Investigating Data Warehouse Decision Support Using Near Real-Time Data Updation

Research Dissertation Submitted By,

Mr. Iftikhar Ali Khan
(400-FBAS / MSCS / S08)

Supervisor

Mr. Muhammad Imran Saeed
Assistant Professor,
Department of Computer Science,
International Islamic University, Islamabad.

Co-Supervisor

Mr. Shahbaz Ahmed Khan
Assistant Professor,
Department of Computer Science,
International Islamic University, Islamabad.

Department of Computer Science,

Faculty of Basic & Applied Sciences,

International Islamic University, Islamabad.

(2010)

1- Data warehousing

2- Database management

With the Name of

Allah,

The most merciful and compassionate the most gracious and beneficent whose help and guidance we always solicit at every step, and every moment.

A dissertation submitted to the

Department of Computer Science,

International Islamic University, Islamabad

as a partial fulfillment of the requirements

for the award of the degree of

Masters of Science in Computer Science.

DEDICATION

To a Person who is
"The Rehmat" for the entire Universe,
&
To My Parents who are like cool shade in the noontide of my life,
particularly to my mother whose hands get tired of praying for my success,
&
To those who pray for me and encouraged me throughout
my educational career.
&
To whom I love and respect.

ACKNOWLEDGMENT

I offer heartiest "DROOD-O-SALAM" to Holly Prophet MUHAMMAD (PEACE BE UPON HIM). I am grateful to almighty ALLAH who is merciful and beneficent, and who enable me to work on this research successfully. Accomplishment of a research thesis requires the help of many people who steer, guide, give confidence and help you. I have also been supported and guided by many people who were always there to help me out in the time of need. First of all I would like to express my sincere gratitude to my supervisor, ***Mr. Muhammad Imran Saeed*** and my co-supervisor, ***Mr. Shahbaz Ahmed Khan*** for their esteemed supervision, encouragement and guidance for successful completion of this thesis. I am also thankful to my friends who always encouraged me to complete this research work. I am heartedly grateful to my parents for their gracious, unconditional support and encouragement throughout my study.

Mr. Iftikhar Ali Khan

Declaration

I hereby declare that this thesis, neither as a whole nor as a part thereof has been copied out from any source. It is further declared that no portion of the work presented in this report has been submitted in support of any application for any other degree or qualification of this or any other university or institute of learning.

Mr. Iftikhar Ali Khan

Abstract

The most precious resource of an organization is knowledge and such knowledge could be achieved if fresh data is available at right time. So time is the most significant part in human existence which is used for finding most wanted data from foundation area in a lesser amount of time. A good decision is based on knowledge which is achieved from most fresh data, so Data Warehouse (DWH) must have most recent data but traditionally DWH is being refreshed in an off-line fashion. Nowadays, naturally users do not accept that a DWH is to be updated in an off-line fashion and prefer to have most fresh data to make their decision process quick and fast at right time. So the most important issue which is being faced by many organizations is the data updating.

For the sake of accommodation the high demands of user that the DWH must be updated in an on-line fashion and fresh data must be available. So for such purpose, in this thesis a technique for Extraction, Transformation and Loading (ETL) activities is being implemented which feeds the data from foundation area to targeted area because there is a demand for real-time ELT tools. The objective or goal of this thesis is to implement a technique for extraction of exact volume of data at the right time to the right palace according to the business rules defined to achieve the business success.

Table of Contents

Ch. No **Contents** **Page No**

Ch. No **Contents** **Page No**

List of Figures

Fig. No **Contents** **Page No**

Fig. No **Contents** **Page No**

List of Tables

Table. No **Contents** **Page No**

Abbreviations

KSE	Karachi Stock Exchange
ETL	Extraction, Transformation, Loading
DWH	Data Warehouse
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
DM	Data Mart
DSA	Data Staging Area
ADSA	Active Data Staging Area
DSS	Decision Support Systems
TDS	Tactical Decision Support
RTE	Real-Time Enterprise
DBMS	Database Management System
CDC	Changed Data Capture
RTDWH	Real Time Data Warehouse
ADWH	Active Data Warehouse
MDB	Message Driven Bean
DBQ	Database Queue
RDI	Real-Time Data Integration
SOA	Service Oriented Architecture
DTS	Data Transformation Service
ERP	Enterprise Resource
CE	Content Enrichment
CBR	Content Based Router
ODS	Operational Data Source
MQM	Message Queue Manager
MV	Materialized View
AT	Auxiliary Table
FIFS	First in First Schedule
OWB	Oracle Warehouse Builder
CTF	Capture Conversion Flow
MD	Master Data
MB	Mega Byte

Chapter #1
Introduction

1.1. Introduction

Data warehouse has emerged as a key platform as a support system for tactical and strategic decisions in business (e/m-business) environment. The active use of this technology has been seen in auction sites and intelligent automated stock exchange brokerage systems. The data warehouse has resulted in many organizational benefits, including providing “a single version of the truth,” better data analysis and time savings for users, reductions in head count, facilitation of the development of new applications, better data, and support for customer-focused business strategies. [35].

The data warehouse technology has become the core technology for making intelligent history based predictive decisions as data is the central repository is acquired from diverse operational data and information foundations. As the technology is maturing and need of business organizations and applications find data warehousing ever more beneficial and the main foundation of accurate multidimensional information extraction. One can execute all sorts of investigation based on cross functional data, and get information of day-to-day business operations more recurrently. The data freshness moving from a few hours to a few minutes will be more likely to meet client expectations as it empowers them for spontaneous decision making on the most recently active data. Unfortunately, the in use decision support systems do not make available the low latencies desirable for decision making in this rapidly altering (highly quasi fractal) heterogeneous environment [15]. Consequently the client demand for highly up-to-date information in data warehouses is greater than ever. At present, the majority of data warehouses are set up to offer us static snapshots of data, and as recent as ten to sixteen hours ago data, which is not often satisfactory to react to current scenario, specially when the decisions are being made in highly volatile situations [20, 24] e.g. a buying selling decision in KSE-100 index of0-3\ Pakistan. The data warehouses need to do a giant leap from nightly batch runs to several-times-a-day runs. But there are a few exceptions as a handful of companies such as 3M Corporation have already started doing data warehouse refreshes several times a day [20].

As it is obvious that the vital feature of data warehousing is to get hold of data and reflect it in the warehouse the instant data alteration/insertion takes place in the foundation to facilitate business applications' expectation. To minimize latency from the data warehousing systems is the main concern at a growing number of enterprises [17] as this can give a great boost in decision aiding capabilities of the client.

Now a days data warehouses have a startling improvement in requirement rate for data refresh ability and much more sophisticated data acquisition mechanisms [24] have come into consideration. The tools use in the current scenario also called ETL tools (extraction-transformation-load) are far outdated to achieve the current expected scenario.

For large enterprises there is a great quantity of data generated in the outfitted databases. The data warehouse is intended for holding hefty volumes of data with chronological and up to date data sets. Conservative ETL gear takes longer time to do ETL alterations when numerous million records are retrieved from compound foundation tables to stack into a warehouse.

Now a day's heavy volume of data (more than hundreds of gigabytes or more than that) is being loaded to data warehouse and it is the requirement of current time to have more and fresher data in such data warehouse. But how a data warehouse has fresh data, this is the most time consuming process which is not acceptable by most of customer.

1.2. Data Warehouse Systems

Active DWH is the key issue in database. Big business communities have a great interest to access more and more bright data for taking good decision on right time to achieve success in their business objectives.

Organization have been investing for last 40 years for computer system to computerize the method of commerce and through this commerce method, organization have achieved the confidence and offer very pleasant services to their costumer. In such last 40 years, the amount of data is increasing which is very helpful for decision-making.

The Bill Inmon pioneer of data warehousing in 1990 defined the term data warehousing in the following manner: "*A warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process*" [6]. Due to the repository of historical data, DWH is the smartest process for any organization because it gives much support for taking decision to enhance their business. Basically DWH is not designed for transaction processing but for analysis and query because DWH contain a huge amount of historical data. DWH has the power of ETL, OLAP and Data Mining which are being used for assembling data and provide sufficient knowledge to the industry users Data warehouse has a great role for supporting the process of decision-making which provide good knowledge to the users which alternatively advance the trade.

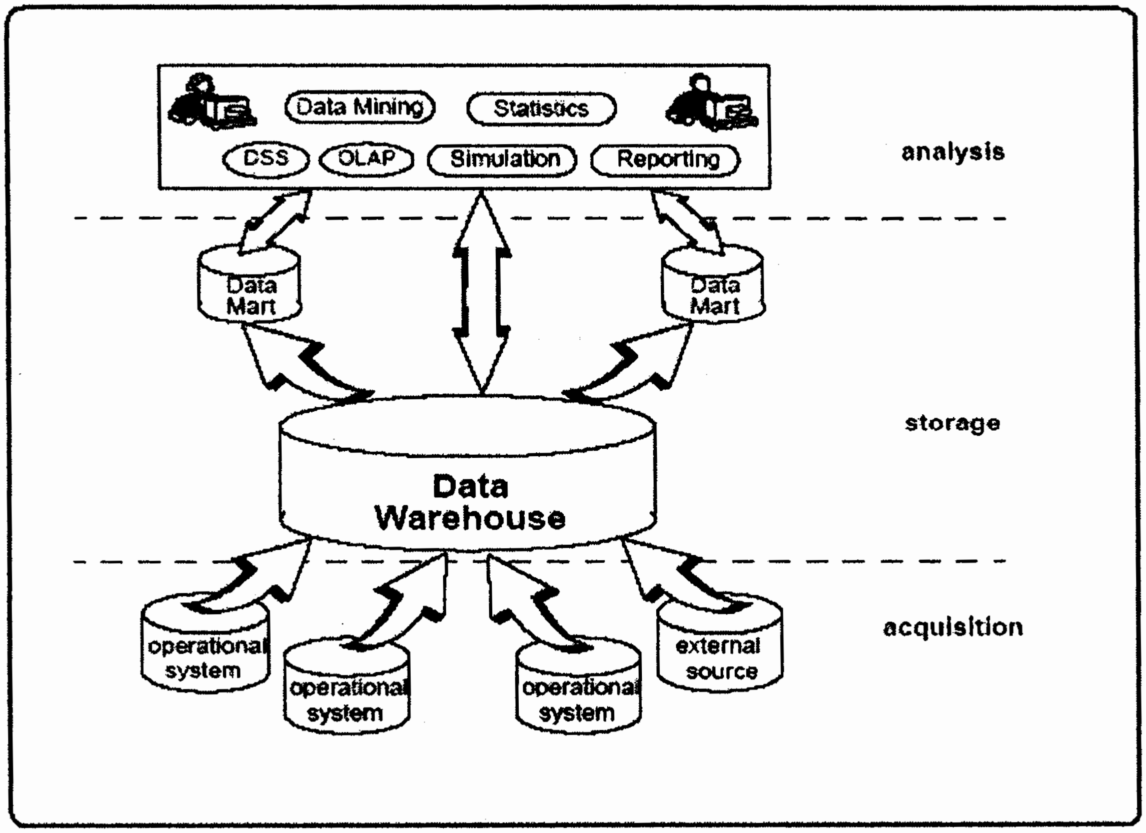


Figure 1.1 Common DWH System Architecture [44]

Figure 1-1 showing the data warehouse environment [Chaudhuri & Dayal, 1997], [Gatziu & Vavouras, 1999]. According to the above architecture, the phases are as:

- In acquisition phase data is being loaded to the data warehouse from different operational sources.
- Particular modeling concepts and different storage structures are being used for storing integrated data in Data warehouse.
- At last, with the help of different gear, users get data from DWH.

1.3. Virtual Data Warehouses

In virtual DWH there is no separate physical place for storing and maintaining the data [44].

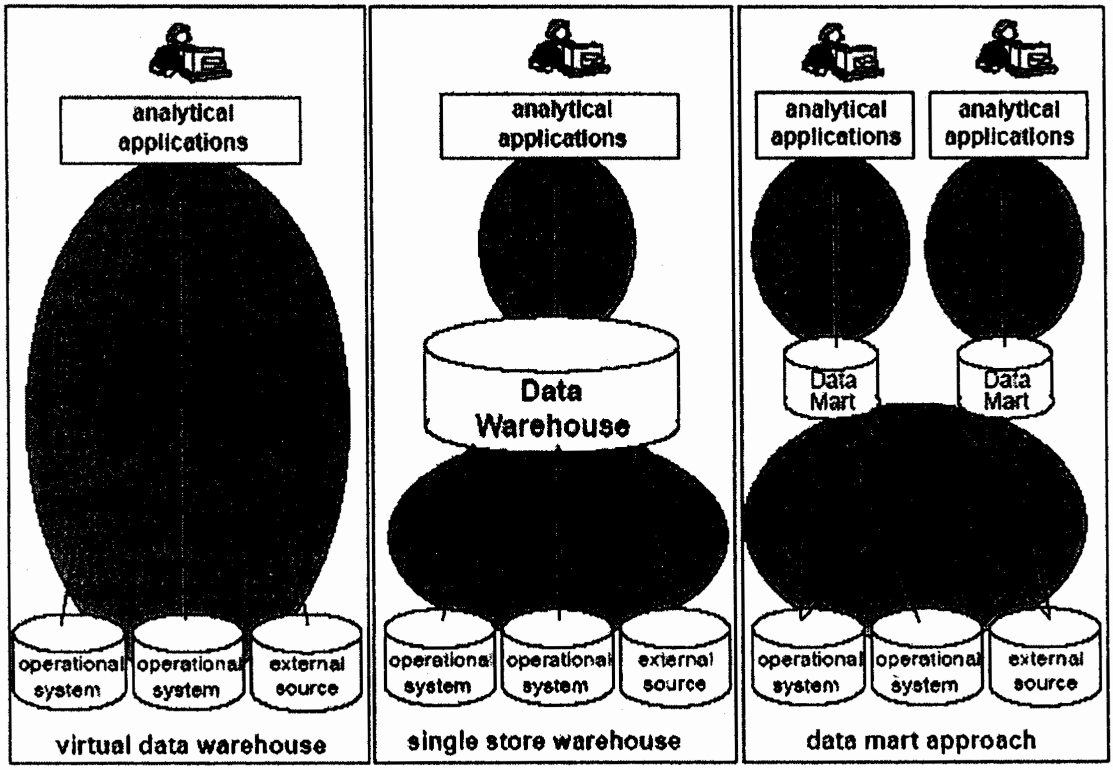


Figure 1.2 Alternative DWH Architecture [44]

1.4. Single Store Warehouse

In single store warehouse there is a separate physical place for storing and maintaining data [44]

1.5. Individual Data Marts

The reason of several and autonomous data marts is to store the extracted data directly from dissimilar functioning systems. Such data marts are being developed according to the particular user cluster [44]

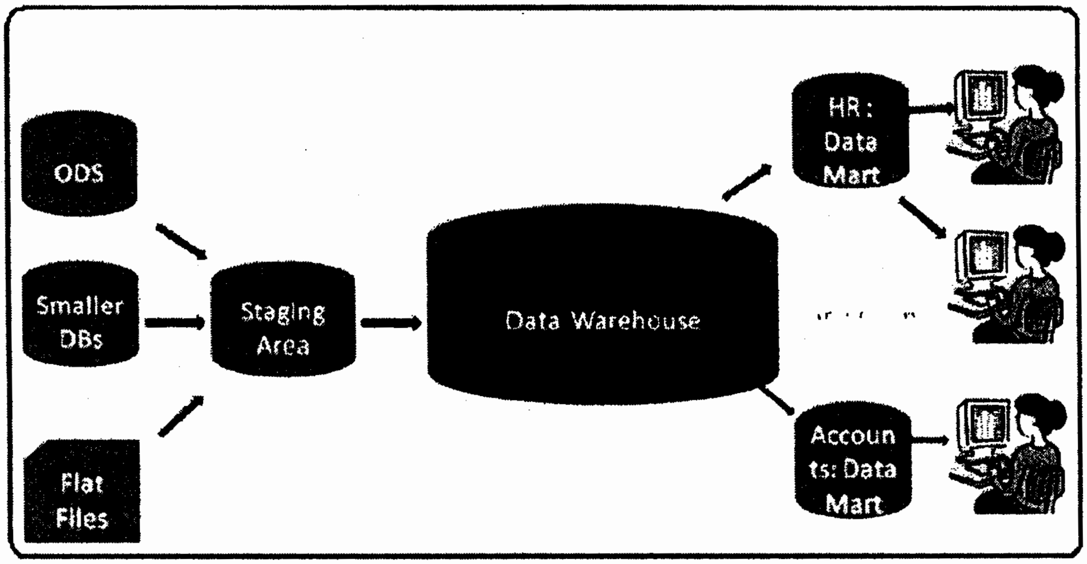


Figure 1.3 DWH Architecture with DM [44]

1.6. Data Warehouse Architecture (with a Staging Area)

It is necessary to clean the operational data before sending it to the DWH for processing. The process of extracting, transforming, and loading data into the warehouse is considered the most crucial and time-consuming part of data warehousing. Because of the crucial complexity involved in the process of updating the D/W house from the source databases (servers).

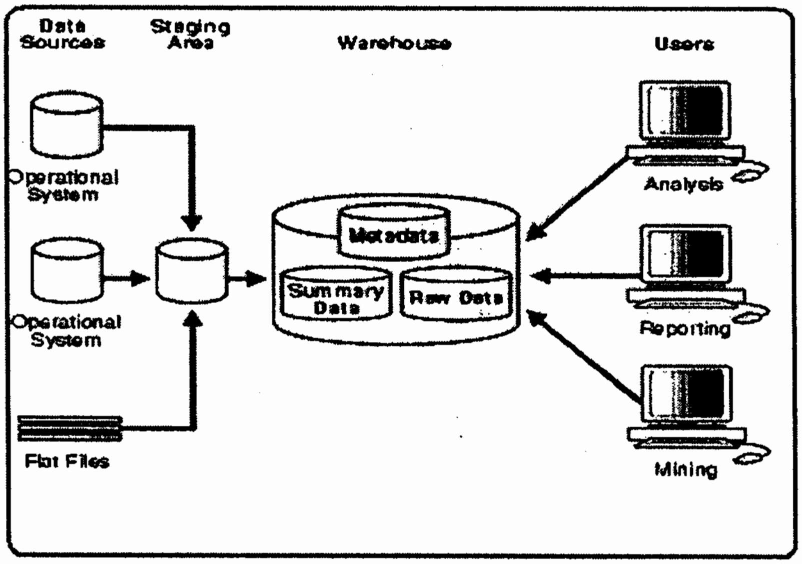


Figure 1.4 Architecture of DWH with Staging Area [44]

D/W house usually have a particular dedicated area called a Data Staging Area (DSA) where three major steps are performed below here:

a) Retrieving/Extracting data from a source,

b) Transforming it,

c) And the Final one is loading into the D/W house [11].

1.7. ETL Process (Extract, Transform, Load)

ETL is playing an extremely important role in data warehouse's establishment and the maintenance process; it is the bridge between the data source and the data warehouse[1]

Figure 1-5 showing the most central element of DWH architecture which gets the data from foundation database to DWH. The ETL software extracts data, transforms values of inconsistent data, cleanses "bad" data, filters data and loads data into a target database [39].

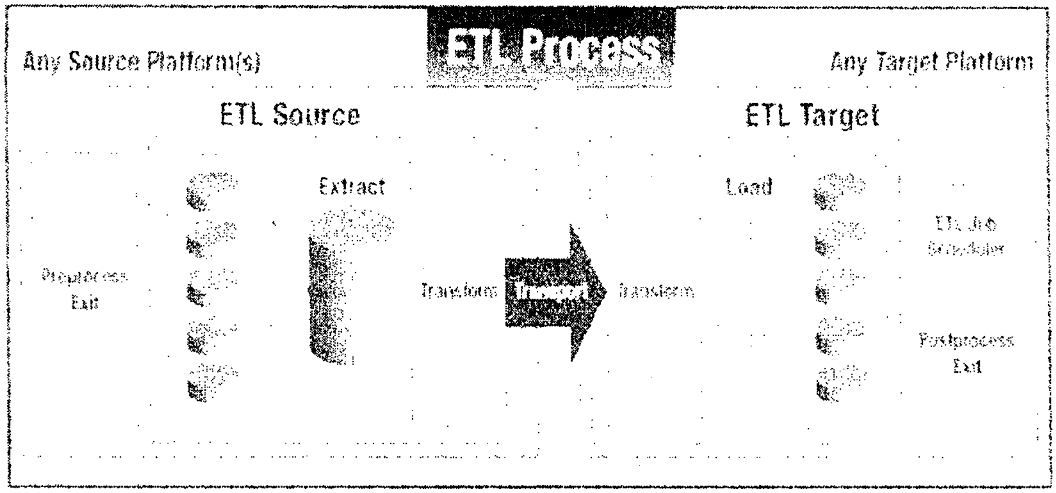


Figure 1.5 ETL Process

There are different pros and cons to maintaining (DSA) in the target D/W house as opposed to outside the D/W house:

1. Directly load tables without renormalization (de-normalized for performance or normalized to create correct record structures) so as to avoid semantic disintegrity. Semantic disintegrity occurs when a user submits a query and (after parsing-fetch-execute) reply an answer, but the answer is not the answer to the question they believe that they asked [8]. This will allow a capture of exact production records in the staging area without any structure change.

2. By placing the (DSA) at the D/W house, the D/W housing operations do not load/burden the source. This way, the source is completely detached from the active data warehousing process [25].

3. Using the D/W house as a staging area (DSA) helps avoid a three-tier architecture and save the setup and maintenance, and enhancement cost, since an extra server, apart from the one used for the warehouse, does not need to be engaged and administered.

4. Due to reasons of semantic or structural incompatibilities, primarily/ intermediate processing stage has to take place, in order to transform and clean the data. Alexandros

Karakasidis, Panos Vassiliadis, and Evaggelia Pitoura [25] refer to this part of the system as the Active Data Staging Area (ADSA). Once ready for loading, the data from the auxiliary layer is loaded at the warehouse, through a set of PL/SQL Blocks or Stored Procedures (Stored Procecure: don't return values).

5. If you deploy a D/W house layer then you can load "staging tables" in the D/W house and perform all or additional transformations on the data and place it in a set of results tables. This also provides a residual benefit in that a minority of users may not want the "transformed data." They may want the actual raw data for various reasons [9].
6. When transformation is crucial or highly complex and requires a multi-step process, transformation based on staging table data makes it easy do aggregation, summation, and other computations.

1.8. Extraction Methods in Data Warehouses

Extraction process is highly time consuming job which puts all load of extraction on source systems. There is no way to improve the extracting method. There are two types of Logical extraction and two types of Physical extraction [40].

1.9. Full & Incremental Extraction (Logical Extraction)

Whole data that is available is being extracted and sent to the DWH. The source will have to provide all the data because there are no records / list of changes to the data at source side (for example, timestamps). The example of full extraction is export to a file.

With the help of timestamp, all the data that is only changed will be extracted and after comparing with the old data, both new and change data will be loaded to the DWH. This is known as incremental extraction.

An important consideration for extraction is incremental extraction, also called Change Data Capture [40].

After choosing the logical extraction technique, capabilities and limitation on the source system, now the data can be extracted with the help of two ways

1.10. Online & Offline Extraction (Physical Extraction)

In an online extraction method the data is extracted directly from the foundation area itself with the help of extraction process which connected straight to the foundation area (for example, snapshot logs or change tables) while in an offline extraction data is extracted from the outside of the foundation area. (For example flat files or dump files) [40].

1.11. Transformation Flow

Transformation of data is very difficult from time processing point of view. ETL is very expensive part [40].

1.11.1. Multistage Data Transformation

In multistage data transformation, for each transformation a separate SQL command and a separate table are being created to store incoming results for each step. For example tables `new_sales_step1` & `new_sales_step2` [40].

This figure illustrates four steps in a multistage data transformation:

- Load into staging table
- Validate customer keys (in this case, by using the customer dimension table)
- Convert source product keys to warehouse keys
- Insert into the warehouse table

Before loading, start with flat files. After loading, the data is in the table.

