Active Datawarehouse Loading by Tool Based ETL Procedure

V.Mallikarjuna Reddy, Sanjay K Jena

Dept. Of Computer Science, National Institute of Technology Rourkela, India

vmreddy89@gmail.com, skjena@nitrkl.ac.in,

Abstract*: Over the years, DataWarehousing has gone through a number of evolutions from a relatively simple reporting database to sophisticated analytical applications such as analyzing customer lifetime values, market basket analyses, potentially defecting customers, fraud patterns, inventory churns, and so on. In all, though, these static sets of data could not give us the most current and recent changes of data necessary to act upon the results of Business Intelligence analyses. So, the traditional DataWarehouses are Static in nature. They are not showing any dynamics in their structure and content. In this paper we present a methodology on how to adapt data warehouse schemas and loading the fresh data with the help of the Tool based [12][13] Extraction, Transformation and Loading (ETL) procedure for Active Data Warehouse. To achieve the dynamic nature for the DataWarehouse, we are using Ricardo-Jorge [1] techniques such as table structure replication and query predicate restrictions for selecting data, to enable continuously loading data in the DataWarehouse with minimum impact in query execution time.*

Keywords**:** ActiveDataWarehouse (Active DW), Real-Time DataWarehouse (RTDW), Data Acquisition, code based ETL, Tool based ETL.

1. INTRODUCTION

A DataWarehouse provides historical data for analytical processing, decision making and for data mining tools. A DataWarehouse collects data from multiple heterogeneous operational source systems (OLTP– On-Line Transaction Processing) and stores summarized integrated business data in a central repository used by analytical applications (OLAP – On-Line Analytical Processing) with different user requirements.. To load the data into the DataWarehouse two types of techniques are used. Code based ETL and Tool based ETL. Here we are using the Tool based ETL technique to the continuous refreshments of the data in the DataWarehouse. The ETL processes are responsible for identifying and extracting the relevant data from the OLTP source systems, Transforming this data into a target table format, cleaning the data and conforming it into an adequate integrated format for updating the data area of the DataWarehouse and, finally, loading the final formatted data into its DataWarehouse.

Traditionally DataWarehouse stores the historical data. Here the data in the DataWarehouse are updated periodically typically in a monthly, weekly or daily. So, this type of DataWarehouse seems to be a Static DataWarehouses. Implying that its data is never up-to-date, for OLTP records saved between these updates are never recorded in the DataWarehouse. This implies that most recent operational records are not included in the DataWarehouse. So, for e-business, stock brokering, online telecommunications, the very recent data is very important to analyze the business process continuously, who rely on it to react in a near real-time manner, according to the new and most recent data captured by an organization's information system. This makes supporting near RTDW a critical issue for such applications. This DataWarehouses are also called as Active DataWarehouses.

The main characteristics of Active Data warehousing environment.[1,3] are

1. Active DataWarehousing means large volumes of data and, therefore, scalability becomes critical and absolutely required to provide the large amounts of detailed data needed to understand business events. Scalability also means the ability to support the concurrent queries described above. Therefore, the technology should provide the means to referee the contending actions and balance the conflicting needs of these various types of users.
2. The availability (and thus the reliability) is perhaps the most distinguishing characteristic of the technology to support both tactical and strategic queries. Traditional data warehouses do not usually have to be functional 24X7X365. As a result, the Active Data Warehouse must never go down, must never be inaccessible, or the business simply cannot operate.
3. Data Acquisition (ETL) must be performed much closer to the time a business event took place. Ideally, the acquisition mechanism will provide a continuous feed of new or changed data into the environment without blocking access to the very tables being updated.

Now a days the demand for fresh data in the DataWarehouse is increased. DataWarehouse refreshments are traditionally performed in an off-line fashion. This means that when this process is going on then the data in the DataWarehouse is not available to the OLAP users and applications to analyze the data. So, to overcome this Active DW's are coming into picture. Active DW [11] refers to a new trend where DataWarehouses are updated as frequently as possible, due to the high demands of users for fresh data.

