences of a twig query in an XML document. Because an XML query processing requires parsing of the XML document structure, it may seem inefficient and time-consuming than SQL queries. But in XML-DMM, Xquery helps in *multi-level (nested) grouping* of complex data, whereas SQL lacks this support. The XML query language, Xquery 1.1 [28], has a construct to explicitly express group-by in FLWOR expressions. In [30], a new XML-DMM and indexing technology is proposed for efficient group-by aggregation query.

The fourth important aspect that has recently gained importance in the DMM study is the *incremental maintenance of data warehouse,* which means loading of data-updates from operational sources to the data warehouse. In relational-DMM, incremental maintenance can be done in two ways. The simplest approach is to re-build the base fact tables and aggregated fact tables whenever there is a change in the operational source. Another approach is to use an alternative storage structure like multidimensional arrays [31] and cubetrees [32] for building aggregated fact tables. This approach requires complex implementation and large processing time. As compare to relational-DMM, less research has been done on the incremental maintenance of XML-DMM. Recently, some techniques have been proposed. In [33] and [34], an XML-DMM storage method and incremental maintenance algorithm are discussed. These techniques leverage the relational DBMS to store XML-DMM and use identifiers for the maintenance of XML-DMM. The XML-DMM update support has been added to the Xquery language [35] and in [36] authors have proposed an update translator module to simplify the Xquery complex queries. This module operates on intermediate algebraic representation of XML queries.

4. REAL-TIME DATA WAREHOUSING FRAMEWORK

Today, the network switches of cellular companies produce billions and trillions of bytes of data per hour. RTDW helps in proper management of large amount of cellular data and track the measured values such as increases in accounts, call volumes, and revenues. We take an example of Utilize-Service (US) records which are generated automatically by a network switch or a network program whenever a customer buys or utilizes different value-added-service packages. These records can be used for billing purposes and to answer several operational queries such as: What packages the customers are using and in what combinations? Which are the most profitable service packages and why? Are the price and service incentives attracting and retaining the most profitable customers?

In this paper, we present a RTDW framework for an example telecommunication organization. The overall architecture of RTDW framework is shown in figure 1. The framework includes four main modules namely: (1) an operational data source which contains current data, (2) an ETL engine for transforming and loading the data (initial and data-updates) to the data warehouse repository, (3) data warehouse repositories based on two DMM (XML and relational), and (4) the OLAP query engine. Following are the details related to the implementation of these modules.

4.1 Operational Data Source

This repository consists of relational tables which contain the current data related to Utilize-Service (US) records. First of all, we explain the basic entities involved in the US records. Every Customer has a unique customerNIC and contains attributes customerName, address, and simCardNo. There are value added Services (e.g. Call, GPRS, MMS, SMS, GPS), which are offered to customers. Each customer can subscribe to more than one service. Each service has a unique serviceID and is described by serviceName and description. The Package belongs to a service, e.g., [SMS: unlimited sms, 1000 sms /day etc] and [Call: 1000 free talktime, conference call, midnight call offer etc.]. A package has packageId, packageName, description, actualCost, governmentTax, and serviceCategory. When a package belonging to a service is utilized by a customer, the date and serviceCharges are recorded against each transaction.

4.2 ETL Engine

The ETL system is the most resource consuming system in a data warehouse framework in terms of implementation and processing time. Generally, there are three types of ETL jobs written for the loading of a data warehouse repository. The first time a data warehouse is populated with an exhaustive *initial-loading ETL job*. For the refreshment of data warehouse, typically *full-reloading ETL jobs* and *incremental-loading ETL jobs* are required.