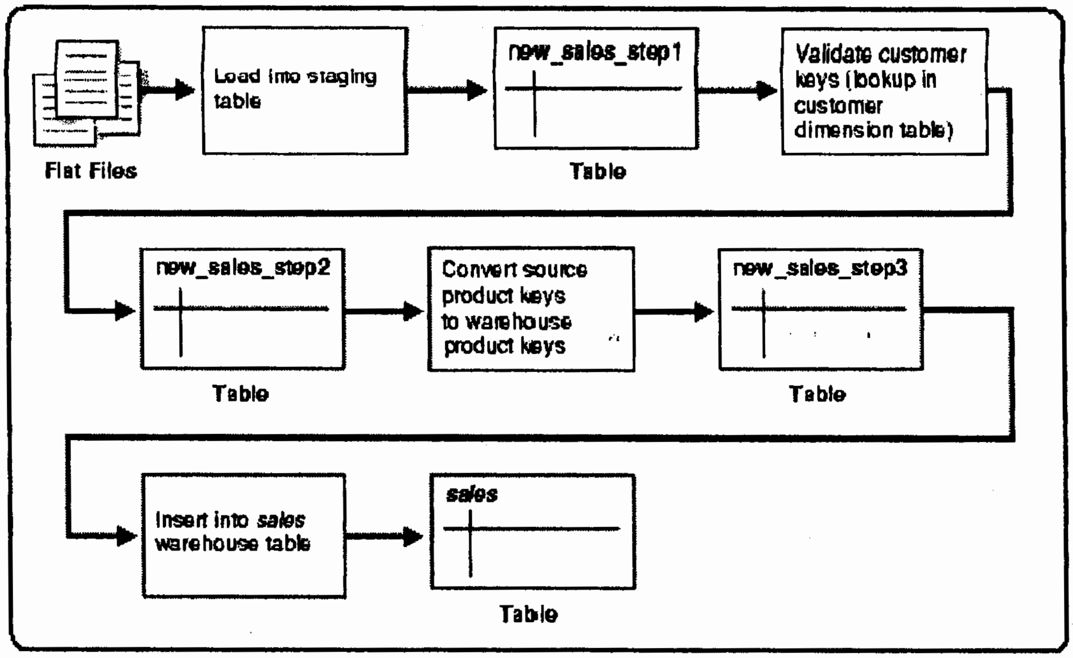

Flat Files
Load into staging table
new_sales_step1
Table
Validate customer keys (lookup in customer dimension table)
new_sales_step2
Table
Convert source product keys to warehouse product keys
new_sales_step3
Table
Insert into *sales* warehouse table
sales
Table

Figure 1.6 Multistage Data Transformation [40]

1.11.2. Pipelined Data Transformation

This figure illustrates the steps in a pipelined data transformation:

- Load external table with flat files
- Validate customer keys (lookup in customer dimension table)
- Convert source product keys to warehouse product keys
- Insert source product keys to warehouse product keys

Before loading, start with flat files. After loading, the data is in the table. The difference between pipelined data transformation and multistage data transformation is that pipelined transformation occurs in database tables [40].

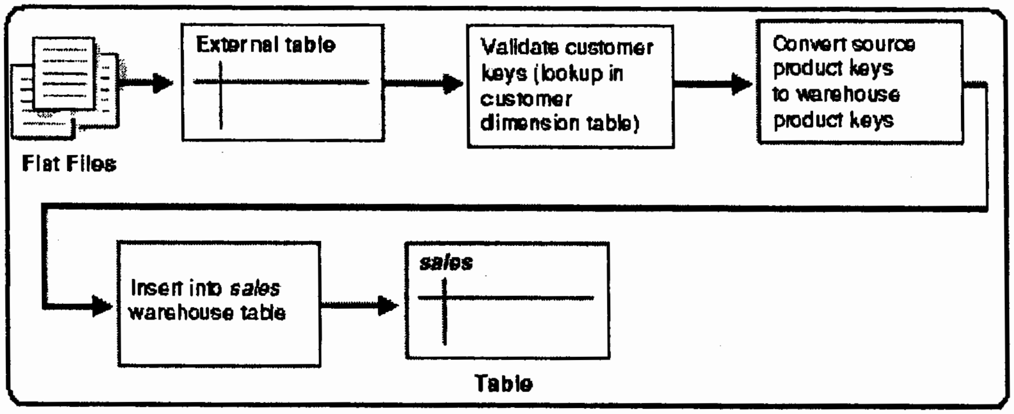

Flat Files
External table
Validate customer keys (lookup in customer dimension table)
Convert source product keys to warehouse product keys
Insert into *sales* warehouse table
sales
Table

Figure 1.7 Pipelined Data Transformation [40]

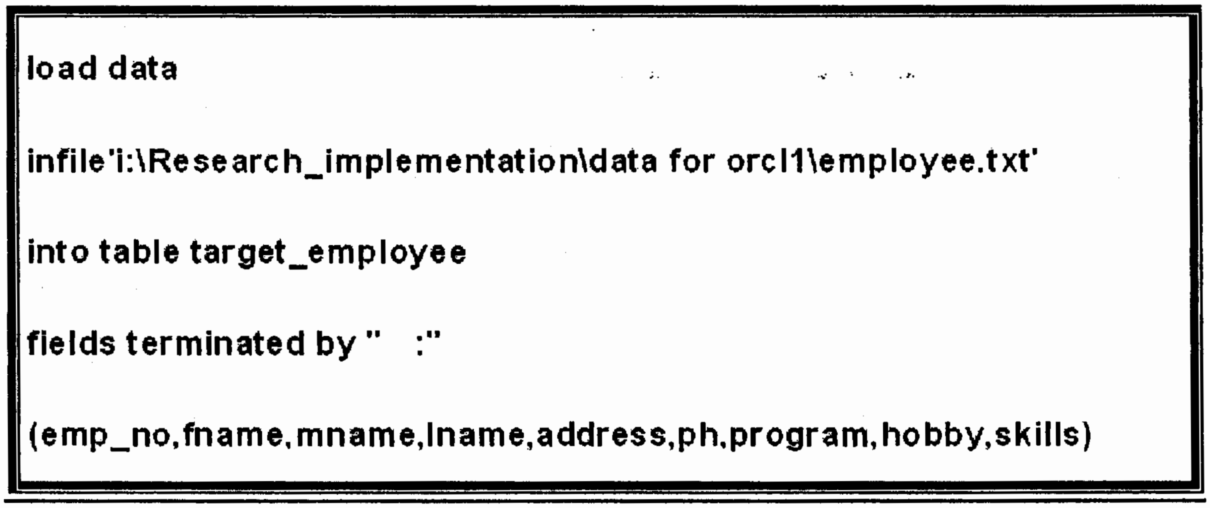
1.12. Loading Mechanisms

- Loading Data Warehouse with SQL * Loader
- Loading Data Warehouse with External Tables
- Loading a Data Warehouse with Export/Import

1.12.1. Loading a Data Warehouse with SQL*Loader

To load data from flat file to Oracle DWH, SQL*Loader is being used. A simple SQL*Loader is

Control File:



```
load data

infile'i:\Research_implementation\data for orcl1\employee.txt'

into table target_employee

fields terminated by "   :"

(emp_no,fname,mname,lname,address,ph,program,hobby,skills)
```

A control file which is load data into target_employee from external flat file employee.txt

1.12.2. Loading a Data Warehouse with External Tables

In this method for loading data an external table is being used and it works like virtual table. Data is being accessed with the help of SQL, PL/SQL and Java [40].

We perform DML operation and create indexes for a regular table but there is no concept of such activity on external table and this is the difference between such both of tables.

1.12.3. Loading a Data Warehouse with Export/Import

When data is being inserted into the target system then export and import is being used and no complex transformation is possible here.

1.13. Transformation Mechanisms

- Transforming Data Using SQL
- Transforming Data Using PL/SQL
- Transforming Data Using Table Function

1.13.1. Transforming Data Using SQL

- CREATE TABLE AS SELECT And INSERT /*+APPEND*/ AS SELECT
- Transforming Data Using UPDATE
- Transforming Data Using MERGE
- Transforming Data Using Multitable INSERT

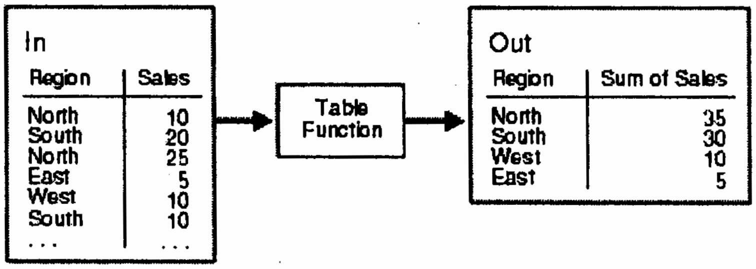
1.13.2. Transforming Data Using PL/SQL

PL/SQL is being used for the implementation of complex transformation, for example PL/SQL give the facility to open multiple cursors and obtain data from various foundation sources. After reading data, such data is being joint according to the industry policy [40].

1.13.3. Transforming Data Using Table Functions

It supports pipeline and parallel execution of transformation in PL/SQL. Table function generates set of rows as output. A table function is defined as a function that can produce a set of rows as output. Additionally, table functions can take a set of rows as input. Prior to Oracle9i, PL/SQL functions: [40].

```
INSERT INTO Out SELECT * FROM ("Table Function"(SELECT * FROM In));
```



In

Region	Sales
North	10
South	20
North	25
East	5
West	10
South	10
...	...

Out

Region	Sum of Sales
North	35
South	30
West	10
East	5

Figure 1.8 Table Function [40]

This figure illustrates the example of an In table containing: [40].

- North 10
- South 20
- North 25
- East 5

These are in columns of Region and Sales. They then go into a table function, which leads to output of: [40].

- North 35
- South 30
- West 10
- East 5

1.14. Maintaining the Data Warehouses

- Using Partitioning to Improve Data Warehouse Refresh
- Optimizing DML Operations During Refresh

- Refreshing Materialized Views
- Using Materialized Views with Partitioned Tables

1.14.1. Optimizing DML Operations during Refresh

➢ Efficient MERGE Operation implementation

1. MERGE Operation
2. Omitting the INSERT Clause
3. Omitting the UPDATE Clause
4. Skipping the UPDATE Clause
5. Conditional Inserts with MERGE Statements
6. Using the DELETE Clause with MERGE Statements
7. Unconditional Inserts with MERGE Statements

1.15. Referential Integrity maintenance

In a few DWH new data will have to be inserted into some tables to ensure referential integrity. For example you have two tables; sales and product. Sales table is refreshed every night but product table is being refreshed on weekly basis because the changes in product are relatively slow. If on Tuesday there is a new product has been introduced. So this new product_id must be presented in DWH for sales table because sales table is being refreshed every night and product table will be refreshed on coming Sunday and today is Tuesday, so in this way decision maker will have to wait for five days which is not suitable for taking good decision on right time [40].

1.16. Purging Data

Rarely, DWH must be free from those data which is now out of trade to give much space for upcoming fresh data. For example if an item ABC has been sold by a company and such item now has gone out of business and business owner has no interest in such item, so this item must be removed from DWH. A 3 to 7 year time horizon for maintaining data is normal for the information warehouse [7].

1.17. Change Data Capture

Incremental extraction which is also knows as Change Data Capture. Since DWH is being refreshed every night, so on coming night only change data will be moved to DWH because CDC gives the most recent changed or new data for further use. Near real-time or on-time DWH is being provided by CDC. With the help of CDC the extraction process becomes faster due to small amount of data. CDC is very tough issue because it disturbs and interfere foundation area when most recent data is being extracted from such area.

Quit a few self-developed CDC methods that have been implemented to capture the most recent changed or new data [40].

1.18. Timestamps

There is an extra column which is used for timestamp in a foundation area. With the help of this column the most recent changed or new data can be identified [40].

1.19. Partitioning

There is different range of partitioning in a number of foundation system which is very helpful to identify the most recent changed or new data [40].

1.20. Triggers

If there are separate triggers for each foundation side then most recent changed or new data can be identified although it affects the performance of foundation system [40].

1.21. Capturing Change Data without Change Data Capture

This method gives us changed data after transporting an entire table from foundation system to staging database when new table is minus from old table. We could also find out removed data by minus old table from new table. Although there some disadvantages but nevertheless, it gives most recent data [40].

1.22. Capturing Change Data with Change Data Capture

Change Data Capture can capture and publish committed change data in either of the following two modes (Synchronous & Asynchronous) [40].

1.22.1. Synchronous & Asynchronous CDC

In synchronous form, changes moves to snapshot tables by identifying directly with the help of triggers on the foundation system during the same transaction. It means that changes are being identified on time with no latency [40].

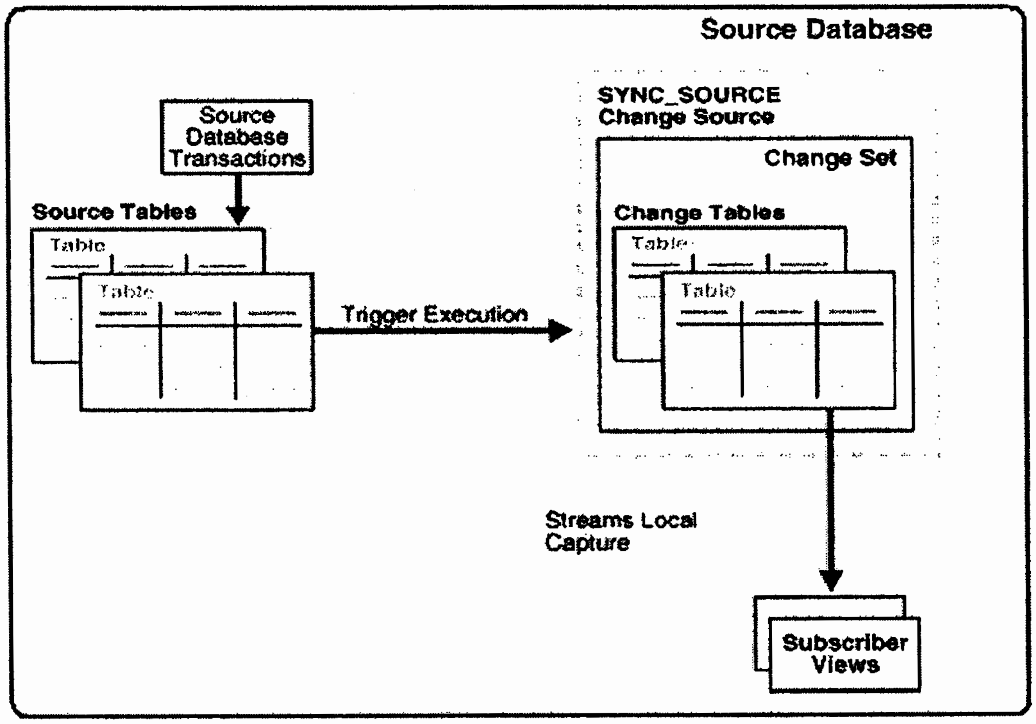


Figure 1.9 Synchronous Change Data Capture Configuration [40]

In the subsequent form changes is not directly identifying during the same transaction but when transaction is committed and at a particular position of user defined time. For example after every 12 hours or on Sunday at 8:00 pm.

For the Asynchronous, there three modes of capturing change data are described in the following sections:

- Asynchronous HotLog Mode
- Asynchronous Distributed HotLog Mode
- Asynchronous AutoLog Mode

1.22.1.1. Asynchronous HotLog Mode

Online redo log file is being used to identify change data in asynchronous HotLog Mode on foundation database with short latency.

A single HotLog change source is predefined at foundation database which holds the foundation database's current online redo log files [40].

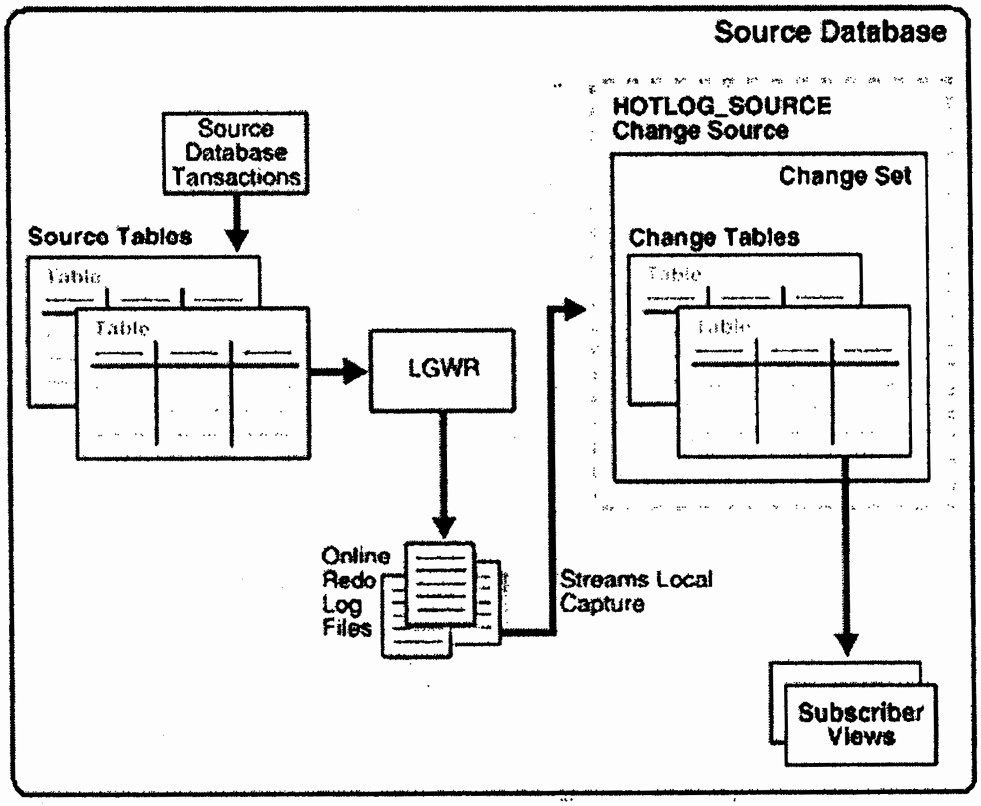


Figure 1.10 Asynchronous HotLog Configuration [40]

1.22.1.2. Asynchronous Distributed HotLog Mode

Online redo log file is being used to identify change data in asynchronous Distribute HotLog Mode on foundation database [40].

There is no such predefined Distributed HotLog change source. A Distributed HotLog change source at foundation database which represent the foundation database's current online redo log files while many Distributed HotLog change sources can be defined by the staging database publishers [40].

Distributed HotLog identify the change data using online redo log files and sent it to the staging database through using stream. There is a link between source database to staging database and staging database to source database which is being used by their individual publisher [40].

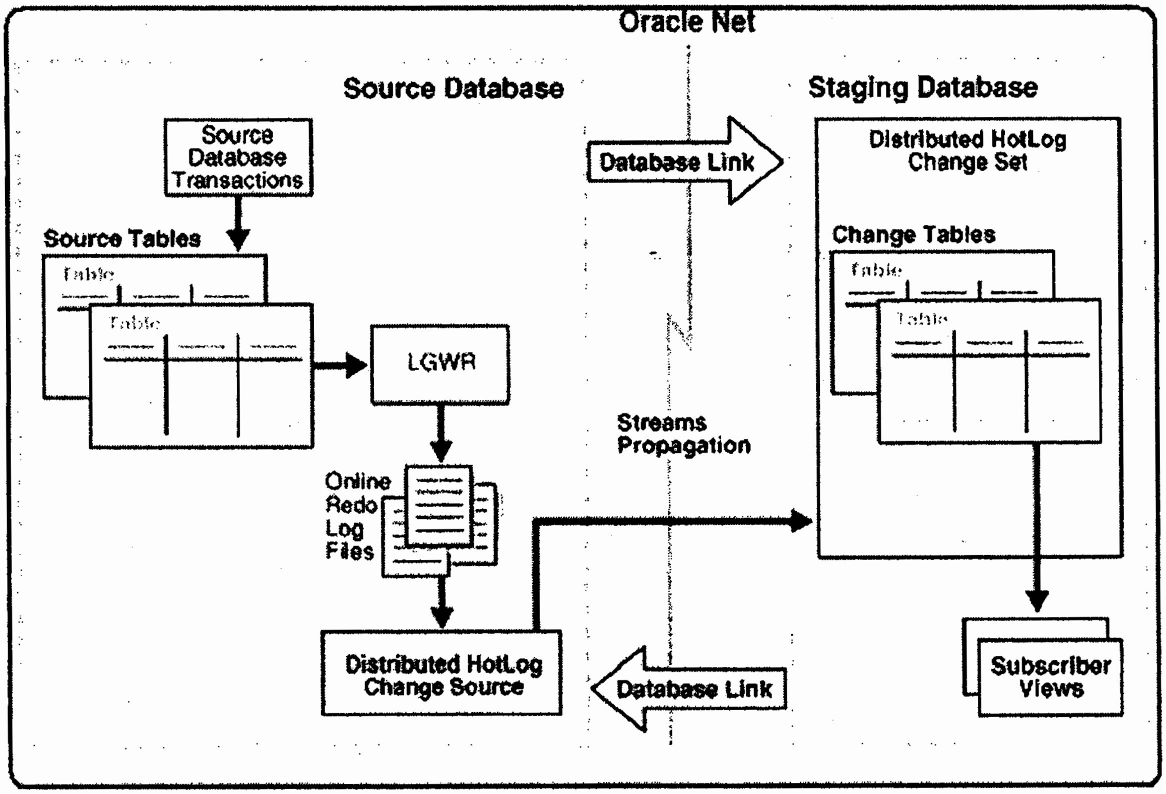


Figure 1.11 Asynchronous Distribute HotLog Configuration [40]

Consequently in enterprise data warehouses jobs are run via batch runs because thousands of jobs run daily. Data warehouses usually have a special area called a staging area where major ETL steps are performed [11]. The purpose of the supplementary structure is to act as a buffer and holding area between the genuine data warehouse and the data translator [21].

The basic working of gadgets (tools) can be summarized in the following prominent tasks or the duty of ETL softwares are [32]:

(a) The identification of relevant information at the starting place (Source);

(b) The mining of this information;

(c) The customization and amalgamation of the information coming from compound foundations into a universal format;

(d) The cleaning of the consequential data set, on the foundation of database and business rules, and

(e) The broadcast of the data to an auxiliary arrangement of the data warehouse.

As soon as the data is staged in the auxiliary structure of the data warehouse, the diagnostic subject areas are refreshed; extracting data from the staging focus areas using conversion logic in stored procedures.

J.H. Hanson, M.J. Willshire, Rifaleh Rami, and Benharkat Aicha Nabila [21, 29] also propose to allow DBMS (database management systems) to play an extended function as a data transformation engine as well as a data stock up.

During last five years the rate of refresh for DWH has increased. The following figure shows the percentage of different kind of DWH refresh which different organizations do according to their rules and feasibilities.