For the continuous loading of the data into the Active DW several issues needs to be taken in to considerations.[2,11] (1) Operational OLTP systems are designed to meet well specified (short) response time

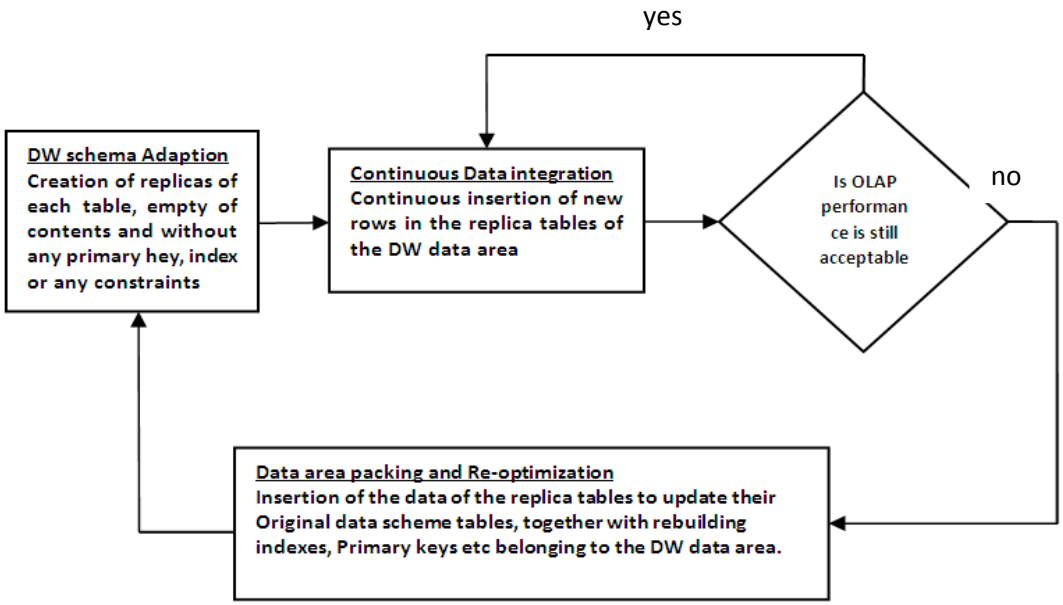

Fig 1: Architecture for proposed ETL procedure

requirements aiming for maximum system availability, which means that an Active DW scenario would have to cope with this in the overhead implied in those OLTP systems; (2) The tables existing in a DataWarehouse's database directly related with transactional records (commonly named as fact tables) are usually huge in size, and therefore, the addition of new data and consequent procedures such as index updating or referential integrity checks would certainly have impact in OLAP systems performance and data availability. Our work is focused on the DataWarehouse perspective, for that reason we present an efficient methodology for continuous data integration, performing the ETL process.

This paper presents a solution which enables the continuous data loading or continuous refreshments of the data from OLTP systems in to the Active DW, while allowing OLAP execution simultaneously, with minimum decrease of performance. Here we are designing the target tables, required Transformations and loading procedures with the help of Informatica tool to meet the requirements of the Active DataWarehouse. i.e. we are designing the ETL procedures to the continuous loading of the data into the DataWarehouse. And at the same time this updated data is always available to the OLAP users and Applications.

To Design the ETL procedures we are using Ricardo-Jorge methodology [1]. In this methodology based on the existing schema(s) of the DataWarehouse's database, we can create a replica (temporary tables) for each of its tables, empty of contents and without defining any type of indexes, primary keys or any other kind of restrictions or constraints. In this new schema, the replicated tables will receive and record the data from the OLTP source Systems, which will be continuously loaded. The fact that they are initially created empty of contents and not having any sort of constraints allows to considerably minimizing the consumption of time and resources which are necessary in the procedures inherent to data integration. Here the simple rule is to load the data into the smaller tables without following any indexing and referential integrity will take very less time for loading of the data into that replica tables. When the DataWarehouse's performance is considered as unacceptable by its users or administrators, the existing data within the replicated tables should serve for updating the original schema. After performing this update, the replicated tables are to be recreated empty of contents, regaining maximum performance. We also demonstrate how to adapt OLAP queries in the new schema, to take advantage of the most recent data which is integrated in real time.