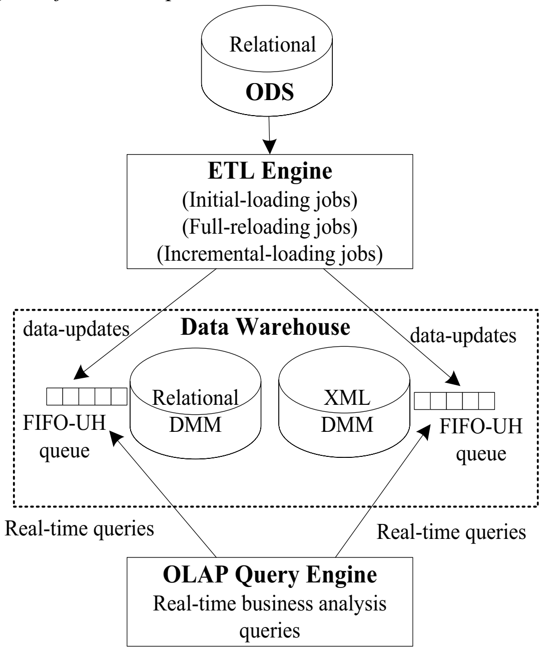

Figure 1. Architecture of real-time data warehouse framework

In [37], the authors have explored the problem of deriving an incremental-loading job from an initial-loading ETL job and they stress that incremental-loading is better approach than the other two approaches.

In this paper, we explore two problems: 1) what is an impact of full-reloading and incremental-loading ETL jobs on the two types of DMMs (Relational and XML)? 2) Which DMM is efficient in incremental DMM maintenance and answering real-time OLAP queries? The change data capture (CDC) techniques discussed in [38] are used to capture only data-updates from ODS. We are interested in only the propagation of *newly-inserted-tuples.* For the implementation of the real-time ETL module, we write the above three types of ETL jobs for relational-DMM with the help of transformation rules given in [37]. The general idea of these transformations is to map an initial-loading ETL expression $\mathcal{E}$ to another ETL expression $\mathcal{E}_{new}$ to propagate data-updates from ODS to data warehouse. For the XML-DMM, we write incremental-loading ETL scripts by using Xquery language update support [35]. The details of the ETL expressions and Xquery ETL scripts used for the loading of relational-DMM and XML-DMM, respectively, are beyond the scope of this paper.

4.3 Data Warehouse Repository

We choose the relational star-schema approach [12] to build an example data warehouse for a telecommunication organization. The star-schema DMM defined for the US data warehouse is shown in the figure 2. It consists of a fact table UtilizeService (US) and dimensions Time, Customer, and Package. For the building XML DMM, we use the concepts proposed in [17]. Figure 3 shows the XML schema for the US data warehouse.

In RTDW, the maintenance of data warehouse and query execution carries out simultaneously. Therefore, there is a need to implement some scheduling algorithm to avoid the data warehouse refreshment anomalies [39]. We used the first in first out (FIFO) update high (UH) [40] scheduling algorithm to schedule the updates and queries. The basic idea of this algorithm is to high priority access of data warehouse to the ETL engine if there is any data-updates available to be delivered to data warehouse. Otherwise, the data warehouse repository can be used by the OLAP query engine, from where business analysts can perform real-time analysis. In this paper, we do not give the implementation details of FIFO-UH algorithm. Interested readers are referred to [40].