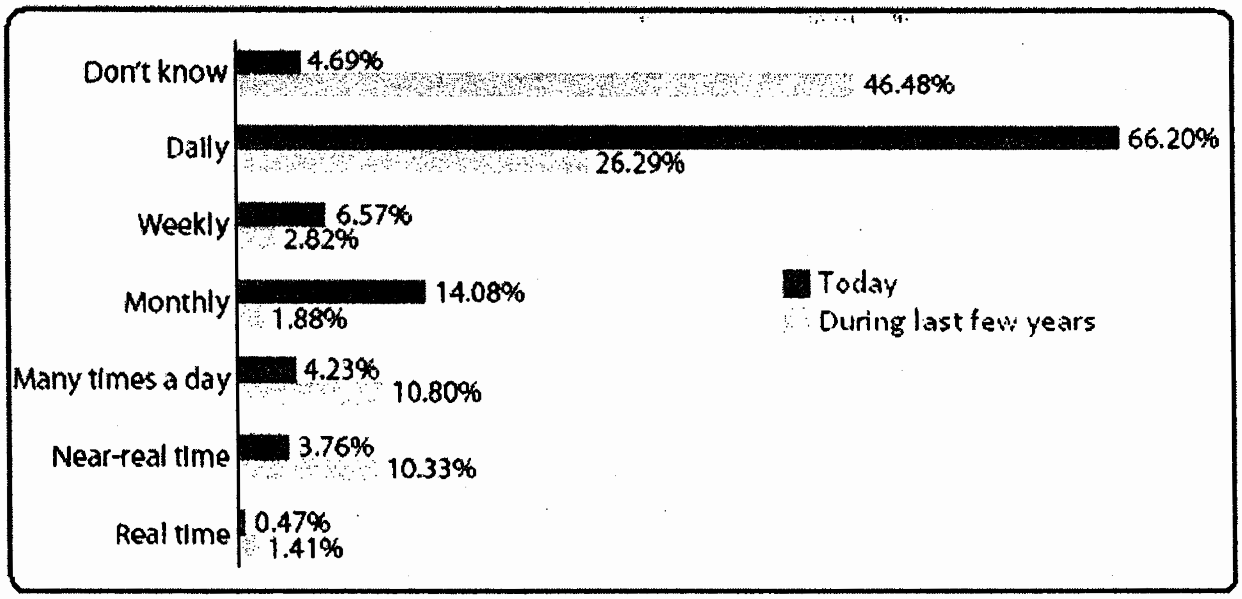
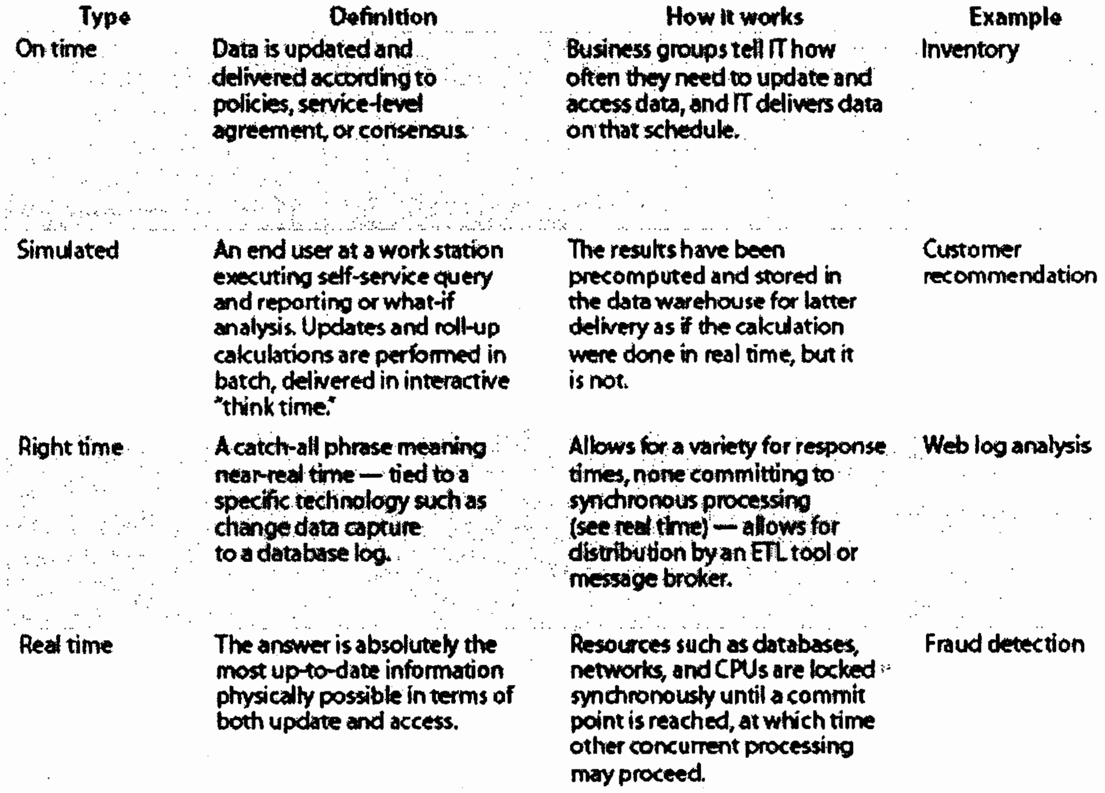


Figure 1.12 DWH Updates Acceleration [17]



Type	Definition	How it works	Example
On time	Data is updated and delivered according to policies, service-level agreement, or consensus.	Business groups tell IT how often they need to update and access data, and IT delivers data on that schedule.	Inventory
Simulated	An end user at a work station executing self-service query and reporting or what-if analysis. Updates and roll-up calculations are performed in batch, delivered in interactive "think time."	The results have been precomputed and stored in the data warehouse for latter delivery as if the calculation were done in real time, but it is not.	Customer recommendation
Right time	A catch-all phrase meaning near-real time — tied to a specific technology such as change data capture to a database log.	Allows for a variety for response times, none committing to synchronous processing (see real time) — allows for distribution by an ETL tool or message broker.	Web log analysis
Real time	The answer is absolutely the most up-to-date information physically possible in terms of both update and access.	Resources such as databases, networks, and CPUs are locked synchronously until a commit point is reached, at which time other concurrent processing may proceed.	Fraud detection

Figure 1.13 Many Meanings of Real Time [17]

Chapter #2
Literature Review

2.1. Literature Review

Literature review has a great role and value for identifying and understanding a problem. Without literature review it is impossible to identify a correct and valuable problem to give solution which advances the knowledge. A minimum work which adds something to the knowledge is called research.

Every researcher needs to address "What", "Why" and "How" for every research problem.

"A review of prior, relevant literature is an essential feature of any academic project. An effective review creates a firm foundation for advancing knowledge. It facilitates theory development, closes areas where a plethora of research exists, and uncovers area where research is needed" [43].

2.1.1. Literature Review Process

The aim of this research is an essential feature of the academic project and pay gratitude to the researchers that have contributed much in the field of Data Warehouse. This research provides the chance of working in Real Time Data Warehouse (RTDWH). Main emphasis of this research is to discuss Near Real Time Data Refresh for Active DWH. During the research different aspects of the Real Time Refresh methods are discussed and analyzed the positive and negative aspects of the Near Real time Refresh for Active DWH.

2.1.2. Objectives of literature review are as follows

- Formulation of search string to find relevant literature
- Formulation of the research questions
- Evidence gathering for existing research
- Identification of research gaps
- Identification of future directions

2.1.3. The Process

To perform literature review following set of activities is performed:

- **Search Criterion Formulation:**

  In this step the search criterion is identified to find the relevant literature.

- **Formulation of Research Questions:**

  Important research questions are identified.

- **Defining Selection Criteria:**

  On the basis of defined terms and their priority the selection criterion is defined.

- **Selection of Relevant Literature:**

  On the basis of search criterion, found literature is investigated and characterize on the basis of selection criterion.

- **Data Summarization:**

  Selected data is summarized to analyze.

- **Research Gaps Identification:**

  On the basis of analysis results the research gaps are identified.

- **Formulation of Future Directions:**

  On the basis of research gaps the future directions are formulated.

- **Report:**

  All the steps discussed above are written in well formatted way.

The flowchart in Fig. 2.1 elaborates all set of activities in detail.

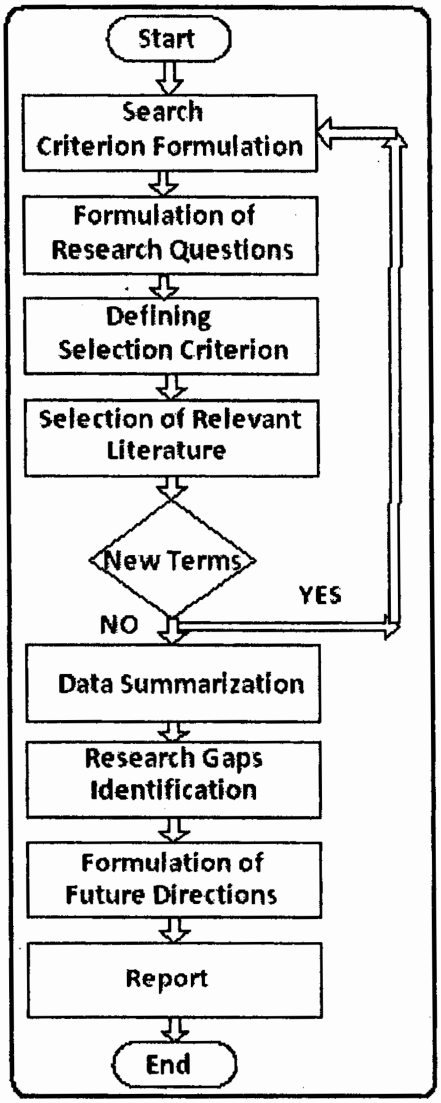

Start
Search Criterion Formulation
Formulation of Research Questions
Defining Selection Criterion
Selection of Relevant Literature
New Terms
YES
NO
Data Summarization
Research Gaps Identification
Formulation of Future Directions
Report
End

Figure 2.1 Literature review Process Flow Chart

2.1.4. Formulation of Research Questions

Identifying the valid research questions is an important component of any literature review [42]. For the formulation of the research questions in this literature review I have used the following terms.

Table 2.1: Terms

Area of interest	Updating DWH using near real-time refresh
Technique	Near real-time
Goal	Updating DWH, minimize the extraction's time during peak hours
Environment	Data Warehouse

2.1.5. Selection of Relevant Literature

- ### Search Process

The important question is "what" to find before going to investigating any thing. To find its answer the following tasks are performed:

1. Terms of the research are defined.
2. Synonyms of key terms are defined.
3. Keywords, alternate terms are defined.
4. terms are combined using OR and AND operator

Synonyms, OR and AND operator use is shown in tables 2, 3 and 4

Table 2.2: Synonyms Derived from Terms

Basic Term	**Synonyms**
Active data warehousing	Real-time, Semi real-time, near real-time, in-time data, Right Time warehousing
Efficient ETL Process	Distribute the ETL Load
Event-Based Near Real-Time DWH	Trigger-Based Near Real-Time DWH, Log-Based Change Data Capture

Table 2.3: Combining synonyms using OR operator

(Active DWH OR Real-time DWH OR Semi real-time DWH OR Near real-time DWH)
(Efficient ETL Process OR Distribute the ETL Load)
(Event-Based Near Real-Time DWH OR Log-Based Change Data Capture OR Trigger-Based Near Real-Time DWH)

Search Sources

The terms defined above are search in different data sources. Data sources used for search are as follows.

- **Online Databases**

1. ACM Digital library
2. CIA - Computer Index Australasia
3. Compendex
4. Computer Database
5. Computing Reviews
6. Derwent Innovations Index
7. Energy Citations Database
8. ENGINE - Australian Engineering Database
9. Gartner Group Intraweb
10. IEEE Xplore
11. Information Technology Case Studies
12. INSPEC
13. ProQuest Computing
14. Scopus
15. SpringerLink
16. Telecommunications
17. Web of Science

Online Search Engines

1. Google scholar
2. CiteSeer
3. Agile alliance

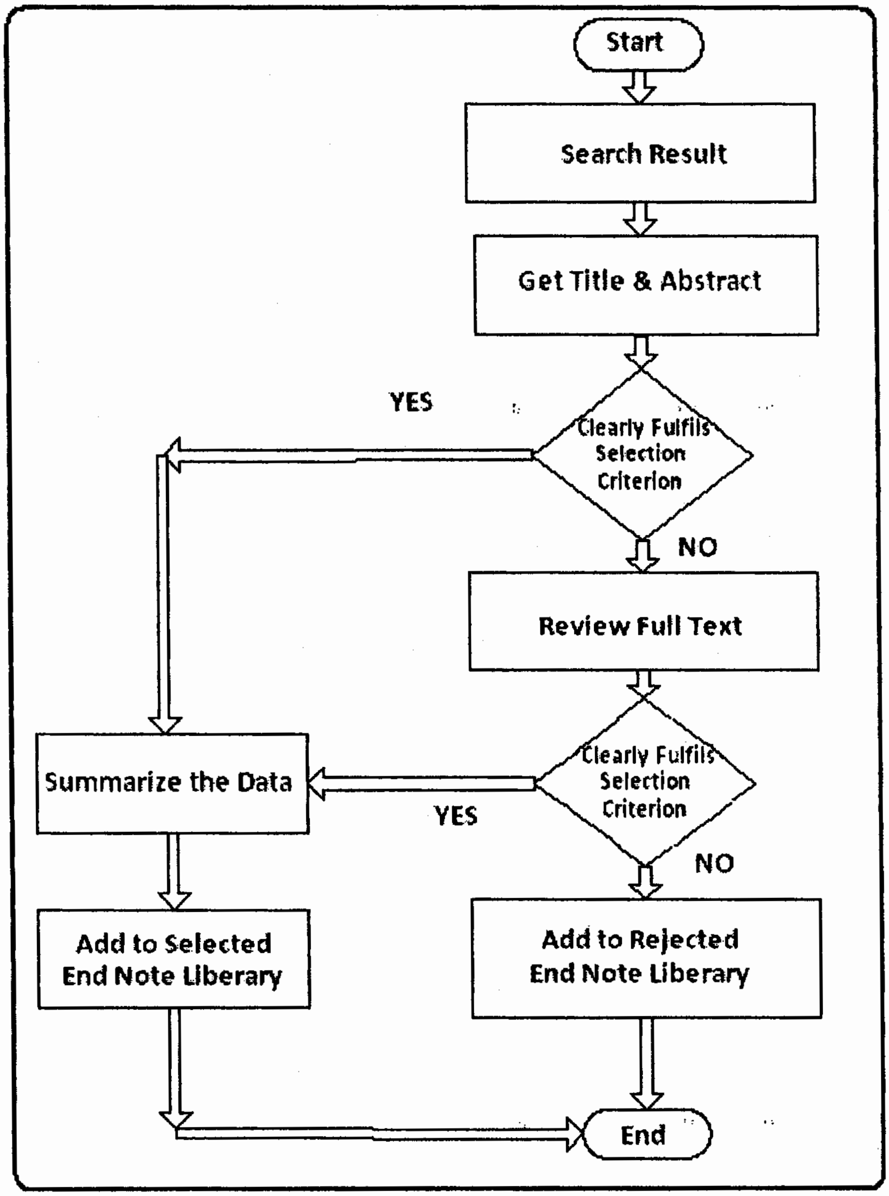

Start
Search Result
Get Title & Abstract
Clearly Fulfils Selection Criterion
YES
NO
Review Full Text
Clearly Fulfils Selection Criterion
YES
NO
Summarize the Data
Add to Selected End Note Liberary
Add to Rejected End Note Liberary
End

Figure 2.2: Identifying relevant literature Flow Chart

2.1.6. Search Results

Table 2-4: Summary of Search Results

Serial No.	Database Name	No. of Papers Found	Selected	Not Selected	Duplicated in other sources.
1	ACM Digital library	12	1/12	11/12	6(IEEE),
2	CIA - Computer Index Australasia	0	-	-	-
3	Compendex, Geobase, Georef	13	-	-	10(IEEE), 3(ACM)
4	Computer Database	0	-	-	-
5	Computing Reviews	0	-	-	-
6	Derwent Innovations Index	5	0/5	5/5	-
7	ENGINE - Australian Engineering	0	-	-	-
8	Gartner Group Intraweb	0	-	-	-
9	IEEE Xplore	25	5/25	20/25	-
10	IT Case Studies	0	-	-	-
11	INSPEC	20	-	-	15 (IEEE), 5(ACM)
12	ProQuest Computing	3	-	-	2(IEEE)
13	Scopus	50	0	0	40(IEEE), 14(ACM)
14	SpringerLink	50	0/8	8/8	2(ACM), 43 Book Ch.
15	Telecommunications	0	-	-	-
16	Web of Science	12	-	-	5(IEEE), 6(ACM)
17	Author Personal web Page	13	-	-	20(IEEE), 5(ACM)

2.1.7. Paper Selected

Following are the papers selected for literature review

P1 → Youchan Zhu, Lei An, Shuangxi Liu, "Data Updating and Query in Real-time Data Warehouse System", IEEE International Conference on Computer Science and Software Engineering, pp. 1295-1297, 2008.

P2 → Alexandros Karakasidis, Panos Vassiliadis, Evaggelia Pitoura "ETL Queues for Active Data Warehousing", ACM, Baltimore, MD, USA, pp. 28-39. June 2005.

P3 → M. Asif Naeem, Gillian Dobbie, Gerald Weber "An Event-Based Near Real-Time Data Integration Architecture" IEEE, 2008.

P4 → Li Chen, Wenny Rahayu, David Taniar "Towards Near Real-Time Data Warehousing" in 24th IEEE International Conference on Advanced Information Networking and Applications 2010.

P5 → JinGang Shi, YuBin Bao, FangLing Leng, Ge Yu "Study on Log-Based Change Data Capture and Handling Mechanism in Real-Time Data Warehouse" IEEE International Conference on Computer Science and Software Engineering, pp. 478-481, 2008.

P6 → Dr.Muhammad Younus Javed, Asim Nawaz "Data Load Distribution by Semi Real Time Data Warehouse", I EEE, 2nd International Conference on Computer and Network Technology, pp. 556-560, 2010.

2.1.8. Salient Features of Selected Studies

Table 2-5: Salient Features of Selected Studies

Sr. No.	Salient Features
P1	A technique is being proposed for real-time DWH based on SOA which captures the changed data with the help of web services. Multiple Caches are being used for storing the real-time data and XML, web service for warehouse updating.
P2	A framework has been introduced in which all activities of ETL have been performed on networks queue for ADWH without maximum changes in the software configuration & less load on source. Queue theory has been suggested for the performance prediction and operation alteration.
P3	Architecture for near real-time ETL is proposed which is event-based to minimize the data loading latency with the help of master data supervision in which only changed data will be picked by Message Driven Bean (MDB) and for this purpose a message oriented Database Queue (DBQ) is being used.
P4	When the DWH desire to be rigorously near real-time then three combinations of actions must be judged, frequency of request, impact of record for DWH refresh and number of affected record. Considerable assistance is offered by the proposed approach.
P5	Log-base change data capture & data extraction framework is being suggested which processes such data on the basis of log analysis. Pushed data in a queue is loaded to the DWH with a little impact on source after processing using priority scheduling algorithm by system to improve data quality & freshness.
P6	A technique is being proposed by using usual and real-time techniques to distribute the job of extraction for the data volume. Finish the ETL process within its time window by utilizing ETL idle time and performance is being achieved by distribution of extraction job.

2.1.9. Detailed Summary of Selected Studies

The detailed study of selected papers is give below.

P1 → Youchan Zhu, Lei An, Shuangxi Liu, "Data Updating and Query in Real-time Data Warehouse System", International Conference on Computer Science and Software Engineering, pp. 1295-1297, 2008.

It is the most significant subject and topic for most of organization that they are facing means real-time DWH updating and it is the key issue. Technology for real-time DWH examined in which service of data captures is being used to capture the changed data by using web service, XML message is being used to send the change data to the DWH and the main intention is to shrink the query load. The cost of implementation has been reduced by using SOA technology. In different parts; the architecture of real-time DWH, Updating data, Capturing data and query are explored.

Goals of this study are as follows:

1. A new method/process for real-time DWH.
2. Real-time changed data capture.
3. An architecture based on SOA for real-time DWH
4. Reduce query load in real-time DWH.

Assumption considered in this study is as follows:

1. The data freshness degree request Q needs is n-minute, then it can be satisfied on n-cache.

The results of this study are shown below

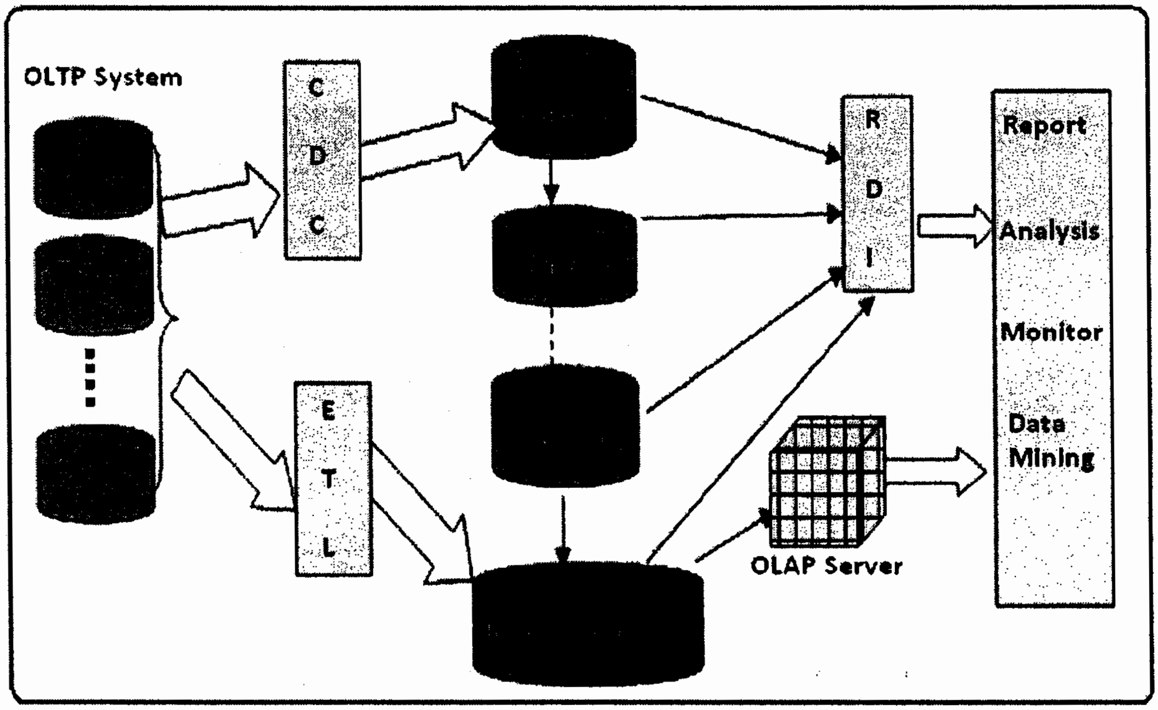


Figure 2.3 Real-time data warehouse architecture [1]

The architecture which is mode up of OLTP, CDC, ETL Multi-cache, OLAP, RDI & the application, shows how changed data has been stored immediately identified by web service and then DWH is being updated with the change data by using XML messages.

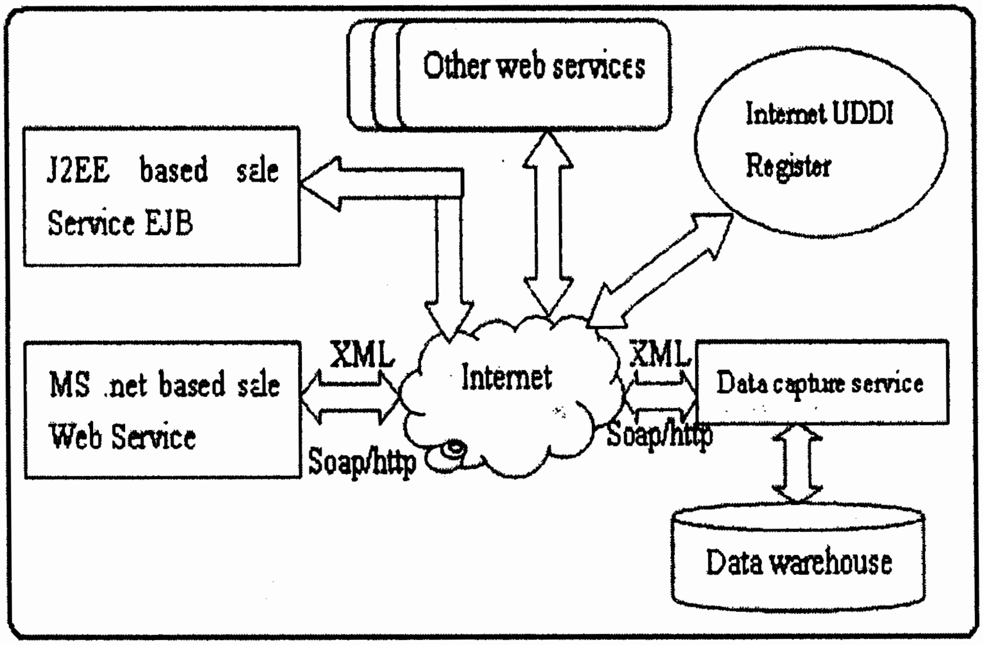


Figure 2.4 Data capture based on web service [1]

There is a message queue in all cache which is managed by MQM (message queue manager) and the main job is to make light of the query load.

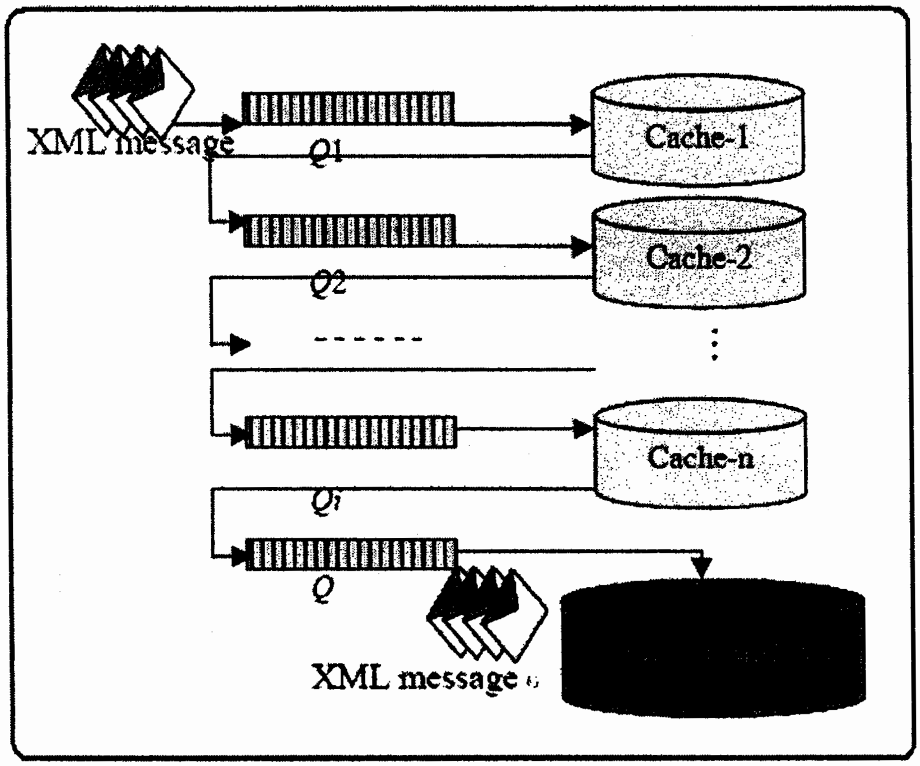


Figure 2.5 Updating Process [1]

Before loading data into DWH, The format of transmitted data in the queue is XML, since there are multistage data caches for storing fresh data in different time and with the time increased data from high level queue is sent to the lower level queue with the help of messages. The refresh cycle of each cache is; Cache-0 is 5 minute, Cache-1 is 10 minute, Cache-2 is 30 minute, Cache-3 is 60 minute separately.