2. RELATED WORK

Up to now research has mostly dealt with the periodical updates of the traditional DataWarehouse in off-line fashion [5,9]. And related literatures also present different tools and algorithms to populate the changed data in to the warehouse in an off-line fashion. In a different line of research, data streams [1, 2, 7,8] could possibly appear as a potential solution. However, research in data streams has focused on topics concerning the front-end, such as on-the-fly computation of queries without a systematic treatment of the issues raised at the back-end of a data warehouse [10]. Much of the recent work dedicated to RTDW is also focused on conceptual ETL modelling [4, 5, 11,12] lacking the presentation of exact specific extraction, transformation and loading algorithms along with their consequent OLTP and OLAP performance issues.

In [4] the researchers used code based ETL to the continuous loading of the data in to the Active DataWarehouse. So, they are writing corresponding procedures and triggers for the tables of the Active DataWarehouse in SQL and PL/SQL. They are written the coding for these entire tables with the considerations of the Indexing of the tables and all the constraints. In [5], the authors describe the ARKTOS ETL tool, which is capable of designing the practical ETL scenarios and mappings by providing explicit data cleansing, data Transformations, data aggregation and data scrubbing using a declarative language.

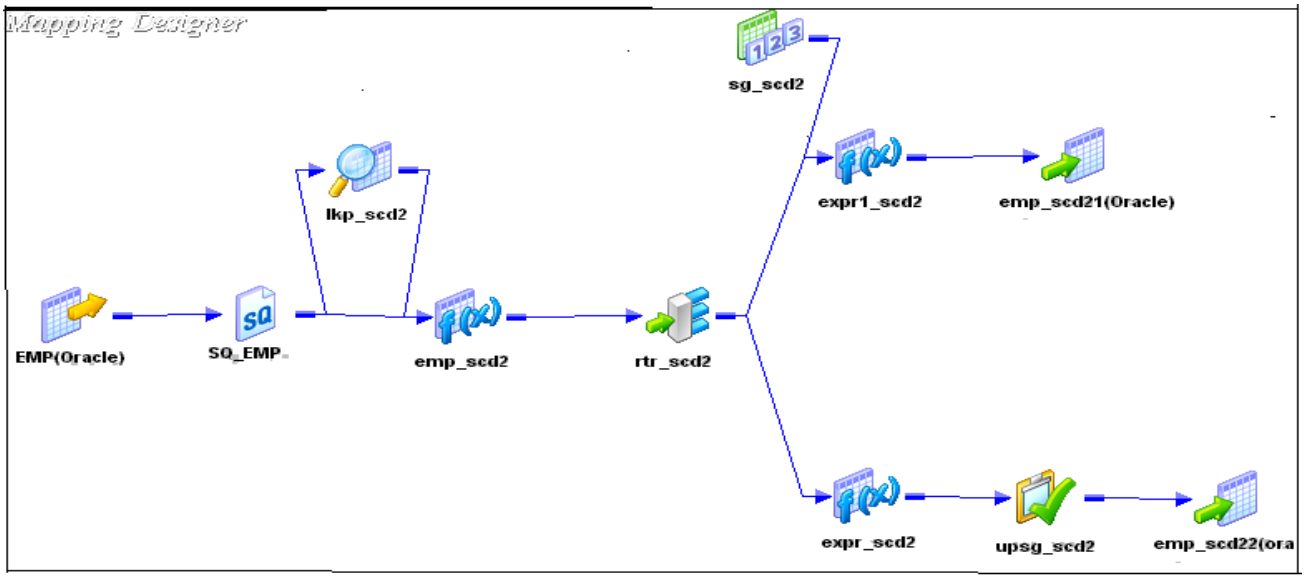

Fig.2. proposed ETL procedure mapping

3. ARCHITECTURE FOR PROPOSED ETL PROCEDURE

The architecture for the proposed ETL procedure is in Fig 1. From the DataWarehouse side, updating huge tables and related structures (such as indexes, materialized views and other referential integrities) and executing the OLAP queries simultaneously with continuous data integration is very difficult task. Ricardo-Jorge architecture [1] minimizes the processing time and workload required for these update processes. It also facilitates the DW off-line update. Because the data is already available within the data area and all OLTP data extraction and/or transformation routines have been executed during the continuous data integration. Furthermore, the data structure of the replicated tables is exactly the same as the original DataWarehouse schemas. This minimizes the time window for packing the data area of the DataWarehouse, since its update represents a one step process by resuming itself as a cut-and-paste action from the temporary tables to the original ones. Based on this architecture we are proposing the ETL procedures. The procedures and the description of the architecture are given in the following section.