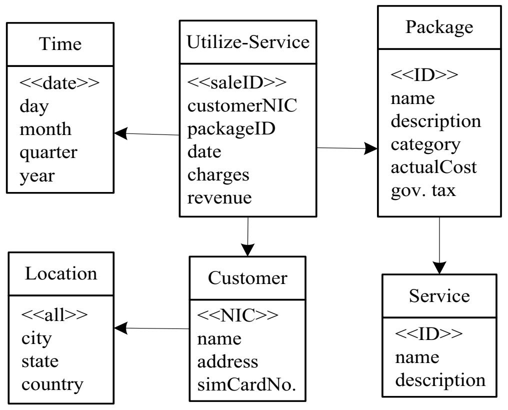

Figure 2. Relational DMM for US data warehouse

```
<schema>
<element name="SUR-Datawarehouse"
type="DWType"/>
  <complexType name="DWType">
  <sequence maxOccurs="unbounded">
    <element name="Utilize-Service"
type="UTType"/>
  </sequence>
  </complexType>
  <complexType name="UTType">
    <sequence>
      <element name="Customer" type="custType"
/>
      <element name="Package" type="packType"
/>
      <element name="Time" type="timeType" />
    </sequence>
    <attribute name="charges" type ="decimal"
/>
    <attribute name="Revenue" type ="decimal"
/>
  </complexType>
  <complexType name="custType">
    <sequence>
      <element name="Name" type="string"/>
      <element name="Address" type="addType"/>
      <element name="SimCardNo"
type="decimal"/>
    </sequence>
    <attribute name="id" type ="string" />
  </complexType>
  <complexType name="addType">
    <sequence>
      <element name="Country" type="string"/>
      <element name="State" type="string" />
      <element name="City" type="string" />
    </sequence>
  </complexType>
  <complexType name="packType">
    <sequence>
      <element name="Name" type="string"/>
      <element name="description"
type="string" />
      <element name="Service" type="scType" />
      <element name="GovTax" type="string" />
      <element name="Cost" type="decimal" />
    </sequence>
    <attribute name="id" type ="string" />
  </complexType>
  <complexType name="scType">
    <sequence>
      <element name="Name" type="string"/>
      <element name="description"
type="string" />
     </sequence>
  </complexType>
  <complexType name="timeType">
    <sequence>
      <element name="Year" type="decimal"/>
      <element name="Qaurter" type="string"/>
      <element name="Month" type="string" />
      <element name="Day" type="decimal" />
    </sequence>
  </complexType>
</schema>
```

Figure 3. XML schema for the US data warehouse

4.4 OLAP Query Engine

Suppose that an analyst wants to know when to withdraw a high priced 'SMS' service packages. The high priced 'SMS' service packages used by the customers are analyzed at the end of every quarter. The analyst compares the number of customers during this period, including the current US data with a standardized indicator. This indicator requires information about customers using the high-priced 'SMS' service package during a quarter. If the predicted number of customers is below some specific threshold, the service package may be withdrawn. The relational query that displays all the customers who have used the high priced 'SMS' packages in the one quarter time-period is given below:

```
SELECT UtilizeService.customerNIC,count(*)as
totalTransaction
FROM   UtilizeService,Customer,Package,Time
```

```
WHERE  UtilizeService.date = Time.date
AND    UtilizeService.packageId =
Package.packageID
AND    Time.quarter = currentdate.quarter; *
AND    Time.year = currentdate.year;
AND    Package.serviceCategory = 'SMS'
AND    Package.actualPrice >= 100
GROUPBY UtilizeService.customerNIC
```

For the analysis in XML data warehouses, XQuery [35], a query language is developed by the W3C. It consists of OLAP operations like extraction, filtration, construction, aggregation, navigation, and grouping. The following query is written in the XQuery expression FLWOR format for the above business analysis.

```
 LET $NewPackage := //Utilize-Service
/Package[/Service-category="SMS" and /cost
>=100]
LET $QuarterSales := //Utilize-Service /time
[/Quarter='Q4' and /Year='2009']
LET $NewSales := //Utilize-Service [/time =
$QuarterSales and /Package=$HighPricedPackage]
For $customerNIC in distinct-values
($NewSales/Customer/@id)
ORDERBY $customerNIC
RETURN
<CustomerTransactions>
<Customer> {$customerNIC} </Customer>
<TransactionCount>{count($NewSales)}
</TransactionCount>
</CustomerTransactions>
```

5. EXPERIMENTS

In this section, firstly we present the comparative performance results of the two types of ETL jobs (full-reloading and incremental) on two types of DMMs (relational and XML). Secondly, we explore the real-time query execution in an operational environment. Our experiment is based on the example telecommunication data warehouse. We populated the ODS relational data tables with sample tuples having cardinality as: Customer relation to be 200,000 tuples, Location to be 300 tuples, Time to be 5000 tuples, Package to be 100 tuples and Service to be 15 tuples. The data warehouse Utilize-Service fact table initially contains 720,000 records. We assume that data warehouse records have uniform storage size.