Table 2.6: Refresh Cycle of each cache [1]

Cache-0	5 minute
Cache-1	10 minute
Cache-2	30 minute
Cache-3	60 minute

Claims of P1

1. Web service captures the changed data.
2. Reduced the implementation cost.
3. Stability of system is increased.

Future Work mentioned in study P1

1. No future work is being mentioned.

P2 → Alexandros Karakasidis, Panos Vassiliadis, Evaggelia Pitoura "ETL Queues for Active Data Warehousing" June 2005

Many DWH are being refreshed every night means when all source system are off. But ADWH is an idea to update the DWH as faster as possible when fresh data arrived to fulfill the high demands of organization. In this paper it has been presented that whenever fresh data is arrived, it immediately moved to ADSA for the sake of transformation and cleaning such data. More and more fresh data is being extracted with no more change in the source side software with fewer burdens.

In ADSA there is no. of queues which gets fresh data from source and then after processing (filtering and transformations) it passes to the next queue. The data delay and system overhead is predicted during the middle layer (ADSA) by using queue theory.
There are some problems which influence the performance of system. In two tier architecture both foundation system and DWH reside on separate device, so where to place DWH Staging Area either with source side or with DWH side? If staging area is place on same machine where source resides then burden will increase but if staging area is moved to DWH machine then it may be good but again if DWH server is too load or its configuration is too complex then it will create problem, so the best option is to put the staging area on totally separate machine.

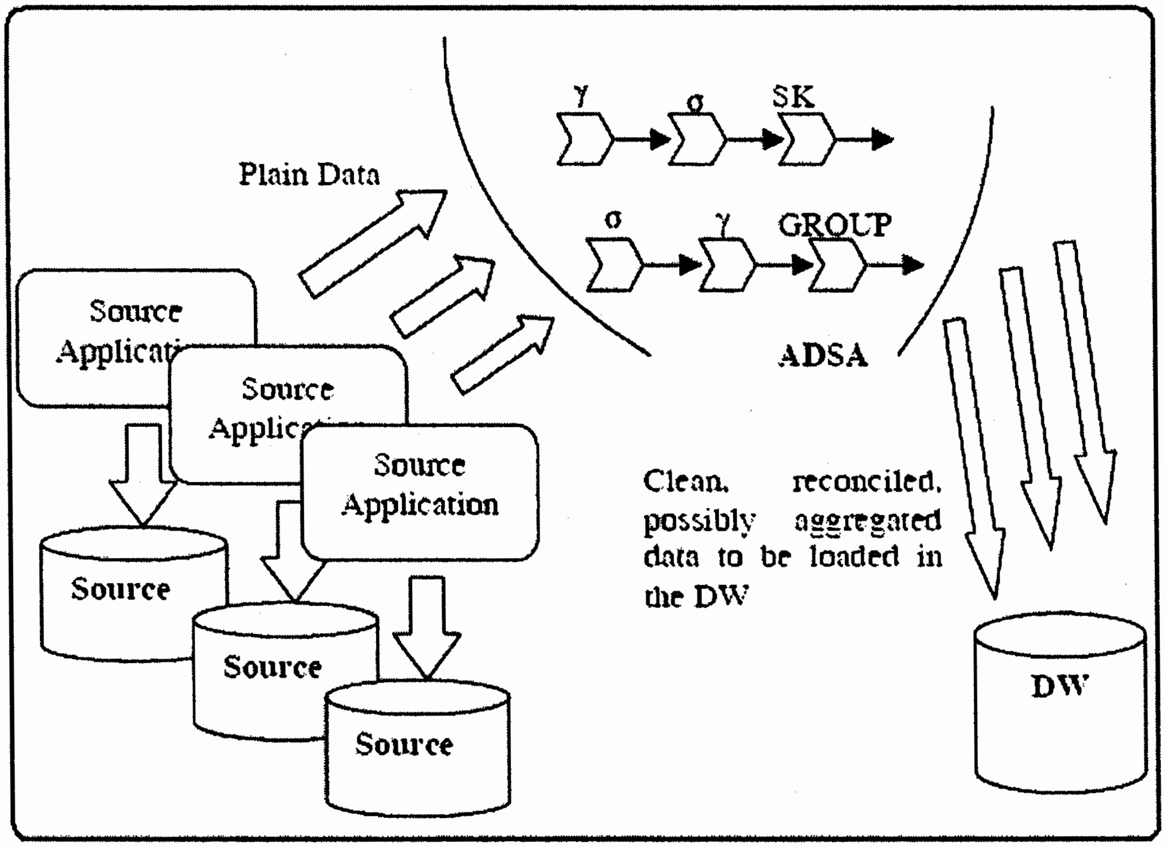


Figure 2.6 Architecture Overview [2]

Goals of this study are as follows:

1. Minimal changes in the software configuration of the primary source.
2. Impose minimum additional workload to the source.
3. Stable interface at the warehouse side.
4. Ensure maximum freshness of data

Assumption considered in this study is as follows:

1. Each node of the network consists of a single server with exponential arrival and exponential service times.
2. Warehouse only store the data and does not perform other task.
3. The legacy application sends 100,000 records to the staging area in block of 100 records over TCP.

4. Some of the tuples, after being transformed, continue through the system as accepted.
5. The number of those tuples equals to the number of tuples produced as a result of the transformation, so the rest of the tuple will be rejected by the system after their service and exit the system.

   | tuples rejected | = | tuples entering service | - | tuples accepted |

6. Some of the incoming customers continue and some exit the system depending on the merging of factors.
7. The static snapshot of the regular flow is being assumed.

The results of this study are shown below

Figure 2.7 shows the impact at the source using packets at the SflowR of various sizes.

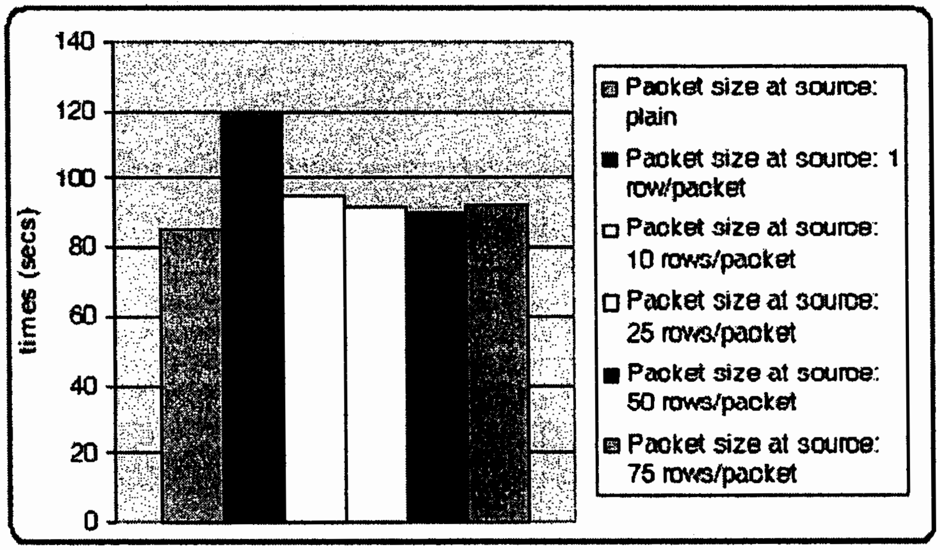


Figure 2.7 Packet size of the SFlowR and impact at source [2]

Figure 2.8 shows the Data Freshness of Online ETL for different scenario

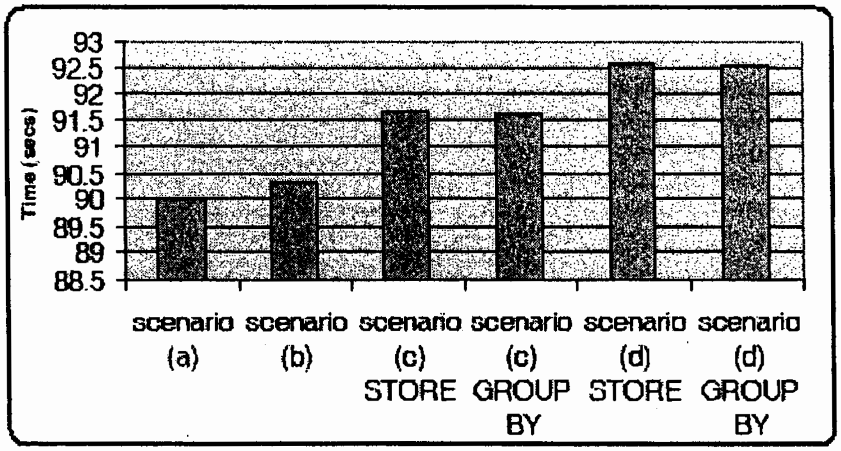


Figure 2.8 Data freshness for each scenario [2]

The x-axis for Figures 2.9 shows the number of rows in a packet. The y-axis of the diagrams measures the throughput of inserting the records.

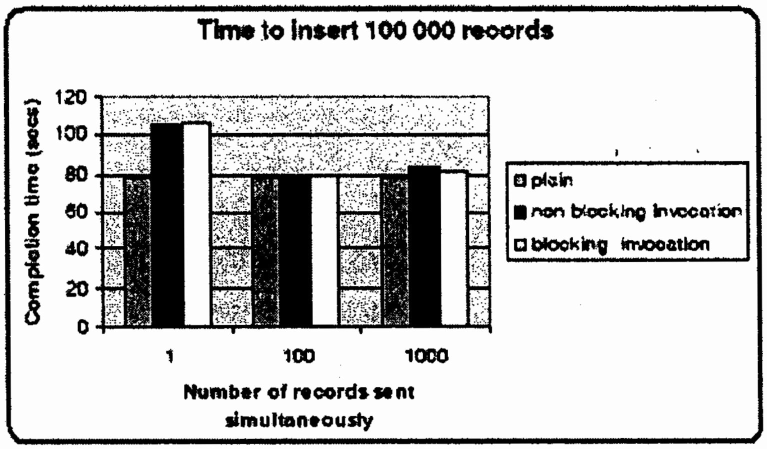


Figure 2.9 Time to insert 100 000 records using two-tier topology [2]

Claims of P2

1. Overall overhead at the source side is around 1.7%
2. The amount of code modification is around 100 lines without affecting application
3. Impact of software configuration & data freshness is quite satisfactory.

Future Work mentioned in study P2

1. Work with the failure management of the components of the environment, to determine safeguarding technique and fast resumption algorithms for the even of a failure.
2. Tuning can be made, by testing multiple concurrent loading sources for the DWH.

P3 → M. Asif Naeem, Gillian Dobbie, Gerald Weber "An Event-Based Near Real-Time Data Integration Architecture" 2008.

ETL tools work according to push technology which is being shown in figure 2.10, in most of DWH data is being loaded from foundation system on nightly basis or weekly basis or even in some cases on daily basis and during loading process all source system must be gone off-line. Traditional DWH has not most current data which is unacceptable to the customer or DWH users. Most fresh data on right time must be available to take good decision.

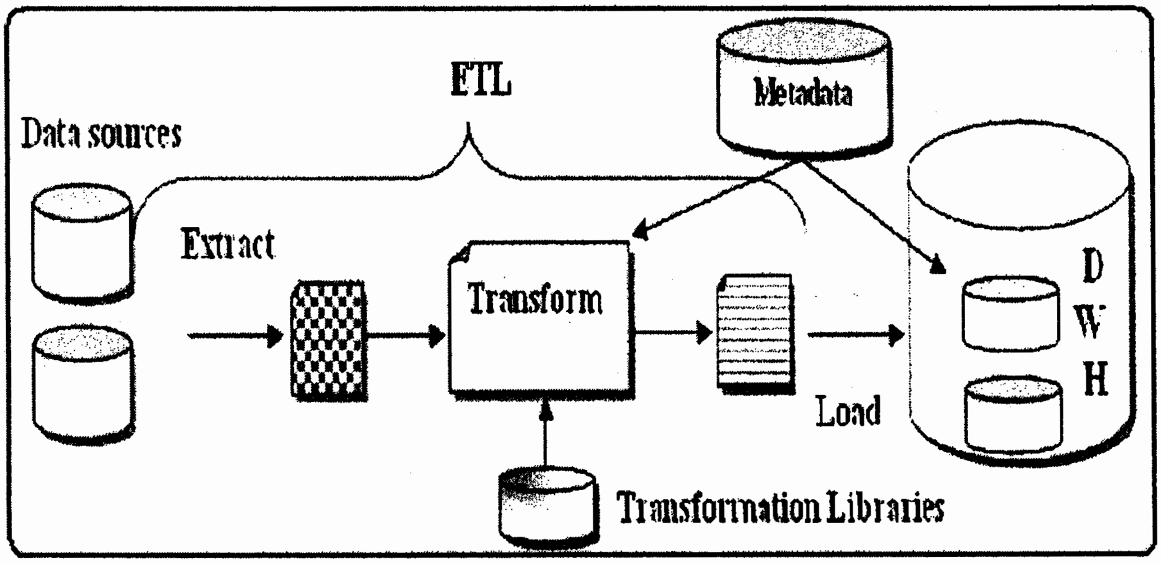


Figure 2.10 Traditional DWH Architecture [3]

Goals of this study are as follows:

1. Minimize the data loading latency.
2. Enhancement of the content in loosely coupled.

3. Examine Trigger-based near real-time ETL layer for transferring and transforming.
4. Master data management in the ETL layer.

Assumptions considered in this study are as follows:

1. In every loading window the product table is small compared to the order table.

The results of this study are shown below

Data integration is being shown using proprietary ETL by Figure 2.11, in which comparison of product table with order data is done in every loading window. So product table will be extracted, if there is either no change or new data in each loading window.

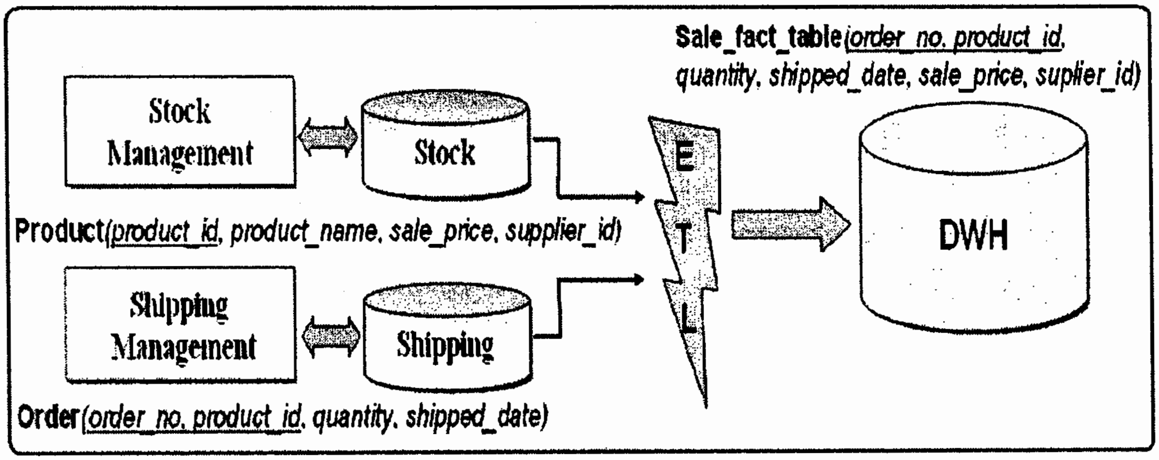

Sale_fact_table(order_no, product_id, quantity, shipped_date, sale_price, suplier_id)
Stock Management
Stock
Product(product_id, product_name, sale_price, supplier_id)
Shipping Management
Shipping
Order(order_no, product_id, quantity, shipped_date)
E
T
L
DWH

Figure 2.11 Data Integration using proprietary ETL [3]

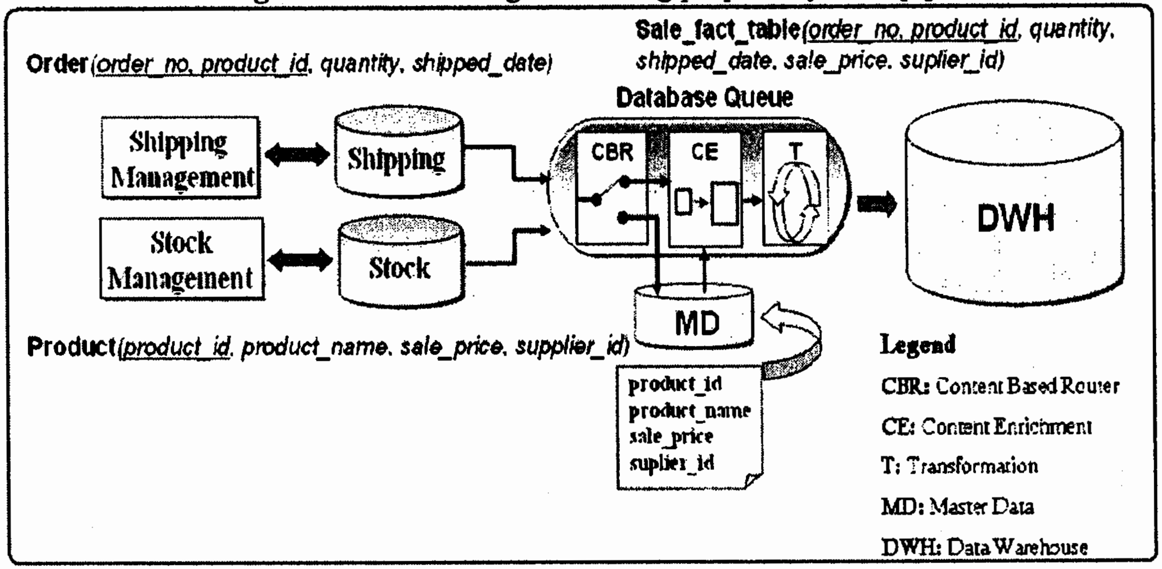

Sale_fact_table(order_no, product_id, quantity, shipped_date, sale_price, suplier_id)
Order(order_no, product_id, quantity, shipped_date)
Database Queue
Shipping Management
Shipping
Stock Management
Stock
CBR
CE
T
DWH
MD
Product(product_id, product_name, sale_price, supplier_id)
product_id
product_name
sale_price
suplier_id
Legend
CBR: Content Based Router
CE: Content Enrichment
T: Transformation
MD: Master Data
DWH: Data Warehouse

Figure 2.12 Data Integration using Traditional & proposed approach [3]

Figure 2.12 shows the proposed architecture which works on the event-based. The data is being divided into two types; master (product table) and transaction (order table). The master data is not changed frequently while transactional data is changed frequently. All master data that is being mentioned by transaction data must be available for the current loading window. The middleware is a software which captures changes without disturbing the source system. The DBQ is used to store messages. The CBR is being used to differentiate master and transactional data. Master data is moved to master table and transaction data is propagated to the enrichment process. MDB continuously monitors the DBQ whenever update data transformed, it is being extracted and moved to the DWH.

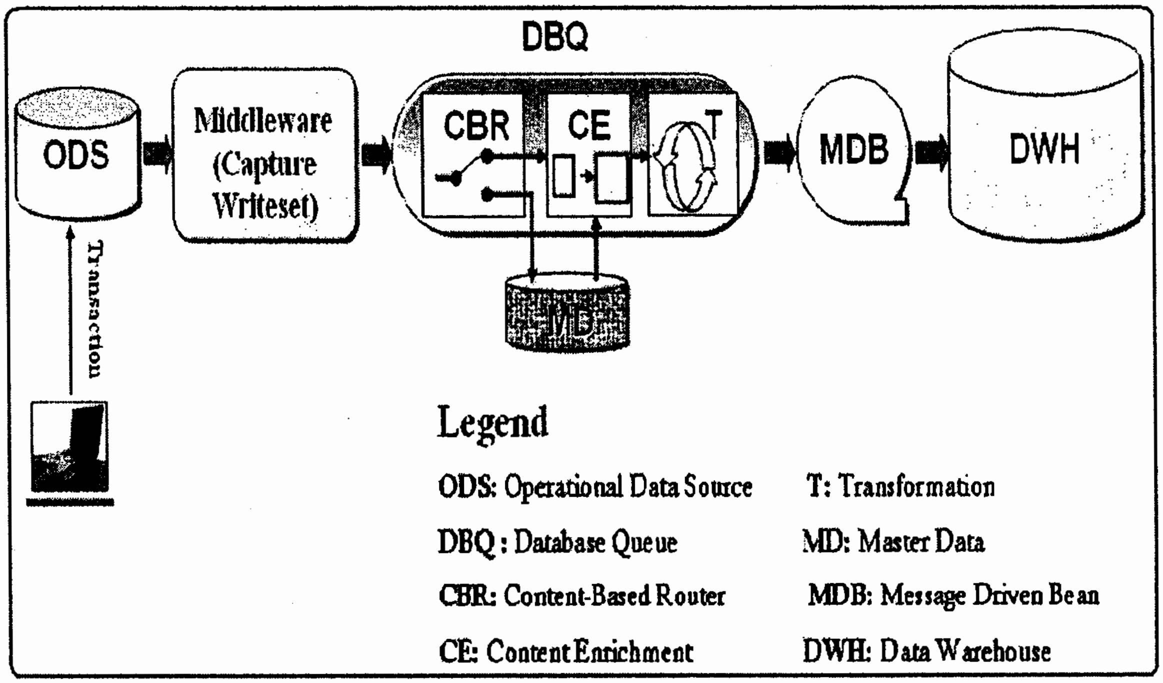


Figure 2.13 Data Integration using Traditional & proposed approach [3]

Claims of S3

1. Cooperative and effective architecture is obtained which is consistent with the information architecture of OLTP
2. Content enrichment which is supported directly by data integration.
3. Writeset extraction using a trigger-based approach.

Future Work mentioned in study S3

1. Propagation of the writeset to the database
2. Implementation of enrichment and transformation processes.
3. performance evaluation of the proposed architecture

P4 → Li Chen, Wenny Rahayu, David Taniar "Towards Near Real-Time Data Warehousing" in 24th IEEE International Conference on Advanced Information Networking and Applications 2010

DWH store huge amount of data extracted from totally autonomous foundation system and DWH users request not only for most fresh data but also want short time response, so the DWH is required to be refreshed more frequently and loading window must be smaller and smaller for data gaining. For such purpose only changed data to be moved to the DWH which is called incremental refresh.

Goals of this study are as follows:

1. Mechanism for near real-time by making the interval of the timestamp.
2. Calculation of the update impact
3. Number of records affected
4. Frequency Request Measure.

Assumptions considered in this study are as follows:

1. DWH has already been built.
2. There are already some data in table-I & table-II.
3. Already know the Materialized View (MV) schema because Auxiliary Table (AT) schema will be defined if MV is known.
4. On the basis of certain timestamp, the data will be updated for traditional DWH.

The results of this study are shown below

In this paper it is being proposed that update data will be stored to Queue and when a valuable update come then all updates will be moved to DWH after impact calculation. In

Table 2.7 shows a queue in which all updates are stored. All updates are stored in such queue before sum(Quantity) 4500 and this is a valuable update. Now it will be compared with average data which is 3780 items per season and max(Price) 600 which is double of the average price, so this update has impact on DWH.