4. PROPOSED ETL PROCEDURES

To load the data in to the Active Datawarehouse we are preparing the target replica tables and required transformations and loading procedures with the help of Informatica tool. The description is given in following sub sections.

4.1 Adapting the DataWarehouse Schema

Creation of an exact structural replica of all tables of the DataWarehouse that could eventually receive the new data. These tables (referred also as temporary tables) are to be created empty of contents, with no defined indexes, primary key, or constraints of any kind, including referential integrity also.

The main idea behind the creation of replica tables is, OLAP operations will take less time to execution. Because in the replica tables there is no indexing and no primary key constraints and this replica tables contains less amount of data. So, automatically an OLAP operation takes less time to execution. And OLAP must be changed to get the replica tables data. But in this paper we are concentrating on only the ETL procedures.

Here each replica table must contain one counter value. This counter value maintains the latest version of the record. And here this column is called as Surrogate key. The OLAP operations will take the latest version of the record based on this surrogate key.

4.2 ETL mappings

To refresh the DataWarehouse, the ETL application has extracted and transformed the OLTP data into the correct format for loading in to the DataWarehouse, if there is a change in the source system then that changed data is inserted as a new row in the correspondent replica table, filling the unique sequential identifier attribute with the auto incremented sequential number.

We are using Informatica tool for this ETL procedures. And this is Tool based ETL. We are taking Employee DataWarehouse as the example here. So, the corresponding mapping for this Employee DataWarehouse is given below. And the explanation for this mapping is in the following sections.

In the above proposed mapping [Fig 2] we are taking Employee DataWarehouse as an example. Here Emp as the source table And emp_scd21, emp_scd22 are the replica tables of the DataWarehouse Schemas. And here emp_scd21 table gets the new records from the source data. emp_scd22 will get the update records only. Here the following transformations are used in the proposed ETL procedure.

1) **Source Qualifier Transformation***:* This is of type an active transformation which allows us to read the data

```
SQL> set linesize 200;
SQL> select *from emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 17-DEC-80      10000                    20
      7499 ALLEN      SALESMAN         7698 20-FEB-81      12345        300         30
      7521 WARD       SALESMAN         7698 22-FEB-81       3333        500         30
      7566 JONES      MANAGER          7839 02-APR-81       2222                    20
      7654 MARTIN     SALESMAN         7698 28-SEP-81       1111       1400         30
      7698 BLAKE      MANAGER          7839 01-MAY-81      20000                    30
      7782 CLARK      MANAGER          7839 09-JUN-81      18000                    10
      7788 SCOTT      ANALYST          7566 19-APR-87       3000                    20
      7839 KING       PRESIDENT             17-NOV-81      25000                    10
      7844 TURNER     SALESMAN         7698 08-SEP-81       5555          0         30
      7876 ADAMS      CLERK            7788 23-MAY-87       7500                    20

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- --------- ---------- ---------- ----------
      7900 JAMES      CLERK            7698 03-DEC-81      15000                    30
      7902 FORD       ANALYST          7566 03-DEC-81       8765                    20
      7934 MILLER     CLERK            7782 23-JAN-82       1300                    10
      1234 raju       prog             7654 23-DEC-99      24000        100         20
      3456 Sandeep    prog             7899 23-DEC-09      20000        500         20

16 rows selected.

SQL>
```

Fig.3 Input source data for the proposed ETL prcedure

from OLTP Systems. Here the OLTP sources may be different databases or flat files (.txt, .doc). And this Source Qualifier Transformation will generate the corresponding SQL queries to get the data from different OLTP systems. And in this Transformation we can also change the default SQL queries as per our requirements. With this transformation, if the data is coming from same Databases then directly we can join that data