5.1 Experimental Setup

The experiments were carried out on Windows XP with 3.16 GHz Pentium 4 processor and 4GB RAM. We created the ODS database and two types of DMMs (relational and XML) in *MS SQL Server 2008 DBMS*. We used the RDBMS native XML support for storing XML-documents. For writing the ETL jobs (initial, full-reloading, and incremental), we used the open source ETL tool known as *Pentaho data integration* (Kettle) [41]. For testing the performance of the two types of data warehouse models in an operational environment, we wrote 20 different real-time queries similar to the query we explained in section IV. These real-time queries and the FIFO-UH scheduling algorithm were implemented in *C# .net Framework 2008*. The queries for XML-DMM are written by using the XQSharp [42], Xquery API for the .Net framework. Experimental results were obtained after running these queries 100 times in the system. The *percentage load* determines the number of *newly-inserted-tuples* that are propagated from ODS to data warehouse at each step. For example, 10% load means that 72,000 tuples are propagated to data warehouse.

5.2 Performance Comparison

In figure 4, we compare the time needed to propagate the *newly-inserted-tuples* from ODS to relational- and XML-DMM by using full-reloading ETL job. In this scenario each time, an ETL initial-loading is performed, resulting data is collected and compared to the previous data warehouse content. Since XML-DMM contains more information content than relational-DMM, full-reloading ETL performs better on relational-DMM, irrespective to the percentage load on the system.