Table 2.7: SALESFACT

Sex	Season	max(Price)	sum(Quantity)
F	Spring	200	2000
F	Summer	250	4500
F	Autumn	70	3900
F	Winter	90	5000
M	Spring	60	2800
M	Summer	80	4580
M	Autumn	90	2760
M	Winter	300	4700

Table 2-8: Summary from SALESFACT

Max(maxPrice)	300
Avg(totalQuantity)	3780
rowCount	8
totalQuantity	30240

Calculation of Update Impact

$$I = \frac{\mid \frac{600}{300} \mid + \mid \frac{4500}{3780} \mid}{2} \approx 1.60$$

Figure 2.14 shows the cost comparison Traditional, Real-time and near real-time DWHs. Traditional DWH is being refreshed on daily, weekly or monthly basis while for real-time data will be inserted directly to the DWH.

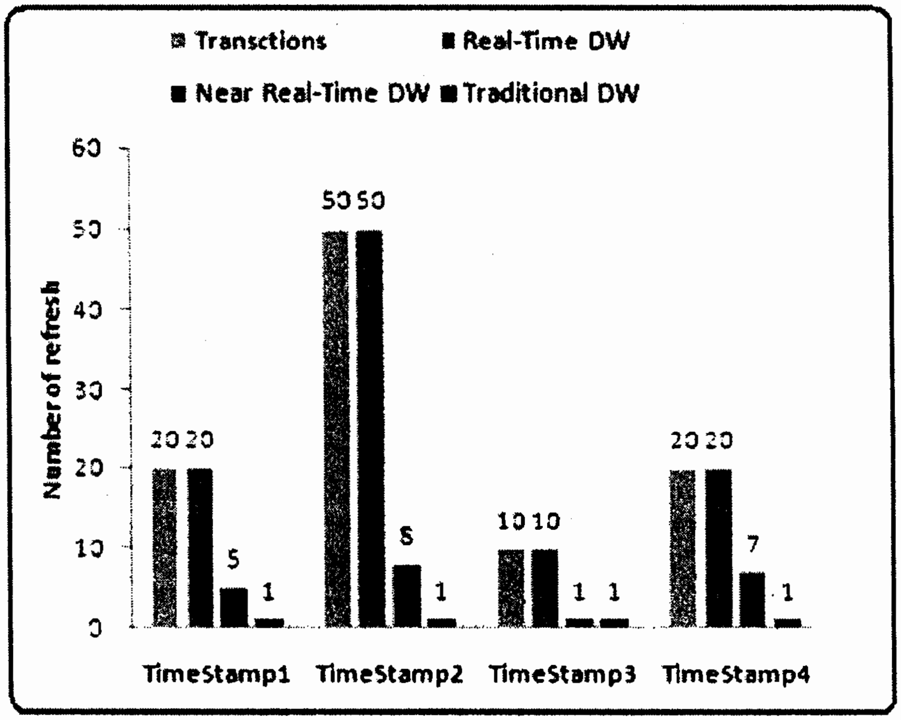


Figure 2.14 Comparison among 1GB DWHs [4]

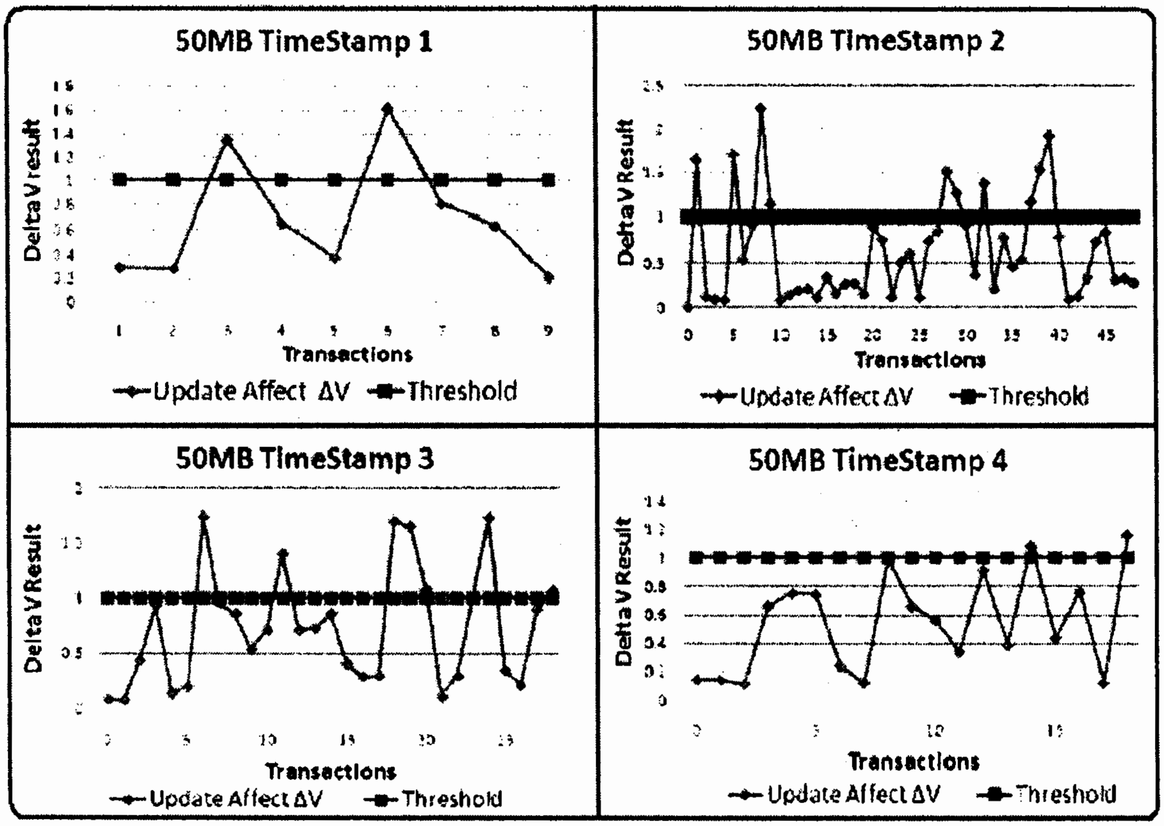
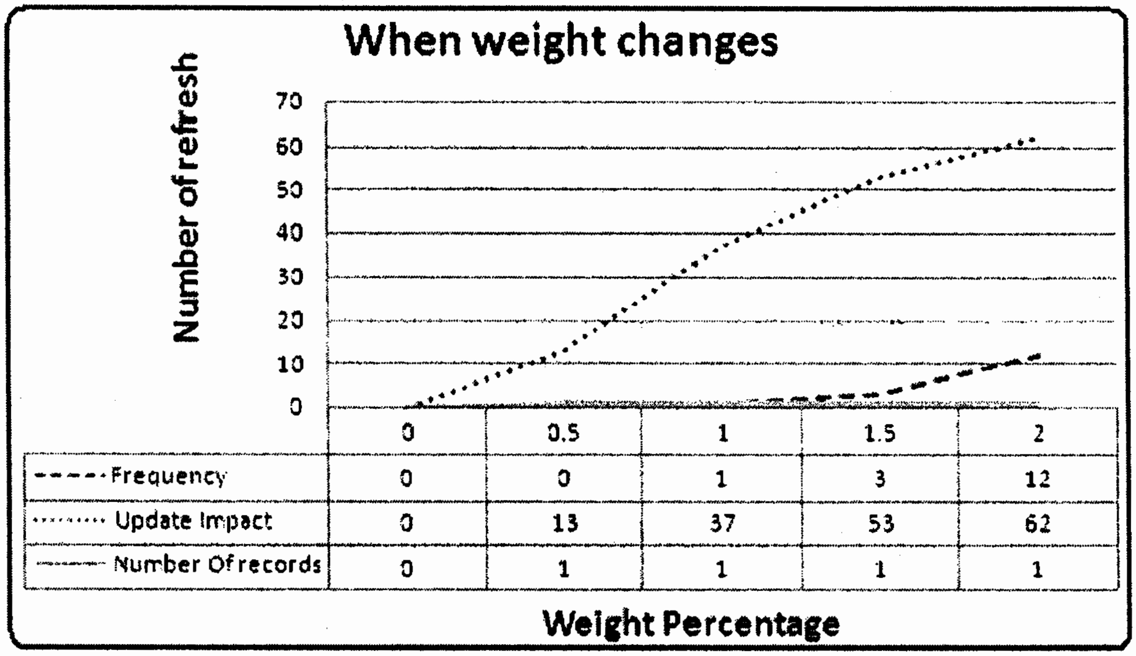


Figure 2.15 Comparison among 1GB DWHs [4]

There are 3 impact updates in time stampl, 10 impact updates in timestamp2, 8 updates in timestamp 3 and 2 updates in timestamp 4, all of these updates will have to wait until the time interval finished in order refreshing DWH, so there will be delay. We need to send only updates 3 out of 10 in timestamp 1 and 2 updates out of 20 in timestamp 4. The other updates are not required to be in real-time.

Figure 2.16 shows that three measure affect the frequency of refreshes of DWH. Due to the different value of measure the frequency of refresh will be different



	0	0.5	1	1.5	2
----Frequency	0	0	1	3	12
········ Update Impact	0	13	37	53	62
——— Number Of records	0	1	1	1	1

Figure 2.16 Comparison among 1GB DWHs [4]

Claims of S4

1. Minimize delay by only sending the important updates to the DWH.
2. Accumulate significant operational costs
3. Frequency of refresh depends on weights (FRM, RAM and Impact of Update)

Future Work mentioned in study S4

1. Extend the current work for covering of XML DWH whereby near real-time updates of XML warehousing can be determined.

P5 → JinGang Shi, YuBin Bao, FangLing Leng, Ge Yu "Study on Log-Based Change Data Capture and Handling Mechanism in Real-Time Data Warehouse" IEEE International Conference on Computer Science and Software Engineering, 2008.

Real-time DWH is a key technology and required due to not have of real-time updates. The most changed data is being captured by using log-based analysis and data quality is being improved by using scheduling algorithm. Figure 2-25 shows the structure of log-based CDC. There are number of components like Data Transition, Loading & Scheduling Controller. Data quality and freshness is being analyzed by FIFS (First in First Schedule) and priority. The CPU utilization rate is monitor using scheduling controller and for assurance the better quality of data a feedback system is used to adjust scheduling policy. Data transition and loading component is responsible for transforming and cleaning the processed data in order to load to the DWH.

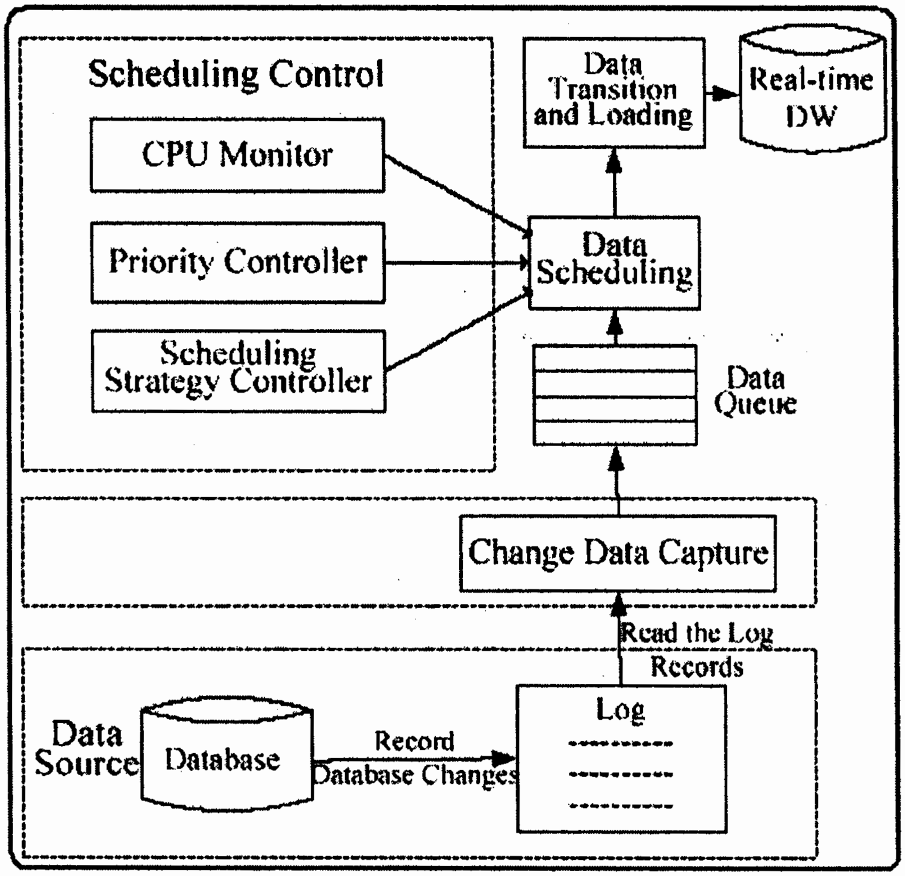


Figure 2.17 Framework of Log-based CDC [5]

Figure 2.18 shows the process of CDC on the basis of logs analysis. This process has log initialization, data dictionary establishment, loading the log file, analysis and collection of data.

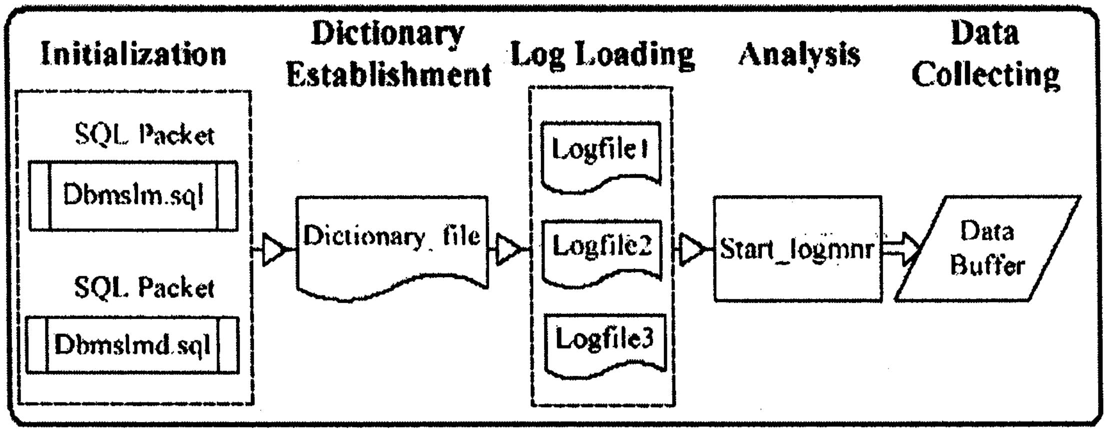


Figure 2.18: Online Log CDC [5]

Goals of this study are as follows:

1. To enhance the quality and freshness of data by using scheduling algorithm
2. Capture the freshest data on the basis of online log-based analysis.
3. Unnecessary data loading will be reduced
4. Real-time scheduling strategy for RTDWH.

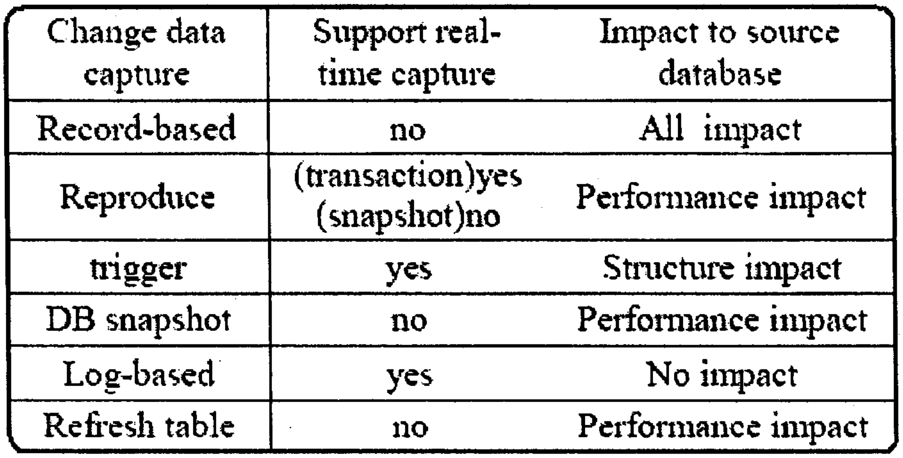
Assumptions considered in this study are as follows:

1. A method that capture the changed data without disturbing the foundation system.
2. Capture changed data within time.
3. Source systems' structure does not change.

The results of this study are shown below

Table 2.9 shows the comparison between the logbased capture methods and other real-time data capture methods.

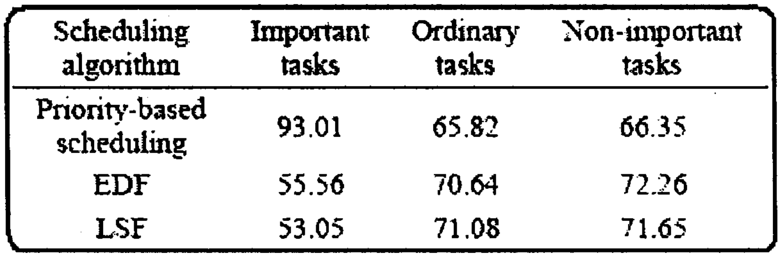
Table 2.9: Changed Data Capture of RTDWH



Change data capture	Support real-time capture	Impact to source database
Record-based	no	All impact
Reproduce	(transaction)yes (snapshot)no	Performance impact
trigger	yes	Structure impact
DB snapshot	no	Performance impact
Log-based	yes	No impact
Refresh table	no	Performance impact

Table 2.10 show the comparison of performance between scheduling algorithm used by data queue.

Table 2.10: The average success rate of different scheduling algorithm (%)



Scheduling algorithm	Important tasks	Ordinary tasks	Non-important tasks
Priority-based scheduling	93.01	65.82	66.35
EDF	55.56	70.64	72.26
LSF	53.05	71.08	71.65

Claims of S5

1. Booming structure and technique for data extraction for real-time DWH
2. Decision analysis will be extremely supported by processed data.
3. The task of scheduling method is improved.

Future Work mentioned in study S5

1. Despite Oracle DB's log, exercise other database logs for Change Data Capture.

P6 → Dr.Muhammad Younus Javed, Asim Nawaz "Data Load Distribution by Semi Real Time Data Warehouse" 2nd International Conference on Computer and Network Technology, pp. 556-560, 2010.

The main purpose of DWH is to support strategic decision making. The data is being extracted, transformed and loaded to such DWH, but this process takes much time to move data from primary source to DWH within given time frame. ETL gets data from heterogeneous & asynchronous source and moves it to the homogeneous environment. DWH is being refreshed in two ways off-line and real-time, off-line is time consuming and real-time overwhelming the sources with extra work load, So in this paper a technique is being proposed to distribute the load of extraction on source systems

Goals of this study are as follows:

1. To distribute the load of OLTP source data
2. To distribute data volume to be extracted in real-time and remaining on off-line.
3. To reduce the time window by utilizing ETL idle time.

Assumptions considered in this study are as follows:

1. Trm is time to be consumed on n/m in real-time.
2. Tcm is time to be consumed on (n-n/m) on off-line

The results of this study are shown below

ETL is time consuming and each step is dependent on one another if one takes much time then other step will have to wait until the first step finish its work. Extraction is totally different from conventional extraction technique due to its data load distribution.

- Source OLTP Systems.
- CDC (Change Data Capture) Mechanism.
- Transformation and Loading through Oracle Warehouse Builder (OWB).
- Data Warehouse (Implemented by Star Schema).
- Data Marts.
- End User Workstations.

Some data will be extracted in real-time fashion and remaining on off-line fashion. Information about which data must be extracted in real-time and which data will be extracted in off-line fashion.

Since DWH is being refreshed after 24 hours, so it is very clear that ETL is almost free for 24 hours and after 24 hours lot of burden will come on ETL when DWH is being refreshed. So there must be a technique which refreshes the DWH before 24 hours to overcome the load that will come after 24 hours.

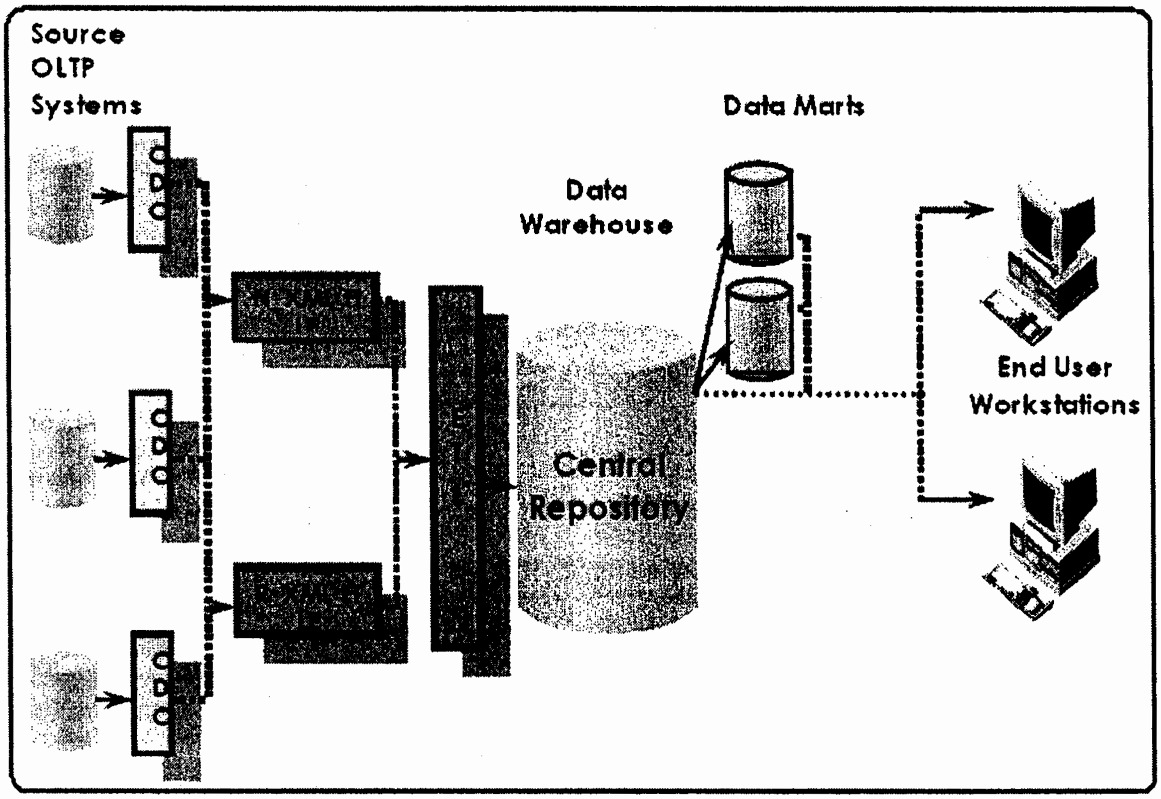


Figure 2.19 Semi RTDWH Architecture [6]

A database having n number of tables

$$T = \{t1, t2, t3, t4, \ldots\ldots, tn\}.$$

As it is very clear that DWH will be refreshed after 24 hours, or week, or after certain time period. Let we consider that a DWH will be refreshed after 24 hours. Let Te, (extraction time), Tt, (transformation time), Tl (loading time) time taken by ETL after 24 hours. Total time taken by the ETL process is

$$\mathbf{T = Te + Tt + Tl.}$$

In this paper, author first calculates the total time that will be taken by ETL. It has been analyzed that which source table will be loaded at real-time and which one will be loaded at off-line. n is the number of tables to be loaded, so m is the number of tables to be extracted in real-time. n/m will be extracted in real-time and remaining (n-n/m) will be extracted off-line.

Tables to be extracted:

- n = total number of tables
- m = number of tables to extracted in real time
- n/m = total number of tables to be extracted in real-time
- n-n/m = total number of tables to be extracted in off-line.

Time to be taken:

- Trm is time consumed by ETL process to complete its job in real-time manner on n/m source tables.
- Tcm is time consumed by ETL process to complete its job in conventional manner on remaining (n-n/m) source tables.

Equation for total Time to be taken:

- The percentage time taken by ETL process to finish its job in real-time fashion is given by the following equation.

$$\% \text{ Time Consumed} = (\text{Trm} \times 100) / \text{T};$$

- and % time taken by ETL process to finish its job in conventional manner is given by the following equation.

$$\% \text{ Time Consumed} = (\text{Tcm} \times 100) / \text{T};$$

- Percentage decrease in time for ETL process after 24 hours is given by the following equation.

% Reduction Time = [(T – Trm) x 100] / T;

33 % process of OLTP primary source is done in real-time and remaining of 67 % is done on off-line fashion. So in this way 33 % extraction is being reduced.