2) **Lookup Transformation***:* We can use a Lookup transformation in a mapping to look up data in a flat file or a relational table, view, or synonym. Use multiple Lookup transformations in a mapping. Here in this ETL procedure we are using Lookup transformation to lookup the target Emp table. So, here the source records are checked with the target records which are already available in the target table. The source Emp records are checked with the target Emp records. If the record is not there in the target table then it is inserted in the emp_scd21 target table with the version number as 1. And if the source record is already available in the target table then it is treated as updated record and the corresponding version number is increased by one and it is inserted in emp_scd22.

3) Expression Transformation*:* We can use the Expression transformation to calculate values in a single row before you write to the target. For example, you might need to adjust employee salaries, concatenate first and last names, or convert strings to numbers. Use the Expression transformation to perform any non-aggregate calculations. You can also use the Expression transformation to test conditional statements before you output the results to target tables or other transformations. Here we are using the expression transformation to create the two new fields. These are New and Update. These two attributes are helpful to find out whether the record is new or update with the help of Lookup transformation.

4) Router Transformation*:* A Router transformation is allow you to use a condition to test data. A Router transformation tests data for one or more conditions and gives you the option to route rows of data that do not meet any of the conditions to a default output group. So, here the Router Transformation takes the conditional attributes as New and Update which are generated in Expression transformation. So, the Router transformation checks the condition and if it is new it will route the source record to the emp_scd21 target table. And if it is Update then it will route the updated record to the emp_scd22 target table.

5)SequenceGenerator Transformation*:* The Sequence Generator transformation generates numeric values. Use the Sequence Generator to create unique primary key values, replace missing primary keys, or cycle through a sequential range of numbers. Here we are using Sequence Generator Transformation to generate the version number of the source records. It will generate the version numbers from 1. Here if the source record is new then the version number for that record is 1. And if the record is updated record then the version number for that record is incremented it's old version number by one. And this acts as a surrogate key.

6) Update Strategy Transformation: This is of type Active Transformation which flags the records for insert, update, delete and reject data driven operations. Here if the source record is updated record then directly with the help of this transformation the corresponding source record is updated in the target table emp_scd22.

After creating this mapping we will create a session variable for this mapping. With the help of the session variable the mappings are executed and the source data is loaded into the target tables. And the session variable contains scheduling properties. In the scheduling properties we are giving the scheduling time as continuous. So, this session variable is executed continuously and the source data is loaded continuously in the Active DataWarehouse. So, the fresh data is also available to the OLAP users and applications in the Active DataWarehouse.

```
SQL> select *from scd2_router;

     DIMKEY      EMPNO ENAME      SAL    VERSION
---------- ---------- ---------- ---------- ----------
       3200       7369 SMITH      10000          0
       3300       7499 ALLEN      12345          0
       3400       7521 WARD       31111          0
       3500       7566 JONES       2222          0
       3600       7654 MARTIN      1111          0
       3700       7698 BLAKE      20000          0
       3800       7782 CLARK      18000          0
       3900       7788 SCOTT      23400          0
       4000       7839 KING       26500          0
       4100       7844 TURNER     18900          0
       4200       7876 ADAMS       7500          0

     DIMKEY      EMPNO ENAME      SAL    VERSION
---------- ---------- ---------- ---------- ----------
       4300       7900 JAMES      15000          0
       4400       7902 FORD        8765          0
       4500       7934 MILLER      1300          0
       4600       1234 raju       24000          0
       4700       3456 Sandeep    20000          0
       3401       7521 WARD       32000          1
       3601       7654 MARTIN     28600          1
       4601       1234 raju       19500          1
       3602       7654 MARTIN     29500          2

20 rows selected.
```

Fig.4 The fresh data after change in the source data

5. PACKING AND RE-OPTIMIZING THE DATAWAREHOUSE

Since the changed data is being integrated with replica tables that do not have access optimization of any kind that could speedup querying, such as indexing or hashing. So, after some time the size of the replica tables also increased. At that time the performance of the OLAP queries is not good without indexing or hashing for analysis. Due to the size of physically occupied space, after a certain number of insertions the performance becomes too poor to consider as acceptable. To regain the performance of the OLAP queries it is necessary to execute a pack routine which will update the original DataWarehouse schema tables using the records in the replica tables, and recreates these replica tables empty of contents, along with rebuilding the original table's indexes, referential integrities and all other constraints, so that maximum speed is obtained once more.