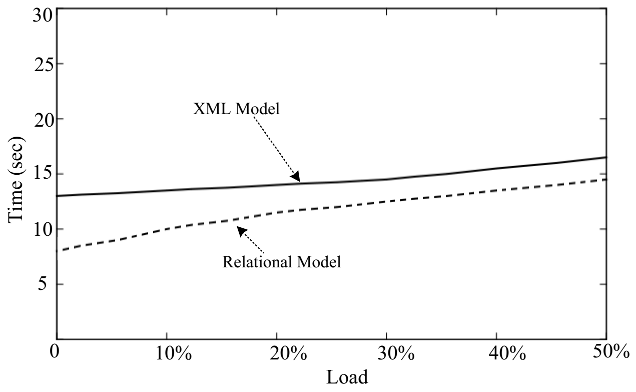

Figure 4. Full- reloading ETL job on relational- and XML-DMM

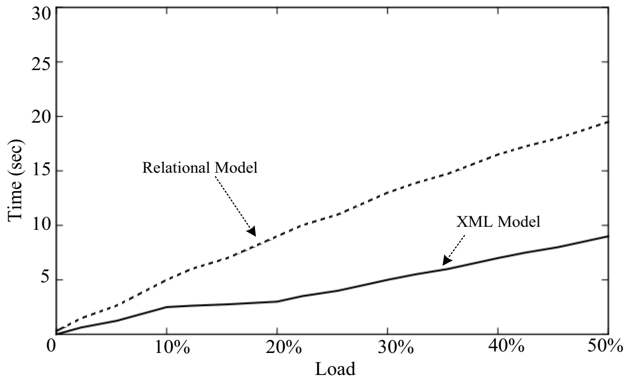

Figure 5. Incremental loading ETL job on relational- and XML-DMM

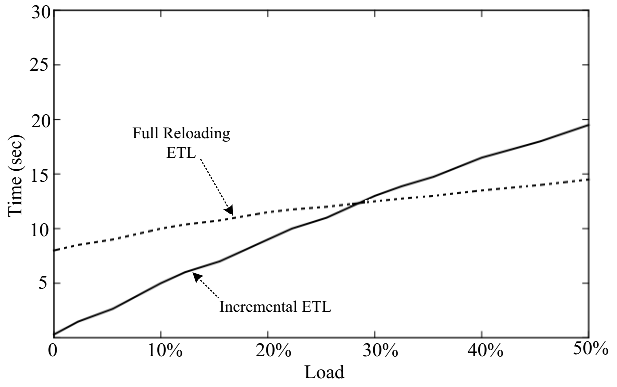

Figure 6. ETL performance comparison on relational-DMM

In figures 6 and 7, we compare the performance of two types of ETL jobs on the same model. Figure 6 clearly shows that on realtional-DMM, incremental-loading outperforms full-relaoding ETL job unless the percentage load (insertions of new tuples) is increased to beyond approximately 28%. After this threshold, the performance of the two jobs almost reverses. In figure 7, the same analysis is done on XML-DMM. But in this scenerio, incremental-loading out-performed full-loading regarless of the percentage load. The performance diffrence between jobs is significantly greater in XML-DMM than in relational-DMM. Although the performance gap between the two jobs shrinks with increase in load, however, even at 40% load incremental-ETL performs more than 100% better than full-reloading ETL Hence, if we use incremental-loading with XML-DMM combination, we can reduce the latency in the change propagation, making a data warehouse more avalaible for real-time queries.

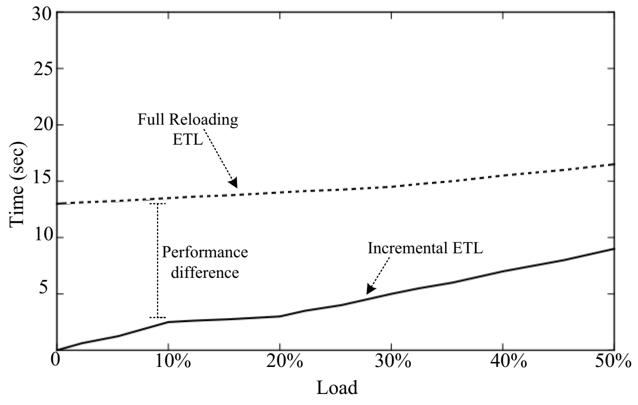

Figure 7. ETL performance comparison on XML-DMM

The 20 different real-time queries were run 100 times on a data warehouse (relational- and XML-DMM), where data-updates are propagated by incremental-loading ETL job and FIFO-UH algorithm is implemented for scheduling data-updates and real-time queries. In figure 8, we plot the query execution times against the percentage load increased step-wise. We observe that under operational environment, XML-DMM requires less time and performs much better than relational-DMM. Furthermore, the query execution time in XML-DMM becomes almost constant for 30% load and beyond. This result validates the assumptions that if we use incremental loading ETL job on XML-DMM, we can significantly enhance the OLAP query performance.

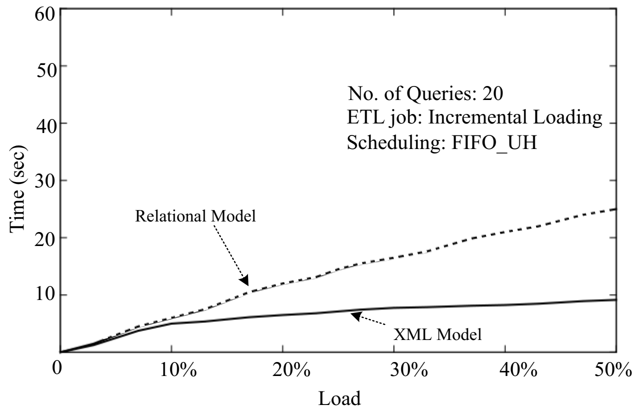

Figure 8. Query execution in relational- and XML-DMM

6. CONCLUSION

In this paper, we investigated and reviewed two main concepts related to real-time data warehousing (RTDW): real-time ETL and data warehouse multidimensional modeling (DMM). We described several ETL tools by using common characteristics: mode, frequency, data volumes, and latency. In addition, we explained the advantages of semi-structured-DMM over structured-DMM. We highlighted the fact that the choice of DMM affects the real-time query execution in RTDW. To prove our assumption, we provided an implementation of RTDW framework for an example telecommunication organization. We compared relational-DMM with XML-DMM under two types of ETL loading methodologies: full-reloading and incremental-loading. Our experimental results show that the semi-structured XML-DMM has more capability for incorporating real-time data-updates from operational sources. Thus, it not only reduces the queries response time, but increases real-time business intelligence in an organization.

7. ACKNOWLEDGEMENTS

The authors would like to thank the Higher Education Commission (HEC) of Pakistan for supporting this research through grant No. 20-931/R&D/07.