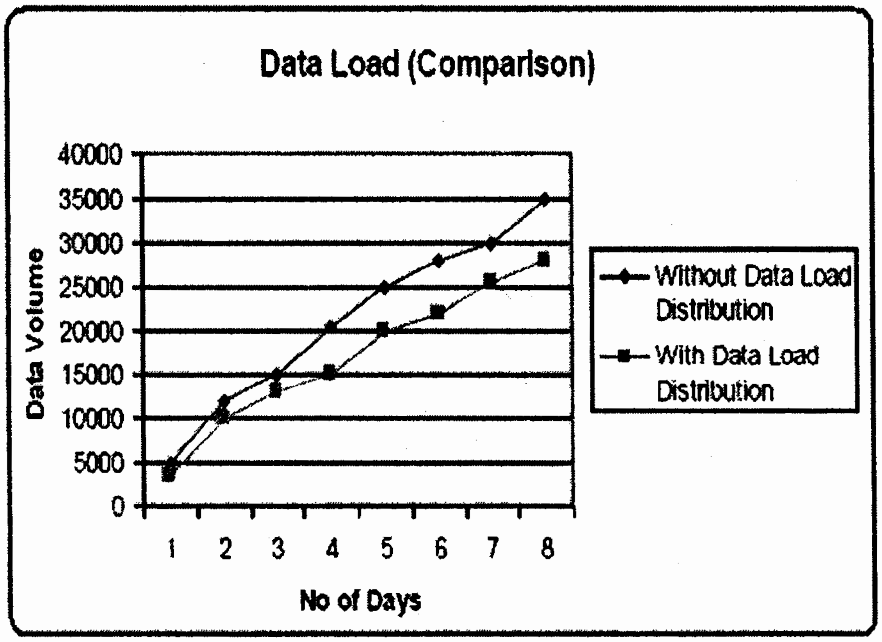


Figure 2.20: Data Load Comparison [6]

Claims of S6

1. Time consumed by ETL process is reduced in which some data is being extracted in real-time and remaining in conventional way.
2. Idle time of ETL is being utilized by distribution of data load for extraction.
3. Performance is being achieved by the data load distribution for extraction.
4. Only data is being extracted in real-time and normal way.

Future Work mentioned in study S6

1. Transformation and loading data in real-time and normal way.
2. Extraction of data with the support of flat files.

Chapter #3
Problem Analysis

3.1. Problem Statement

In most cases relations are quite large. Current industry ETL tools are not yet ready to do transformation with such a large volume of data within a satisfactory time frame. Data warehouse are being upgraded when all sources off.

For large volumes of data from multiple foundation relations, results from a source founds that one of the leading ETL tools took about ten hours for transformation and loading of six million records into the target table/relation, extracting from multiple foundation tables. Customers cannot afford such a long time to get data in the data warehouse.

It has been tried to address problems in extraction of data to be loaded into warehouse faster. The refresh of data warehouses using ETL tools is a well-explored field of research but, none of the research in this area has focused on metadata determined data warehouse refreshment.

The data warehouse will be refreshed according to the business rules. Only critical data will be loaded to the data warehouse during peak hours, when source system is on and working, which not only refreshes the data warehouse quickly but also supports the decision making process fast. This approach picks only critical data during peak hours to update the data warehouse.

It has been proposed to migrate the actions/tasks of transformation and loading the D/W house to the target database engine. The conventional ETL tools load target staging tables pulling from source table. The source and target table will have one-to-one relation with the same table (relation) structure. In this way, loading staging tables via ETL tools will be much simpler and faster.

3.2. Objectives

- Is DWH updated on near real-time basis a feasible solution?
- What help can ETL tool provides to support the procedure?
- What needs to be done for better and efficient DWH updation in context of solutions presented in literature?
- How much improvement can be made by updating real-time on basis of proposed solution?

The objective or goal of this thesis is to implement a technique for extraction of exact volume of data at the right time to the right palace according to business rules defined to achieve the business success.

Appendix A
References

Chapter #4
Proposed Framework & Results

4.1. Proposed Solution

The proposed methods aim at refreshing data warehouses faster and maintaining the batch cycle time shorter. This is suitable for large data warehouses with hundreds of subject areas and thousands of tables, where a load happens in a four-to-two hour window. Instead of processing complex data transformation in an expensive and proprietary ETL engine, my theory is based on the solid factual proof, I found that it took 6 hours to load about 3 million rows in a consequential table in the data warehouse to do transformation and loading using a commercial ETL tool.

It has been proposed the addition of a supplementary structure called a staging subject area in the data warehouse environment as opposed to outside the warehouse. This research discusses how to make batch jobs for the diagnostic subject area refresh at the right time per service-level agreement with the customers. With batch process it is being proposed to increase the regularity of batch runs to shorten the latency or delay, optimize the load process, omit the jobs that do not get new data from starting place and do incremental refresh for the tables that have fresh data in the foundation which minimizes the processing time.

4.2. Objectives and Methodology

The objectives of this research are as follows:

i. Intend to propose highly-efficient method for near real-time data warehousing, focusing on a metadata model determined and using DBMS software based functionality. The proposed methods enable batch cycle runs faster than what they are doing now.

ii. To build up a strategy which ensures batch loads are made shorter from nightly to 4-hour increments to one third that rate or even shorter than that.

iii. Approach is based on the following two step theory. First, data is acquired from the ERP and then loaded into the staging subject area tables in the data warehouse via different commercial ETL tools, as well as database utilities, and also by ERP and by any other ETL tool.

iv. To complete batch cycle runs faster, it is being proposed two methods for diagnostic subject area refreshment in data warehouses:

a) Skip jobs when foundation data has not changed.

b) Do incremental refreshes of only that or those tables for which foundation data has been changed

v. The data warehouse refresh techniques proposed in this research work fine for doing both incremental loads from foundation to auxiliary subject areas in data warehouse and then loading the diagnostic subject areas. For auxiliary table refresh flat files or DTS (data transformation services) tools are used to load the staging tables and no transformation is needed – only cleansing and formatting needed – this makes any vendor tool or home grown tool's task easier and efficient. The diagnostic subject area refresh, on the other hand, is straight forward as the tables will be loaded only when staging table data has changed and will be loaded via DBMS-specific stored procedures.

Finally, Hanson and Willshire [21] consider an ETL setting like ours, and define the concept of auxiliary tables in a data warehouse in order to enable efficient resumption interrupted warehouse loads. Although similar in overall spirit, the use of auxiliary tables in Hanson and Willshire's paper [21] is different from my view point of using the auxiliary tables and does not consider full utilization of auxiliary tables that I consider in this research. In addition, I consider a more effective use of auxiliary tables than those considered by Hanson and Willshire [21].

vi. Propose to extract only critical data from foundation systems using conventional ETL tools to load the staging subject areas in the data warehouse. In this process the ETL tools will be used only to load the staging tables without any kind of transformation tasks. As soon as staging tables are refreshed the data warehouse software will be used to do all kinds of complex transformation, including summarization, aggregation and other computations to load the actual data warehouse (diagnostic subject areas). There are several advantages of staging foundation data in auxiliary tables first and doing the diagnostic subject area refresh based on data changing in the staging tables. First, a staging tables refresh will be faster as foundation and target table relations will be one-to-one and there is no transformation involved in this step. So, continuous flow of foundation data via incremental data feed can be made instantaneously. On the other hand, actual data warehouse table refreshing can be metadata determined based on availability of new data in the staging tables. In staging subject areas of data warehouse, data is acquired incrementally and identified by timestamp. The diagnostic subject area can be updated incrementally via batch-determined update procedures. The diagnostic tables could be refreshed from scratch with full data files from the foundation when needed. Normally, incremental updates are to be performed in terms of refresh cycles, so that the batch cycles can be made faster and shorter. There are many techniques to improve performance of data warehouse queries, ranging from classical database indexes and query optimization. A number of indexing strategies has been proposed for data warehouses in literature and are heavily used in practice.

4.3. Goals

Main goals of proposed solutions are as follows:

1. Refresh DWH using near real-time or on right time.
2. Reduce the extraction time.
3. Extract only required/particular volume of data which depends on the business rules defined.

4.4. Limitation on Lab Environment

Following limitations/constraints exist for experimental setup and validation:

1. There are limited numbers of tables in experimental warehouse.
2. Tables have same schemas and somewhat same amount of data.
3. For the sake of experimentation, the student data has been declared as critical due to its characteristics.

4.5. Proposed Architecture

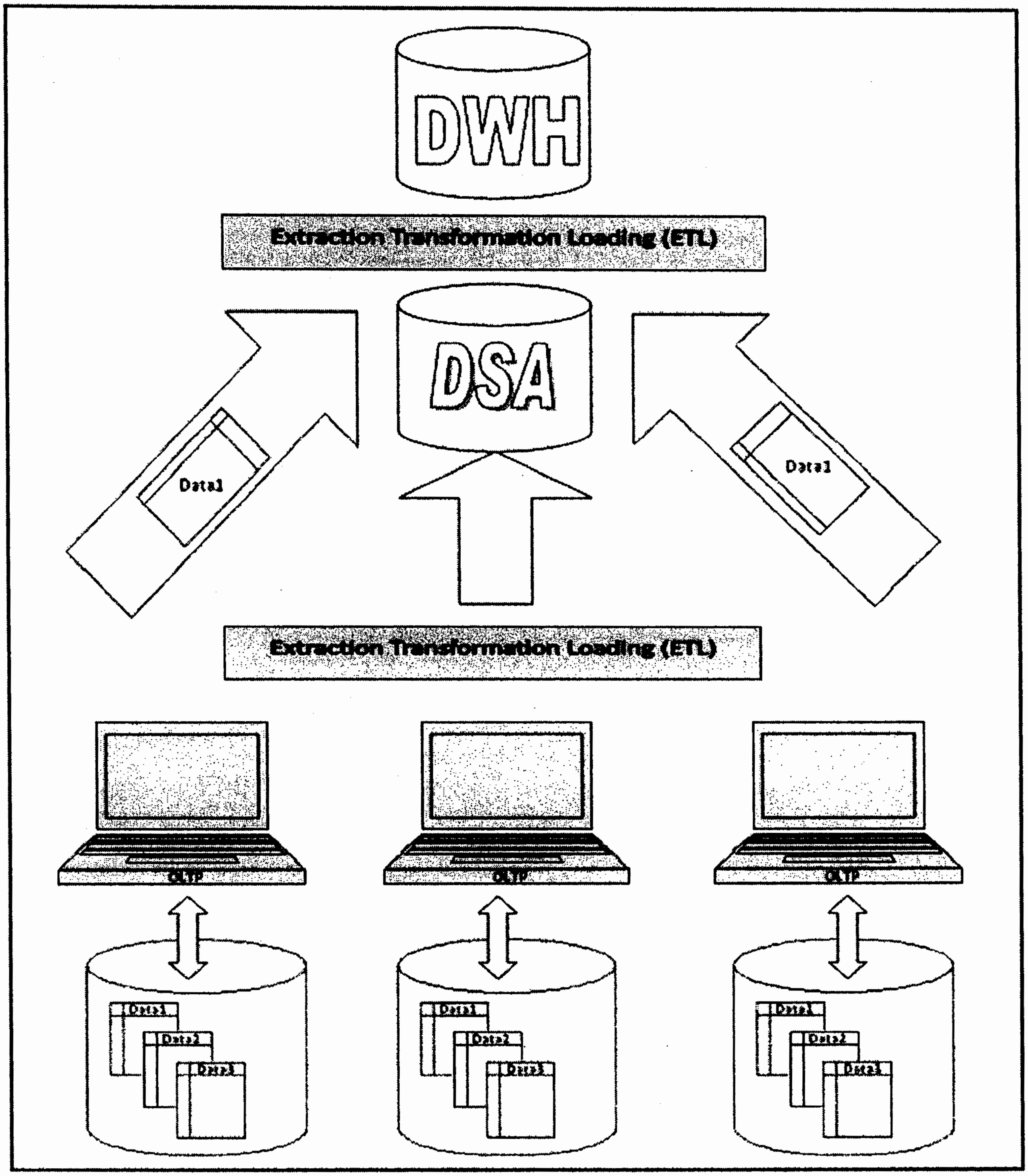
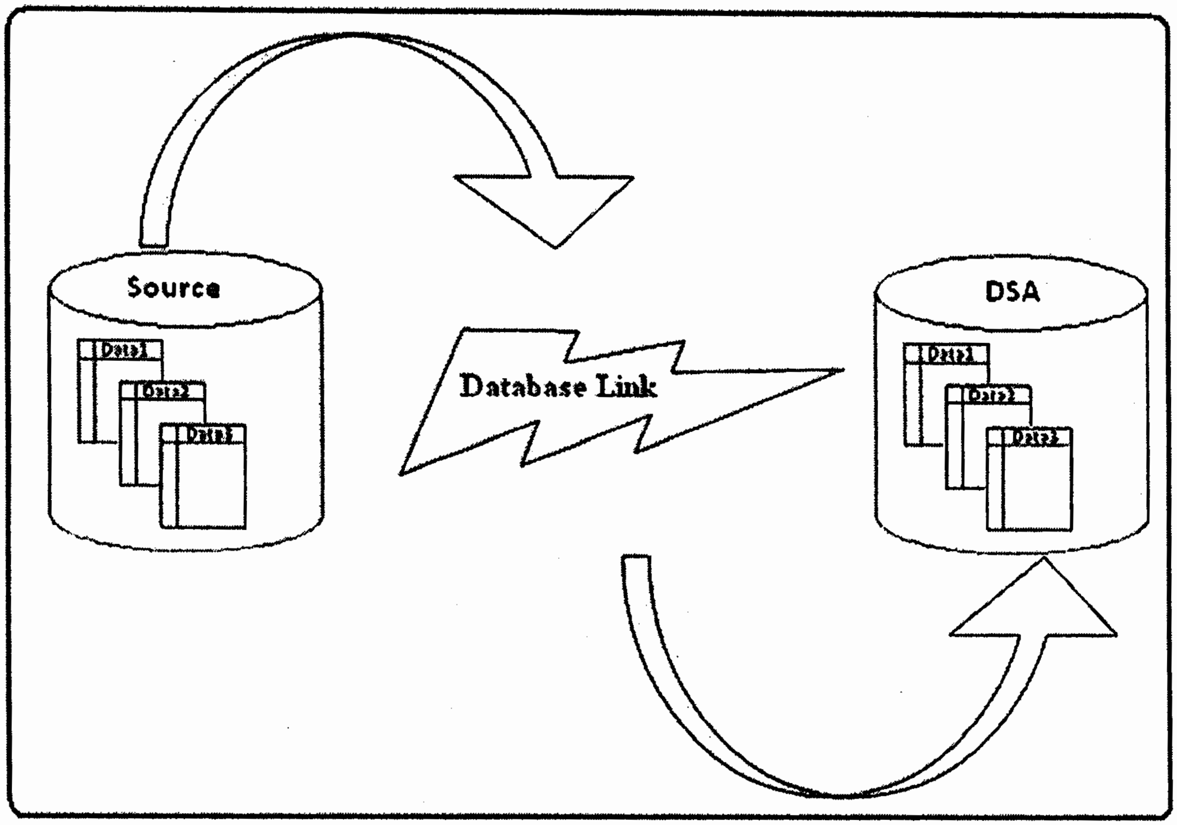
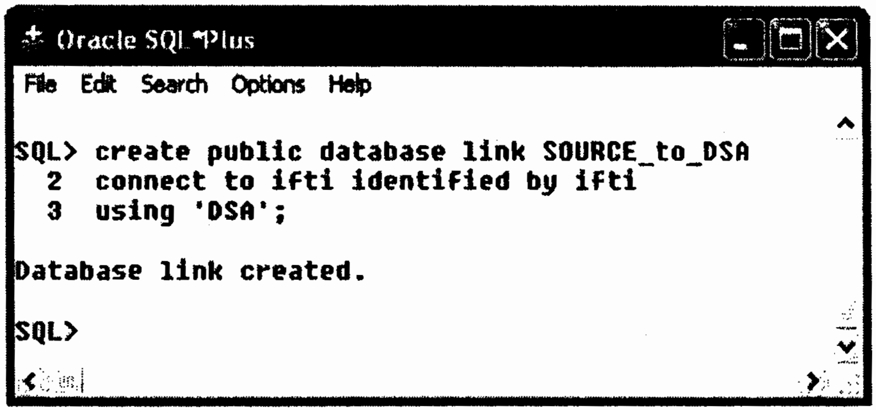

DWH
Extraction Transformation Loading (ETL)
DSA
Data1
Data1
Extraction Transformation Loading (ETL)
OLTP
OLTP
OLTP
Data1
Data2
Data3
Data1
Data2
Data3
Data1
Data2
Data3

Figure 4-1: Proposed Architecture



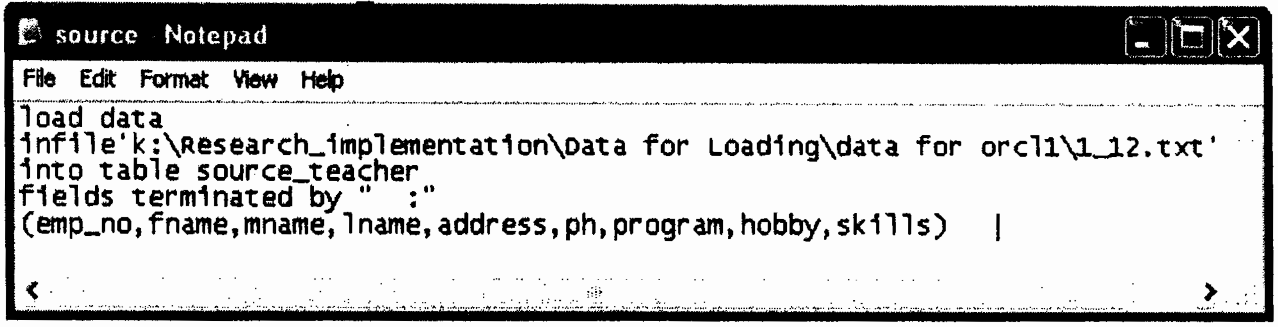


```
SQL> create public database link SOURCE_to_DSA
  2  connect to ifti identified by ifti
  3  using 'DSA';

Database link created.

SQL>
```

4.6. Data Loading

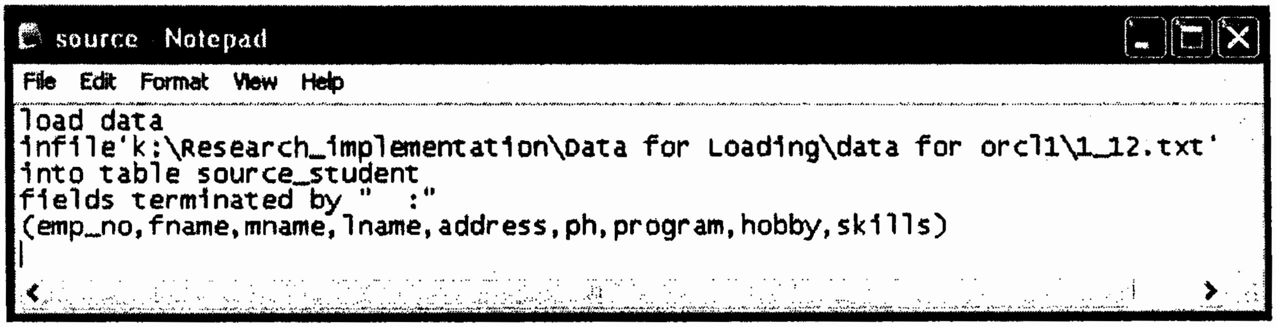
Using SQL Loader to load data, there must be a control file before loading data into target table. The following snapshots show the respective control files. There are three different control files for different tables.

- **Control file for source_teacher:**



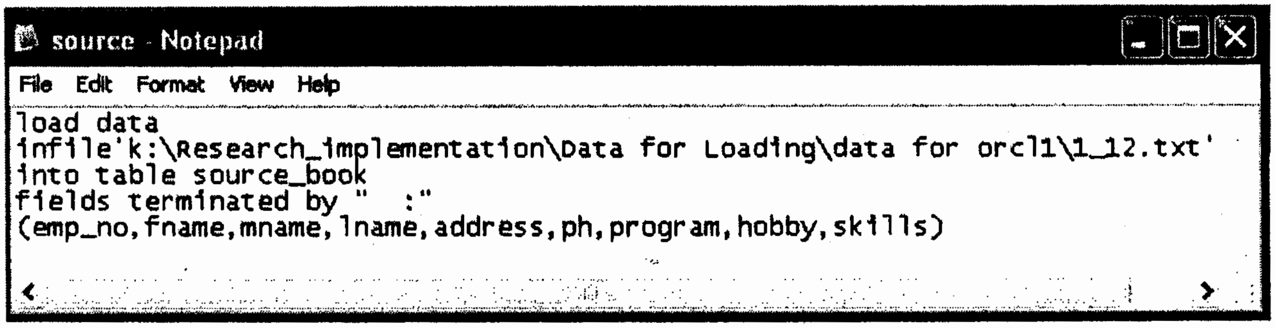
```
load data
infile'k:\Research_implementation\Data for Loading\data for orcl1\1_12.txt'
into table source_teacher
fields terminated by "  :"
(emp_no,fname,mname,lname,address,ph,program,hobby,skills)
```

- **Control file for source_student**



```
load data
infile'k:\Research_implementation\Data for Loading\data for orcl1\1_12.txt'
into table source_student
fields terminated by "  :"
(emp_no,fname,mname,lname,address,ph,program,hobby,skills)
```

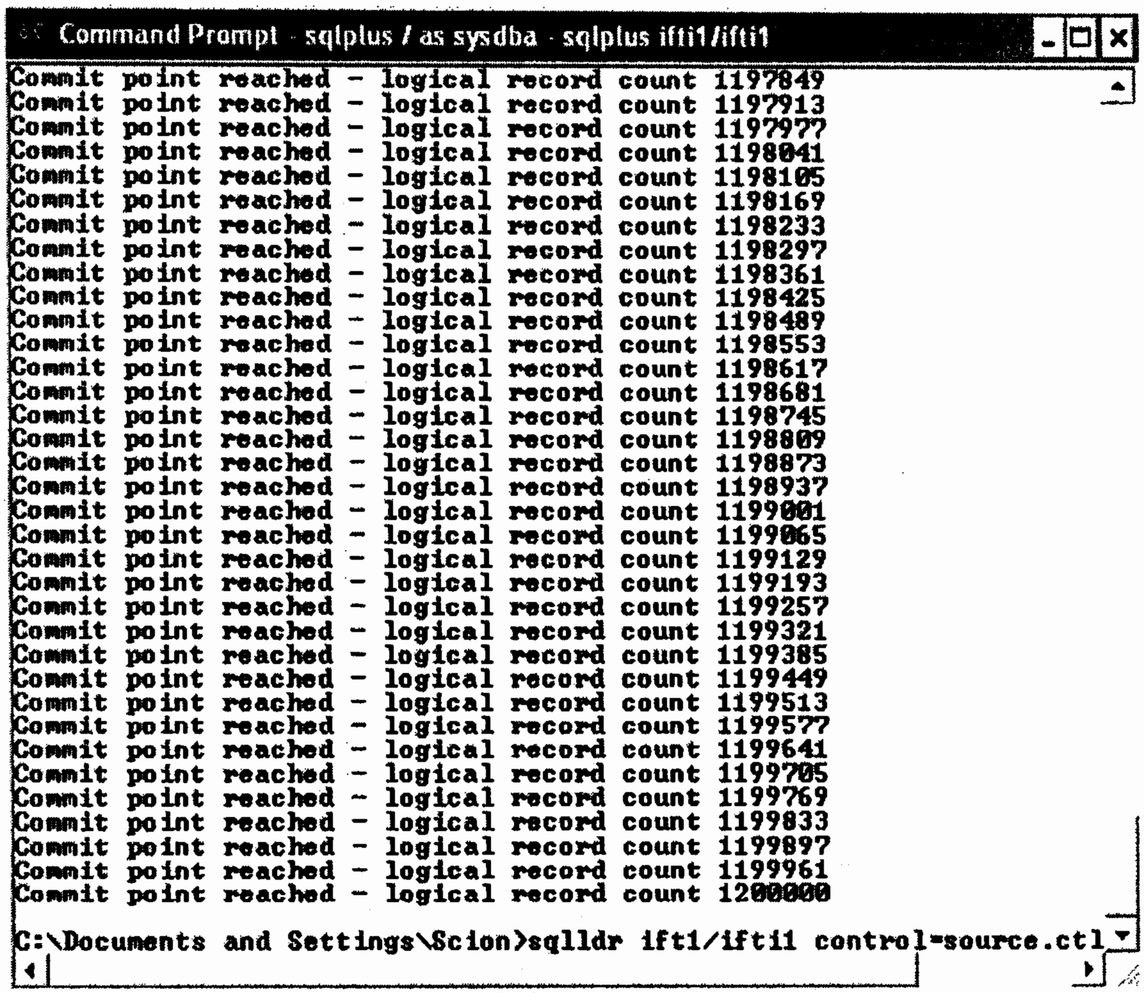
- **Control file for source_book**



```
load data
infile'k:\Research_implementation\Data for Loading\data for orcl1\1_12.txt'
into table source_book
fields terminated by "  :"
(emp_no,fname,mname,lname,address,ph,program,hobby,skills)
```