For updating the original DataWarehouse tables, the rows in the replica tables should be aggregated according to the original table's primary keys, maintaining the rows with highest unique counter attribute value represents the most recent records. The time needed for executing these update procedures represents the only period of time in which the DataWarehouse is unavailable to OLAP queries and end users, for they need to be executed exclusively. The appropriate moment for doing this may be determined by the Database Administrator, or automatically, taking under consideration parameters such as a determined number of records in the temporary tables, the amount of physically occupied space, or yet a predefined period of time. the amount of time it takes away from all user availability, This Packing and Re-optimizing will be done only the OLAP operations performance is not acceptable with the replica tables.

6. OLAP QUERY ADAPTION

Consider the following old OLAP query, for calculating the total salary of the entire employees group by deptno.

```
SELECT deptno,sum(sal) AS all_emp_sal
FROM emp GROUP BY deptno.
```

The above OLAP query will get the total salary of all employees from the traditional DataWarehouse. This query will not get the fresh data.

So, here to get the fresh data and old data for analysis all the OLAP queries must be modified as the resultant tables must get the data from the original tables as well as replica tables. Simply the queries will be changed as the FROM clause should join all the rows from the required original tables and replica tables with relevant data.

The above query will be changed to

```
SELECT deptno,sum(sal) AS all_emp_sal
FROM (SELECT deptno,sal from emp UNION
ALL (SELECT st_sal from temp_emp) GROUP
BY deptno;
```

Here temp_emp is the replica table for original emp table. So, OLAP query getting the old as well as fresh data also.

And here if the OLAP users want only the fresh data then they can get the fresh data from the replica tables only. Here replica tables are small in size. This minimizes the CPU utilization, memory and I/O costs involved in most recent data query processing.

7. RESULT

The source table [Fig 3] contains the Employee data. Suppose in the source table data there is a change in the salary of the employee Martin. If In this way all the employee's salary's are changed frequently then all these changes are captured by the target DataWarehouse with the help of the proposed ETL mapping. In this way if there is a continuous change in the source Database then all these changes are captured by the Target DataWarehouse with the zero latency. And here to get the fresh data continuously the session variable for that mapping is running continuously with the help of schedulers in Informatica. And the target DataWarehouse identifies the fresh data with the version number. In this example Martin's salary is changed so the version number is increased to 1 [Fig.4]. later once again his salary is increased. So, version number is increased to 2.Now the fresh data is with version number 2. And this is recent data. The rows with Version number 0 and 1 are identified as old data.

In this way in e-business we can get the fresh data. With this fresh data all the OLAP users can analyze their business with fresh data as well as old data.

8. CONCLUSION AND FUTURE WORK

This paper presents a GUI based ETL procedure to the continuous loading of the data in the Active DataWarehouse by enabling continuous data integration while minimizing impact in query execution on the user end of the DataWarehouse. This is achieved by data structure replication and adapting query instructions in order to take advantage of the new real time DataWarehousing schemas.

And for designing the conceptual mappings for this Active Data Warehouse in code based ETL takes long time, execution of the this code based ETL will consume more time but in this GUI based ETL design for the execution of the mappings the processor takes very less time compared to code based ETL. In code based ETL we need to develop the procedures, functions and triggers, this consumes the maximum time but in tool based ETL we don't need to prepare the procedures, functions and triggers. Simply we are preparing the mappings and required transformations. The SQL code for all these mappings will be generated automatically by the Informatica Tool. Further if any changes are required in this tool generated code, we can open that code we can do the required modifications. And here we can get the fresh data continuously from the replica tables which are smaller in size. So, automatically the performance of the OLAP queries and Applications is improved. And with the help of schedulers we can run the session variables continuously. So, the fresh data is loaded and is available for OLAP users and applications.