8. REFERENCES

[1] Bruckner, R. M., List, B., and Schiefer, J. 2002. Striving towards near real-time data integration for data warehouses. In *Proc. of Int. Conf. DaWaK*, LNCS 2454, 317-326.

[2] Bouzeghoub. M., Fabret. F., and Matulovic. M. 1999. Modeling data warehouse refreshment process as a workflow application. In *Proc. of Int. Workshop on DWDM.*

[3] Simitsis, A., Vassiliadis, P., and Sellis, T. 2005. Optimizing ETL processes in data warehouses. In *Proc. of Int. Conf. on Data Eng.*, 564-575.

[4] Vassiliadis, P., Vagena, Z., Skiadopoulos, S., Karayannidis, N., and Sellis, T. 2001. ARKTOS: towards the modeling, design, control and execution of ETL processes. *J. of Inf. Systs.* 26, 8, 537-561.

[5] Italiano, C. and Ferreira, J. E. 2006. Synchronization options for data warehouse designs. *Computer* 39, 3, 53-57.

[6] Santos, R. J. and Bernardio, J. 2008. Real-time data warehouse loading methodology. In *Proc. of Int. Symposium on Database Eng. and Applications*, 49-58.

[7] Abrahiem, R.. 2007. A new generation of middleware solutions for a near-real-time data warehousing architecture. In *Proc. of IEEE Int. Conf. on Electro/Info. Tech.*, 192-197.

[8] *Going real-time for data warehousing and operational BI*, 2009. GoldenGate Software Inc., Available from http://datasolutions.searchdatamanagement.com/document;5132934/datamgmt-abstract.htm.

[9] Stonebraker, M. 2007. *The problem with one-size-fits all databases*. Article published in REDHAT Magazine, Available from:

http://magazine.redhat.com/2007/04/13/the-problem-with-one-size-fits-all-databases/ .

[10] Zhu, Y., An, L., and Liu, S. 2008. Data updating and query in real-time data warehouse system. In *Proc. of IEEE Int. Conf. on Comp. Sci. and Software Eng.,* 1295-1297.

[11] Li, C. and Wang, X. S. 1996. A data model for supporting on-line analytical processing. In *Proc. of Int. Conf. on Inf. and Knowledge Management*, 81-88.

[12] Agarwal, R., Gupta, A., and Sarawagi, S. 1997. Modeling multidimensional databases. *In Proc. of Int. Conf. on Data Eng.*, 232-243.

[13] Gyssens, M. and Lakshmanan, L. V. S. 1997. A foundation for multidimensional databases. *In Proc. of Int. Conf. Very Large Data Bases*, 106-115.

[14] Cabibbo, L. and Torlone, R. 1998. A logical approach to multidimensional databases. *In Proc. of . Int. Conf. on Extended Database Tech.*, LNCS 1377, 183-197.

[15] Datta, A. and Thomas, H. 1999. The cube data model: a conceptual model and algebra for on-line analytical processing in data warehouses. *Decision Support Systs.* 27, 3, 289-301.

[16] Pederson, T. B. and Jensen, C. S. 1999. Multidimensional data modeling for complex data. In *Proc. of Int. Conf. on Data Eng*., 336-345.

[17] Golfarelli, M., Rizzi, S., and Vrdoljak, B. 2001. Data warehouse design from XML sources. In *Proc. of ACM Int. Workshop on Data warehousing and OLAP*, 40-47.

[18] Hummer, W., Bauer, A., and Harde, G. 2003. XCube, XML for data warehouses. *In Proc. of ACM Int.Workshop Data warehousing and OLAP*, 33-40.

[19] Rusu, L. I., Rahayu, W., and Taniar, D. 2004. On building XML data warehouses, *Intelligent Data Eng. and Automated Learning* 3177, 293-299.

[20] Levy, A., Rajaraman, A., and Ordille, J. 1996. Querying heterogeneous information sources using source descriptions. In *Proc. of Int. Conf. on Very Large Data Bases*, 251-262.