- **Loading data using SQL Loader:**
  (c:>sqlldr username/password control=source.ctl)

The following snapshot shows the process of loading data into a specific table using the commond, c:>sqlldr ifti/ifti control=source.ctl

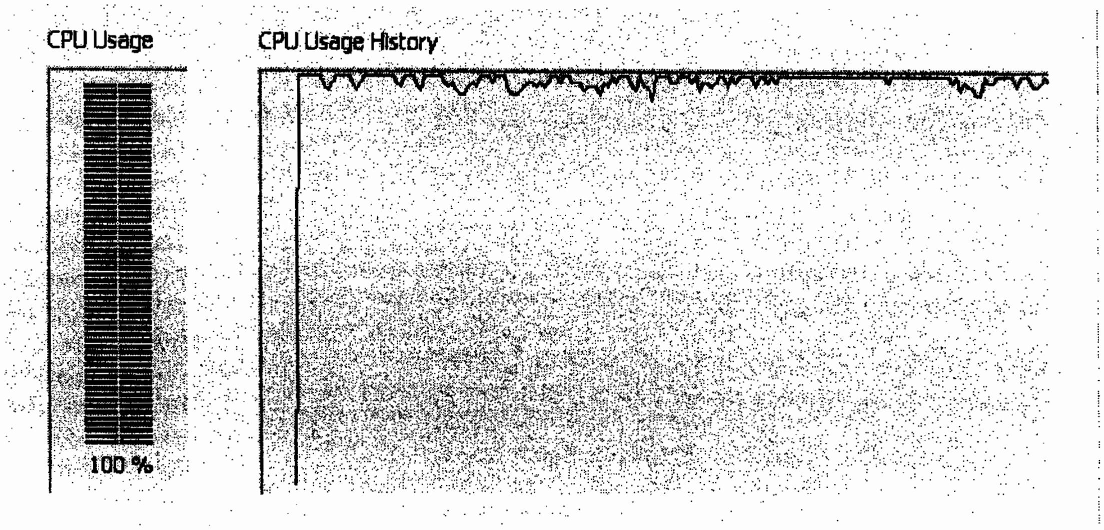


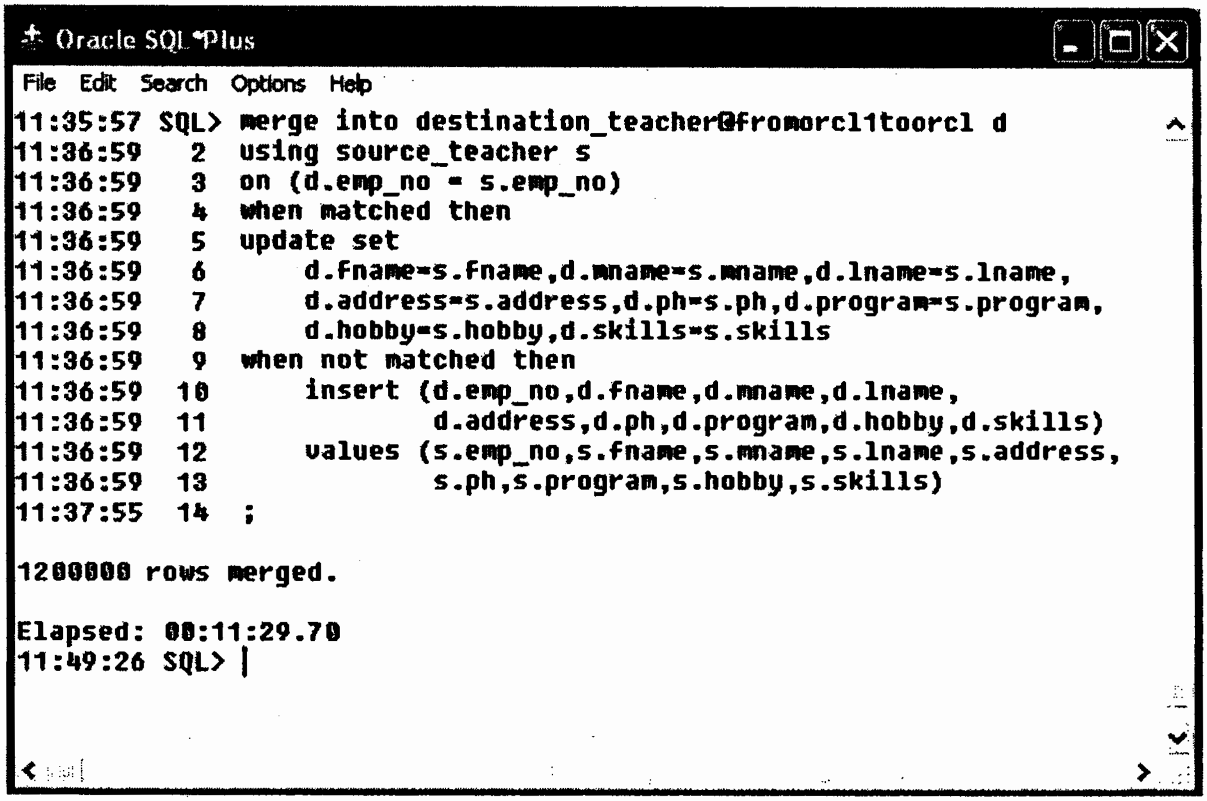
```
Commit point reached - logical record count 1197849
Commit point reached - logical record count 1197913
Commit point reached - logical record count 1197977
Commit point reached - logical record count 1198041
Commit point reached - logical record count 1198105
Commit point reached - logical record count 1198169
Commit point reached - logical record count 1198233
Commit point reached - logical record count 1198297
Commit point reached - logical record count 1198361
Commit point reached - logical record count 1198425
Commit point reached - logical record count 1198489
Commit point reached - logical record count 1198553
Commit point reached - logical record count 1198617
Commit point reached - logical record count 1198681
Commit point reached - logical record count 1198745
Commit point reached - logical record count 1198809
Commit point reached - logical record count 1198873
Commit point reached - logical record count 1198937
Commit point reached - logical record count 1199001
Commit point reached - logical record count 1199065
Commit point reached - logical record count 1199129
Commit point reached - logical record count 1199193
Commit point reached - logical record count 1199257
Commit point reached - logical record count 1199321
Commit point reached - logical record count 1199385
Commit point reached - logical record count 1199449
Commit point reached - logical record count 1199513
Commit point reached - logical record count 1199577
Commit point reached - logical record count 1199641
Commit point reached - logical record count 1199705
Commit point reached - logical record count 1199769
Commit point reached - logical record count 1199833
Commit point reached - logical record count 1199897
Commit point reached - logical record count 1199961
Commit point reached - logical record count 1200000

C:\Documents and Settings\Scion>sqlldr ifti1/ifti1 control=source.ctl
```

4.7. Extraction of whole Data during Peak Hours

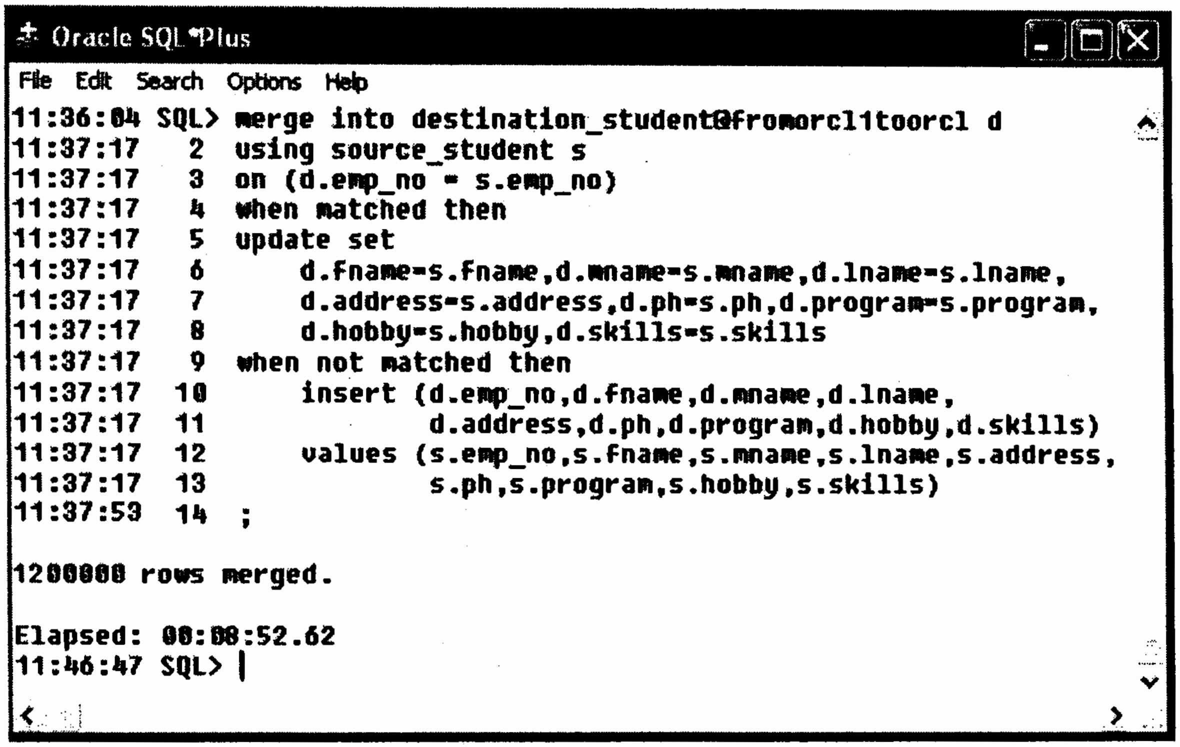
- **Day One Data:**

The CPU usage shows that the source system is fully busy up to 100% when whole data is being extracted and move to DSA

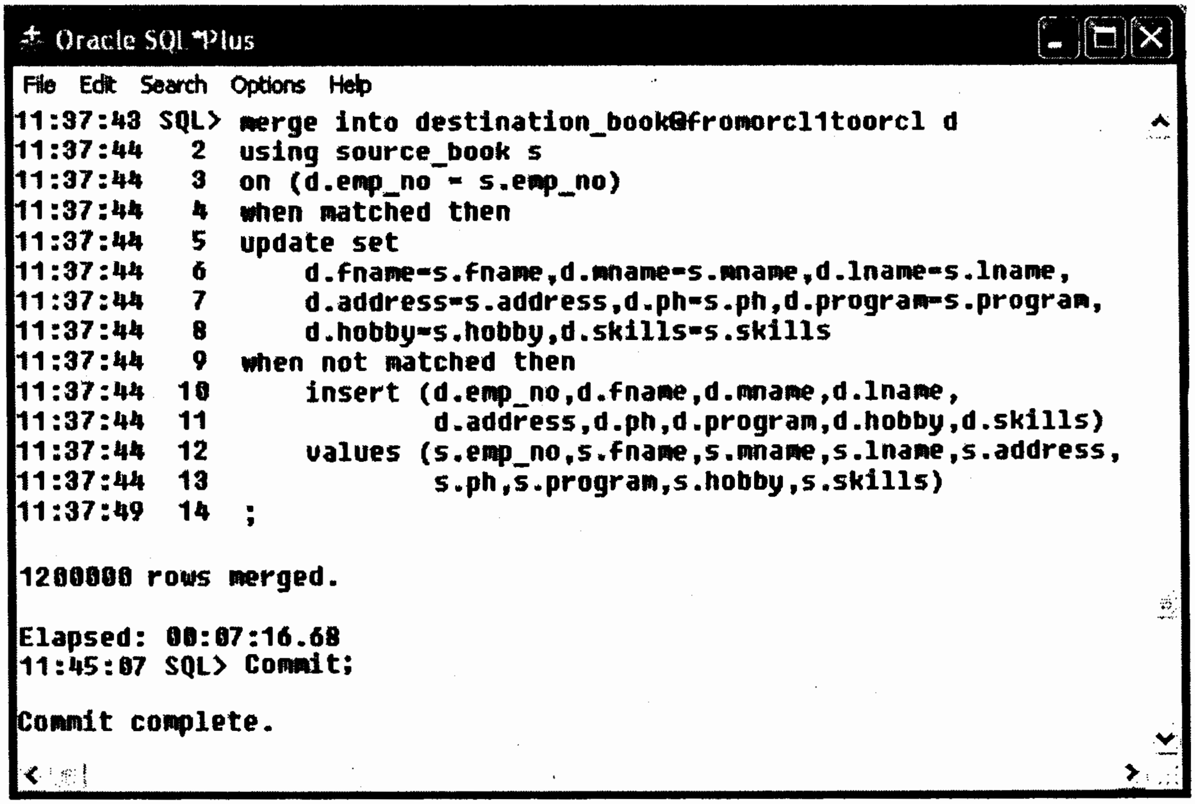




```
11:35:57 SQL> merge into destination_teacher@fromorcl1toorcl d
11:36:59   2  using source_teacher s
11:36:59   3  on (d.emp_no = s.emp_no)
11:36:59   4  when matched then
11:36:59   5  update set
11:36:59   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
11:36:59   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
11:36:59   8      d.hobby=s.hobby,d.skills=s.skills
11:36:59   9  when not matched then
11:36:59  10      insert (d.emp_no,d.fname,d.mname,d.lname,
11:36:59  11              d.address,d.ph,d.program,d.hobby,d.skills)
11:36:59  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
11:36:59  13              s.ph,s.program,s.hobby,s.skills)
11:37:55  14  ;

1200000 rows merged.

Elapsed: 00:11:29.70
11:49:26 SQL> |
```



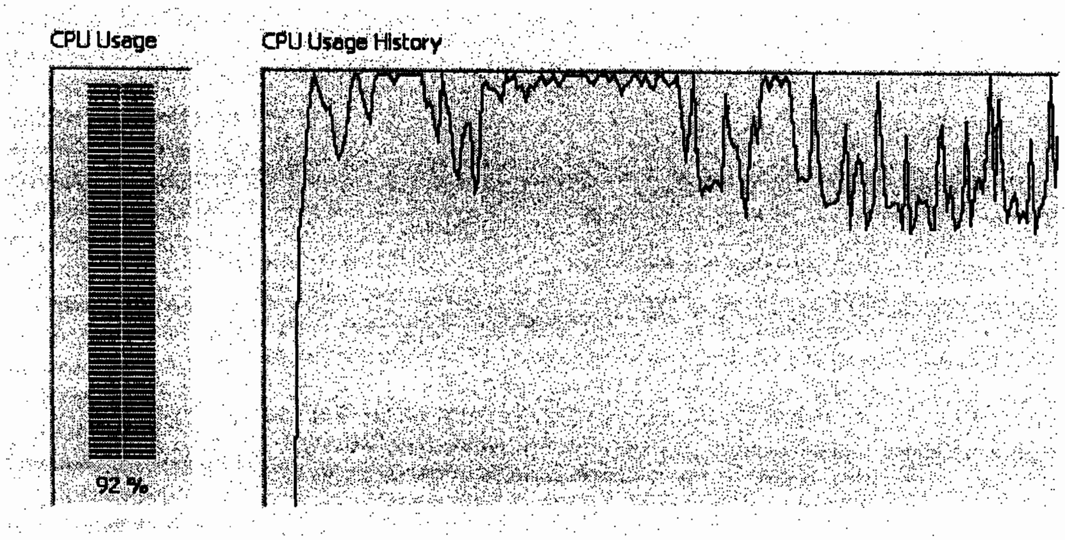
```
11:36:04 SQL> merge into destination_student@fromorcl1toorcl d
11:37:17   2  using source_student s
11:37:17   3  on (d.emp_no = s.emp_no)
11:37:17   4  when matched then
11:37:17   5  update set
11:37:17   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
11:37:17   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
11:37:17   8      d.hobby=s.hobby,d.skills=s.skills
11:37:17   9  when not matched then
11:37:17  10      insert (d.emp_no,d.fname,d.mname,d.lname,
11:37:17  11              d.address,d.ph,d.program,d.hobby,d.skills)
11:37:17  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
11:37:17  13              s.ph,s.program,s.hobby,s.skills)
11:37:53  14  ;

1200000 rows merged.

Elapsed: 00:08:52.62
11:46:47 SQL> |
```

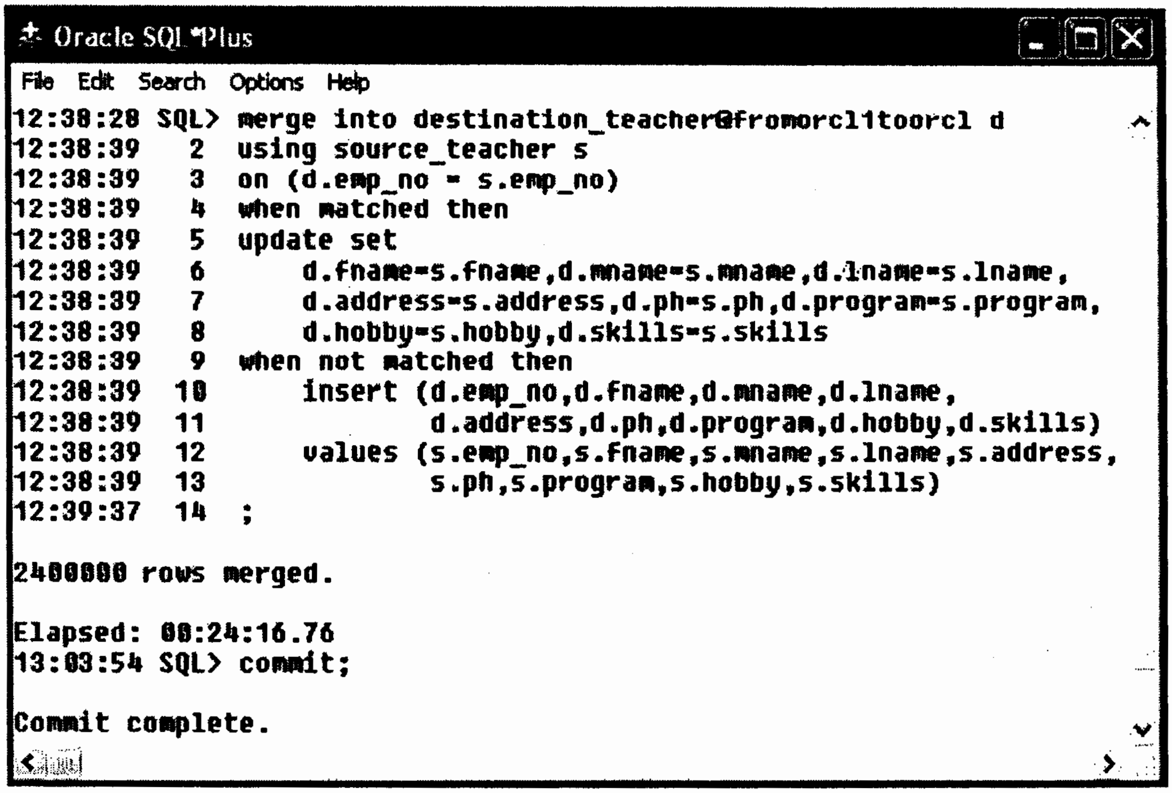


```
11:37:43 SQL> merge into destination_book@fromorcl1toorcl d
11:37:44   2  using source_book s
11:37:44   3  on (d.emp_no = s.emp_no)
11:37:44   4  when matched then
11:37:44   5  update set
11:37:44   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
11:37:44   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
11:37:44   8      d.hobby=s.hobby,d.skills=s.skills
11:37:44   9  when not matched then
11:37:44  10      insert (d.emp_no,d.fname,d.mname,d.lname,
11:37:44  11              d.address,d.ph,d.program,d.hobby,d.skills)
11:37:44  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
11:37:44  13              s.ph,s.program,s.hobby,s.skills)
11:37:49  14  ;

1200000 rows merged.

Elapsed: 00:07:16.68
11:45:07 SQL> Commit;

Commit complete.
```

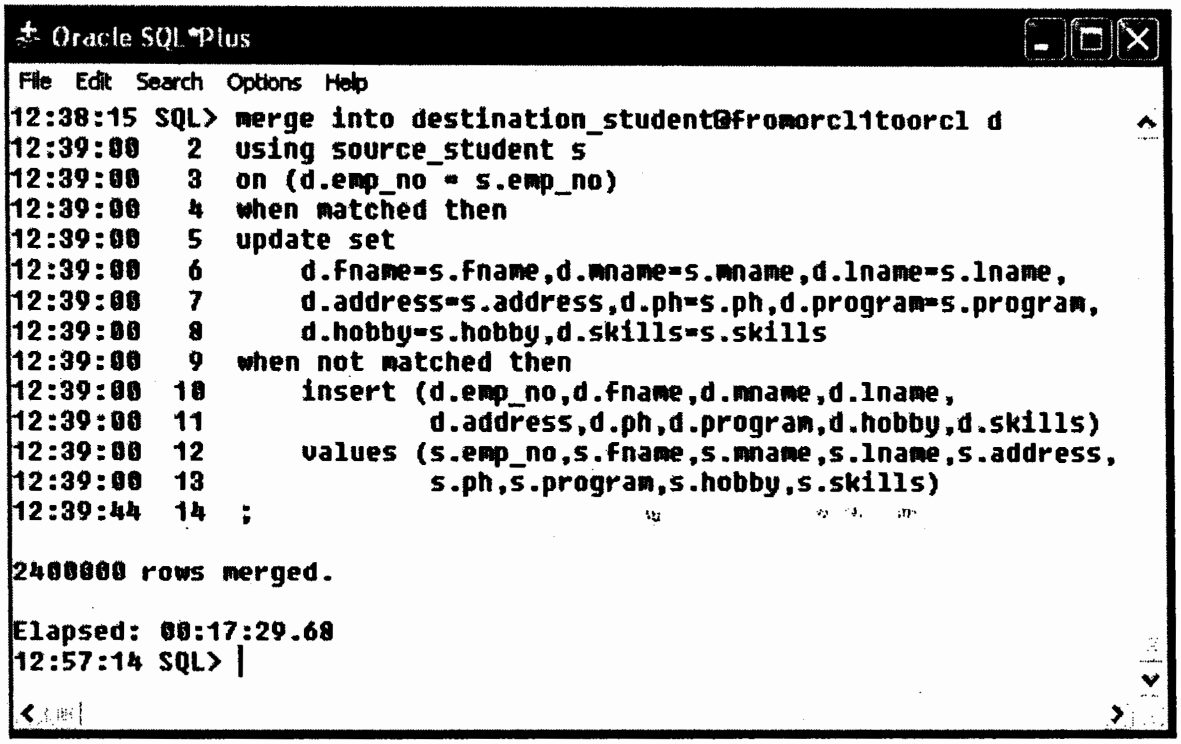
- **Day Two Data:**

The CPU usage shows that the source system is fully busy up to 92% when whole data is being extracted and move to DSA.