As future work we intend to develop an ETL tool especially for Active DW which will integrate this Ricardo-Jorge architecture with extraction and transformation routines for the OLTP systems.

REFERENCES

[1] Ricardo Jorge Santos and Jorge Bernardino," *Real-Time DataWarehouse Loading Methodology*", IDEAs'o8, ACM 978-1-60158-188-0/08/09,2008.

[2] R. M. Bruckner, B. List, and J. Schiefer,*"Striving Towards Near Real-Time Data Integration for DataWarehouse*", International Conference on Data Warehousing and Knowledge Discovery (DAWAK), 2002.

[3] W. H. Inmon, R. H. Terdeman, J. Norris-Montanari, and D.Meers, "*Data Warehousing for E-Business"*, J. Wiley &Sons, 2001.

[4] A. Simitsis, P. Vassiliadis and T. Sellis, *"Optimizing ETL Processes in Data Warehouses"*, International Conference on Data Engineering (ICDE), 2005.

[5] P.Vassiliadis Z. Vagena, S. Skiadopoulos, N. Karayannidis, and T. Sellis, *"ARKTOS: Towards the Modelling, Design, Control and Execution of ETL Processes"*, Information Systems, Vol. 26(8),2001.

[6] R. M. Bruckner, and A. M. Tjoa, "*Capturing Delays and Valid Times in Data Warehouses – Towards Timely Consistent Analyses", Journal of Intelligent Information Systems (JIIS)*, 19:2, pp. 169-190, 2002.

[7] Informatica corporation, www.informatica.com

[8] Araque, F., Samos, " *Data warehouse refreshment maintaining temporal consistency"*, 5th International Conference on Enterprise Information Systems, ICEIS´03. Angers, France, 23-26, April 2003.

[9] Ramamritham, R. Sivasankaran, J. A. Stankovic, D. T. Towsley and M. Xiong, "*Integrating Temporal, Real-Time, and Active Databases*", ACM Sigmod Record, Vol. 25. No. 1, March 1996, pp. 8-12., 2002.

[10] Oracle Corp. Oracle9i™ Warehouse Builder User's Guide, Release 9.0.2. November 2001.

[11] Paton, N.W., Diaz, "*Active Database Systems*". ACM Computing Surveys (CSUR) 31(1), pp63-103, 1999.

[12] Karakasidis, A., Vassiliadis and P., Pitoura, *" ETL Queues for Active Data Ware- housing."*, Proceedings of the 2*nd* International Workshop on Information Quality in Information Systems (IQIS'2005), New York, USA, ACM Press 28-39,2005.

[13] Dr. R.S.Chhillar, BarjeshKochar, "*Extraction Transformation Loading-A Road to DataWarehouse*", 2nd National Conference Mathematical Techniques: Emerging Paradigms for Electronics and IT Industries, sep 26-28,2008.

[14] Thomos Jorg and Stefan Debloch, " *Towards generating ETL process for Incremental Loading*", IDEAS08 2008, September 10-12, Coimbra [Portugal].

[15] J. Yang, and J. Widom, 2001. *"Incremental Computationand Maintenance of Temporal Aggregates"*, 17th Intern. Conference on Data Engineering (ICDE).

[16] T. Zurek, and K. Kreplin, 2001. *"SAP Business Information Warehouse – From Data Warehousing to an E-Business Platform"*, 17th International Conference on Data Engineering (ICDE).

[17] D. Theodoratus, and M. Bouzeghoub, 1999. *"Data Currency Quality Factors in Data Warehouse Design"*, International Workshop on the Design and Management of Data Warehouses (DMDW).

[18] C. Thomsen, T. B. Pedersen, W. Lehner. RiTE: "*Providing On-Demand Data for Right-Time Data Warehousing*". In ICDE, pp. 456-465, 2008.

[19] Labio, W. and Yang, J. "*Performance Issues in Incremental Warehouse Maintenance"*. Techincal Report, Stanford University, 1999.