[21] Cappellen, M., Cordewiner, W., and Innocenti, C. 2008. Data aggregation, heterogeneous data sources and streaming processing: how can XQuery help?. *IEEE Data Engineering Bulletin* 31, 4, 57-64.

[22] Boussaid, O., Darmont, J., Bentayeb, F., and Loudcher, S. 2008. Warehousing complex data from the web. *Int. J. of Web Eng. and Tech.* 4, 4, 408-433.

[23] Gupta, A., Harinarayan, V., and Quass, D. 1995. Aggregate-query processing in data warehousing environments. In *Proc. of Int. Conf. on Very Large Data Bases*, 358-369.

[24] Wang, M. and Iyer, B. 1997. Efficient roll-up and drill-down analysis in relational databases. In *SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, 39-43.

[25] Kim, W. 1982. On optimizing an sql-like nested query. *ACM Trans. Database Syst.* 7, 3, 443-469.

[26] Gray, J., Keulen, M., and Teubner, J. 2004. Accelrating XPath evaluation in any RDBMS. *ACM Trans. Database Syst.* 29, 91-131.

[27] Beyer, K., Chamberlin, D., Colby, L. S., Ozcan, F., Pirahesh, H., and Xu, Y. 2005. Extending XQuery for analytics. *In Proc. of Int. Conf. on Management of Data ACM SIGMOD*, 503-514.

[28] Engovatov, D. 2007. *XML query (Xquery) 1.1 requirements*. W3C Working Draft.

[29] Wu, H., Ling, T. W., Xu, L., and Bao, Z. 2009. Performing grouping and aggregate functions in XML queries. In *Proc. of Int. Conf. on World Wide Web*.

[30] Zhao, Y., Ma, T., and Liu, F. 2010. Research on index technology for group-by aggregation query in XML cube. *Inf. Tech. J.* 9, 10, 116-123.

[31] Jin, D., Tsuji, T., Tsuchida, T., and Higuchi, H. 2009. An incremental maintenance scheme of data cubes. *Databases Systs. for Advance Applications* LNCS 4947, 172-187.

[32] Kotidis, Y. and Roussopoulos, N. 1998. An alternative storage organization for ROLAP aggregate views based on cubetrees. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, 249-258.

[33] Baril. X. and Ballahsene, Z. 2006. Incremental method for XML view maintenance in case of non-monitored data sources. *Theory and Practice of computer Science* LNCS 3831, 148-157.

[34] Choi, B., Cong, G., Fan. W., and Viglas, S. D. 2008. Updating recursive XML views of relations, *J. of Comp. Sci. and Tech.* 23, 4, 516-537.

[35] Chamberlin, D., Florescu, and D., Robie, J. 2006. *Xquery Update Facility*, W3C, Avalaible from http://www.w3.org/Tr/xqupdate.

[36] Foster, J. N., Simeon, R. K, and Villard, J., L. 2008. An algebraic approach to Xquery view maintenance, *ACM SIGPLAN Workshop on Prog. Lang. Tech. for XML*.

[37] Jorg, T. and Dessloch, S. 2009. Formalizing ETL jobs for incremental loading of data warehouses, *Business Tech.and Web*, 57-64.

[38] Labio, W. and Garcia-Molina, H. 1996. Efficient snapshot differential algorithms in data warehousing. In *Proc. of Int. Conf. on Very Large Data Bases*.

[39] Jörg, T. and Dessloch, S. 2010. Near real-time data warehousing using state-of-the-art ETL tools. *Enabling Real-Time Business Intelligence*, *Lecture Notes in Business Inf. Processing* 41, 100-117.

[40] Shi, J., Bao, Y., Leng, and F., Yu, G. 2009. Priority-based balance scheduling in real-time data warehouse , In *Proc. of IEEE Int. Conf. on Hybrid Intelligent Systs*, 301-306.

[41] *Pentaho Open Source Business Intelligence*. Available from http://kettle.pentaho.com/

[42] *XQSharp: Xquery for the .Net Framework*, Available from http://www.xqsharp.com/xqsharp/