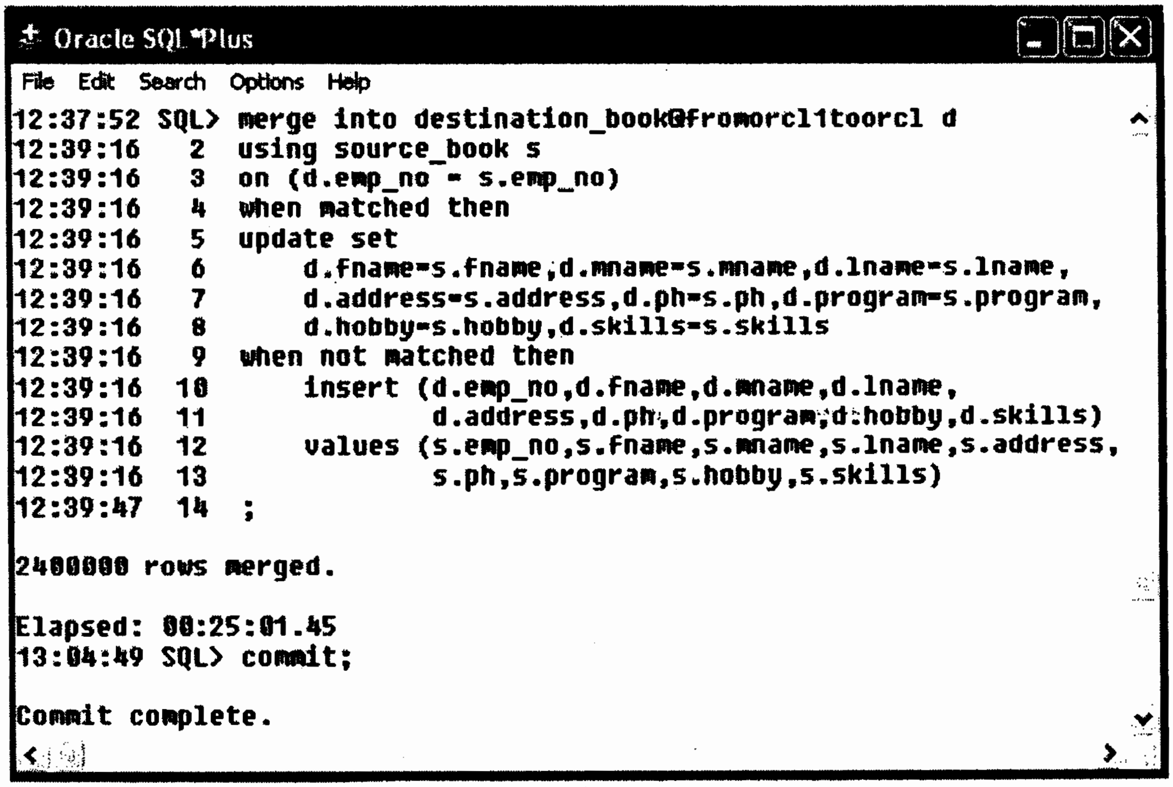




```
12:38:28 SQL> merge into destination_teacher@fromorcl1toorcl d
12:38:39   2  using source_teacher s
12:38:39   3  on (d.emp_no = s.emp_no)
12:38:39   4  when matched then
12:38:39   5  update set
12:38:39   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
12:38:39   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
12:38:39   8      d.hobby=s.hobby,d.skills=s.skills
12:38:39   9  when not matched then
12:38:39  10      insert (d.emp_no,d.fname,d.mname,d.lname,
12:38:39  11              d.address,d.ph,d.program,d.hobby,d.skills)
12:38:39  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
12:38:39  13              s.ph,s.program,s.hobby,s.skills)
12:39:37  14  ;

2400000 rows merged.

Elapsed: 00:24:16.76
13:03:54 SQL> commit;

Commit complete.
```



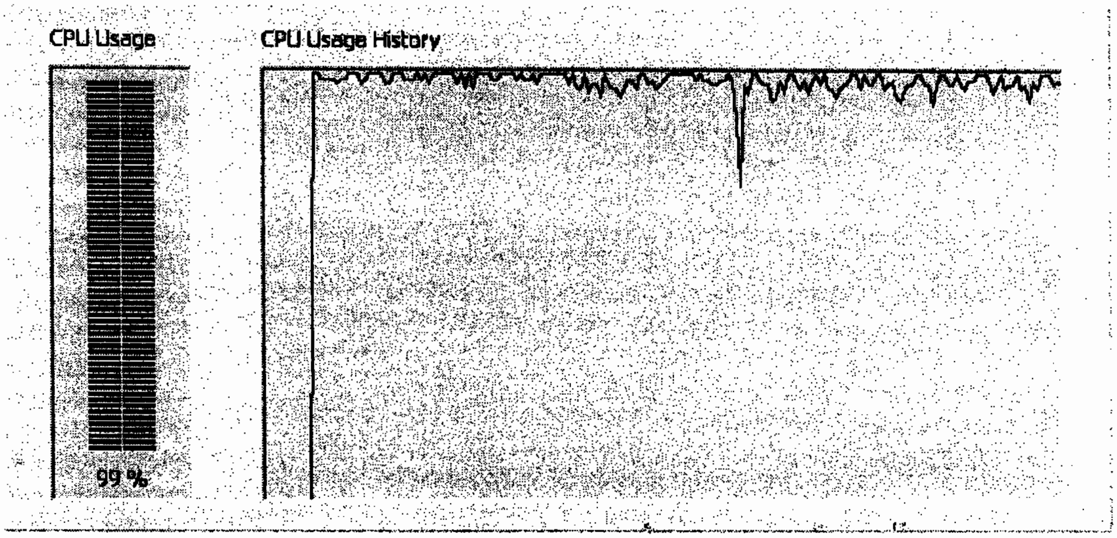
```
12:38:15 SQL> merge into destination_student@fromorcl1toorcl d
12:39:00   2  using source_student s
12:39:00   3  on (d.emp_no = s.emp_no)
12:39:00   4  when matched then
12:39:00   5  update set
12:39:00   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
12:39:00   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
12:39:00   8      d.hobby=s.hobby,d.skills=s.skills
12:39:00   9  when not matched then
12:39:00  10      insert (d.emp_no,d.fname,d.mname,d.lname,
12:39:00  11              d.address,d.ph,d.program,d.hobby,d.skills)
12:39:00  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
12:39:00  13              s.ph,s.program,s.hobby,s.skills)
12:39:44  14  ;

2400000 rows merged.

Elapsed: 00:17:29.68
12:57:14 SQL> |
```

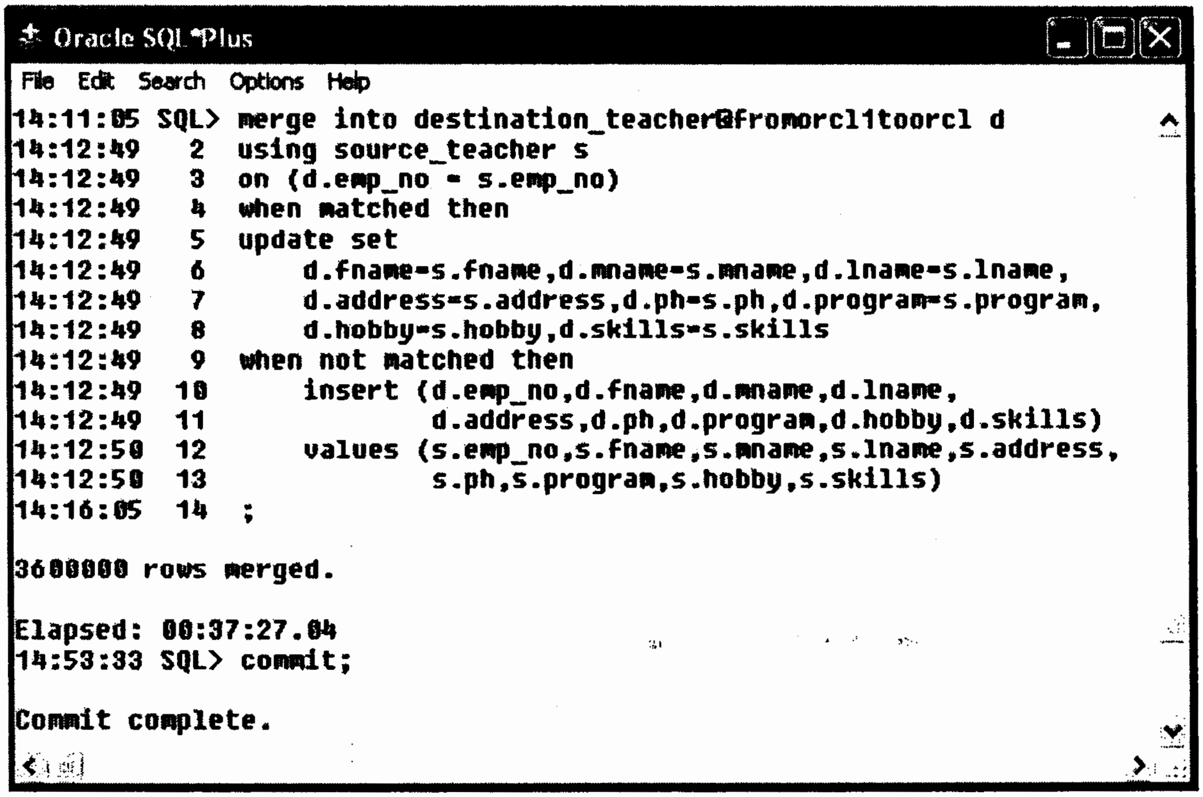


```
12:37:52 SQL> merge into destination_book@fromorcl1toorcl d
12:39:16   2  using source_book s
12:39:16   3  on (d.emp_no = s.emp_no)
12:39:16   4  when matched then
12:39:16   5  update set
12:39:16   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
12:39:16   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
12:39:16   8      d.hobby=s.hobby,d.skills=s.skills
12:39:16   9  when not matched then
12:39:16  10      insert (d.emp_no,d.fname,d.mname,d.lname,
12:39:16  11              d.address,d.ph,d.program,d.hobby,d.skills)
12:39:16  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
12:39:16  13              s.ph,s.program,s.hobby,s.skills)
12:39:47  14  ;

2400000 rows merged.

Elapsed: 00:25:01.45
13:04:49 SQL> commit;

Commit complete.
```

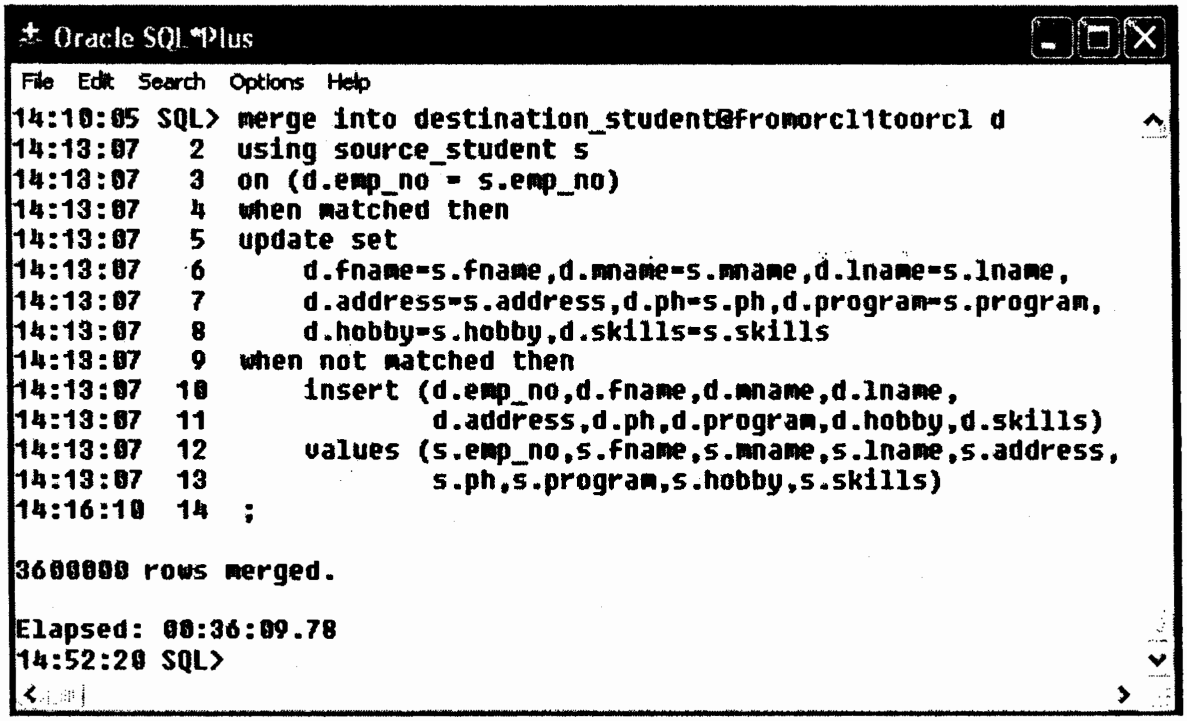
- **Day Three Data:**

The CPU usage shows that the source system is fully busy up to 99% when whole data is being extracted and move to DSA.

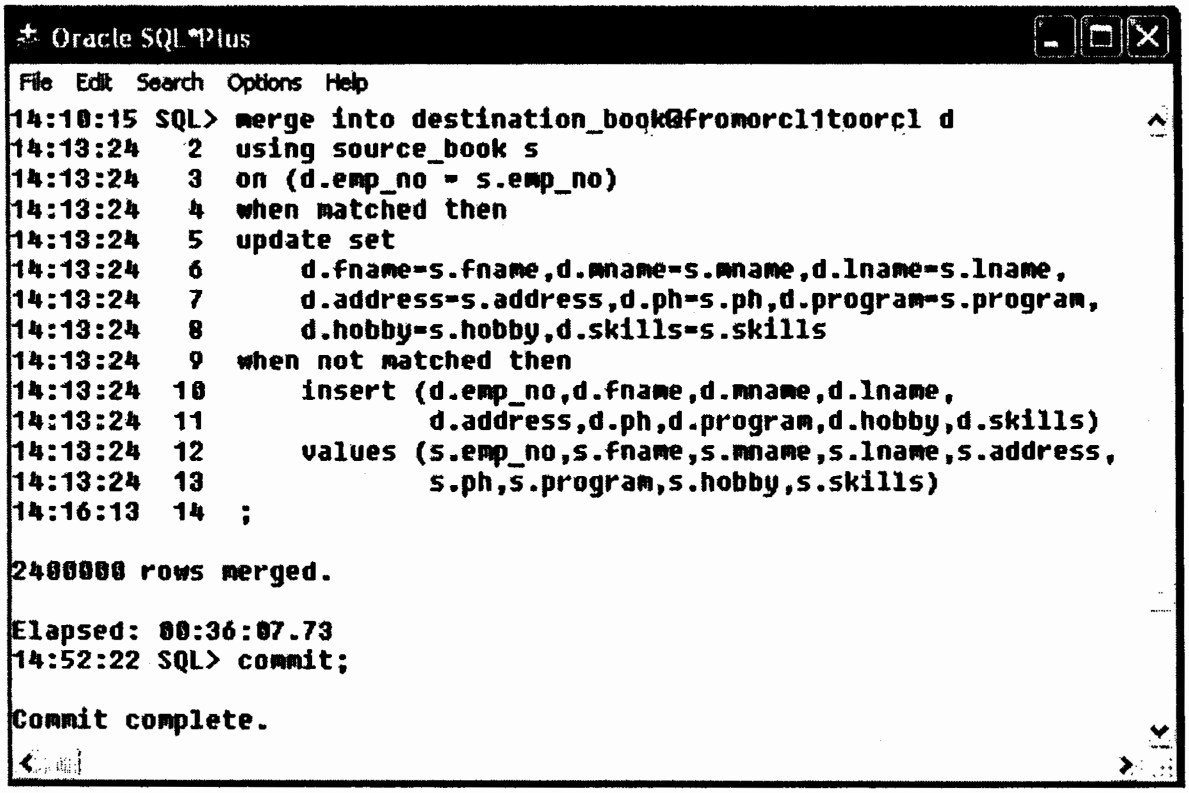




```
14:11:05 SQL> merge into destination_teacher@fromorcl1toorcl d
14:12:49   2  using source_teacher s
14:12:49   3  on (d.emp_no = s.emp_no)
14:12:49   4  when matched then
14:12:49   5  update set
14:12:49   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
14:12:49   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
14:12:49   8      d.hobby=s.hobby,d.skills=s.skills
14:12:49   9  when not matched then
14:12:49  10      insert (d.emp_no,d.fname,d.mname,d.lname,
14:12:49  11              d.address,d.ph,d.program,d.hobby,d.skills)
14:12:50  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
14:12:50  13              s.ph,s.program,s.hobby,s.skills)
14:16:05  14  ;

3600000 rows merged.

Elapsed: 00:37:27.04
14:53:33 SQL> commit;

Commit complete.
```



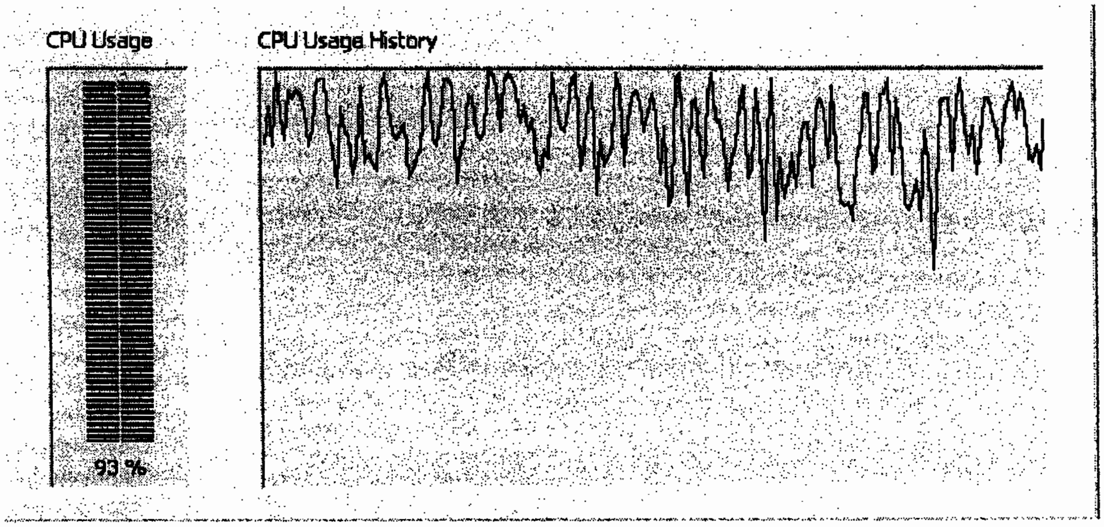
```
14:10:05 SQL> merge into destination_student@fromorcl1toorcl d
14:13:07   2  using source_student s
14:13:07   3  on (d.emp_no = s.emp_no)
14:13:07   4  when matched then
14:13:07   5  update set
14:13:07   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
14:13:07   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
14:13:07   8      d.hobby=s.hobby,d.skills=s.skills
14:13:07   9  when not matched then
14:13:07  10      insert (d.emp_no,d.fname,d.mname,d.lname,
14:13:07  11              d.address,d.ph,d.program,d.hobby,d.skills)
14:13:07  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
14:13:07  13              s.ph,s.program,s.hobby,s.skills)
14:16:10  14  ;

36000000 rows merged.

Elapsed: 00:36:09.78
14:52:20 SQL>
```

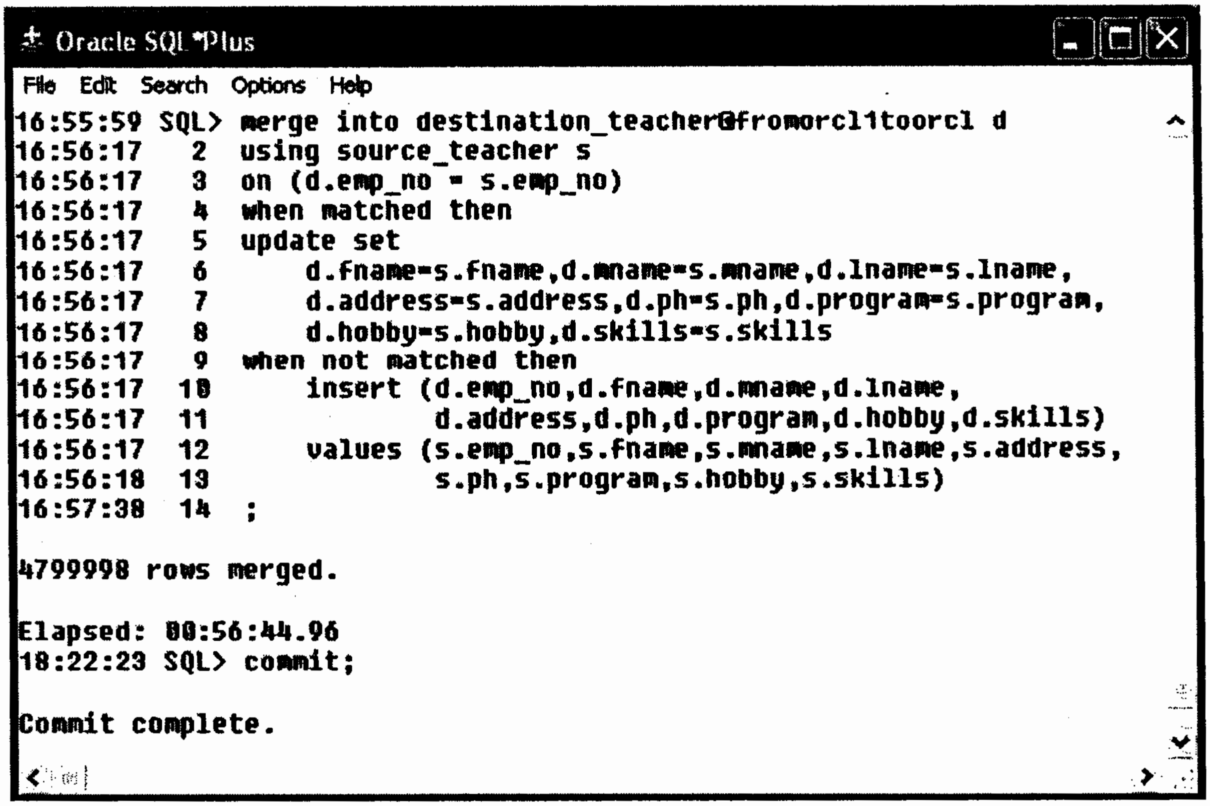


```
14:10:15 SQL> merge into destination_book@fromorcl1toorcl d
14:13:24   2  using source_book s
14:13:24   3  on (d.emp_no = s.emp_no)
14:13:24   4  when matched then
14:13:24   5  update set
14:13:24   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
14:13:24   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
14:13:24   8      d.hobby=s.hobby,d.skills=s.skills
14:13:24   9  when not matched then
14:13:24  10      insert (d.emp_no,d.fname,d.mname,d.lname,
14:13:24  11              d.address,d.ph,d.program,d.hobby,d.skills)
14:13:24  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
14:13:24  13              s.ph,s.program,s.hobby,s.skills)
14:16:13  14  ;

24000000 rows merged.

Elapsed: 00:36:07.73
14:52:22 SQL> commit;

Commit complete.
```

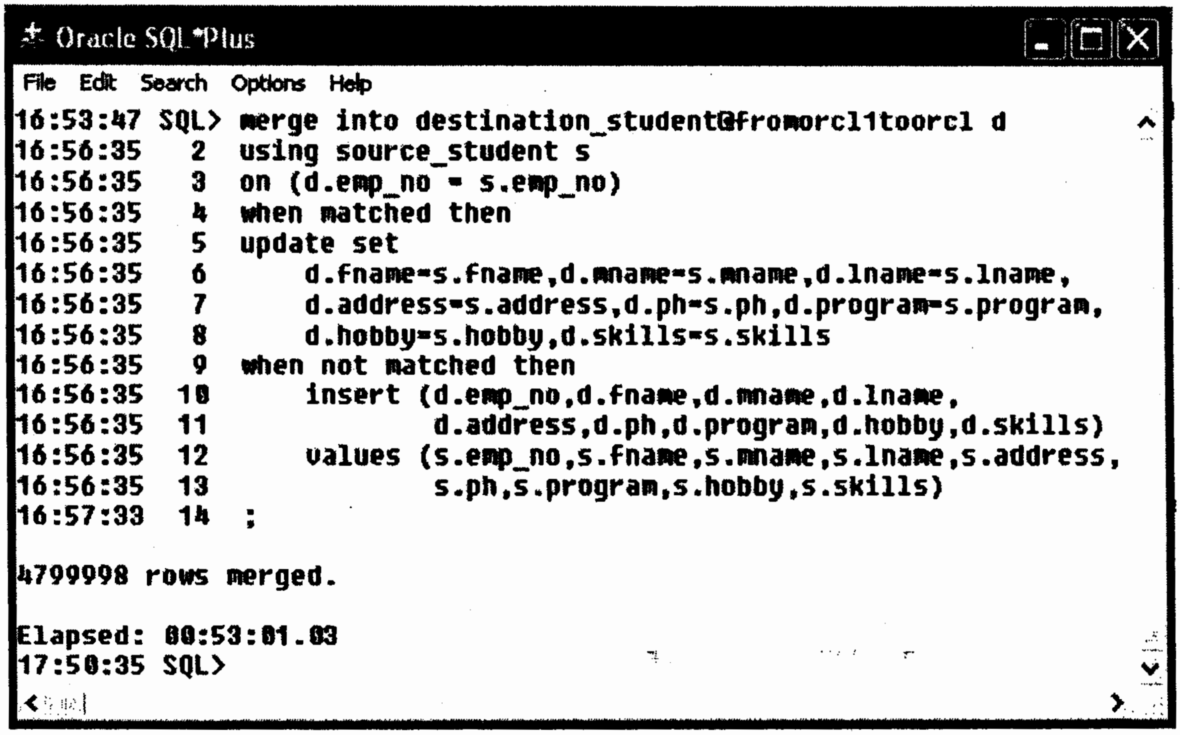
- **Day Four Data:**

The CPU usage shows that the source system is fully busy up to 93% when whole data is being extracted and move to DSA.

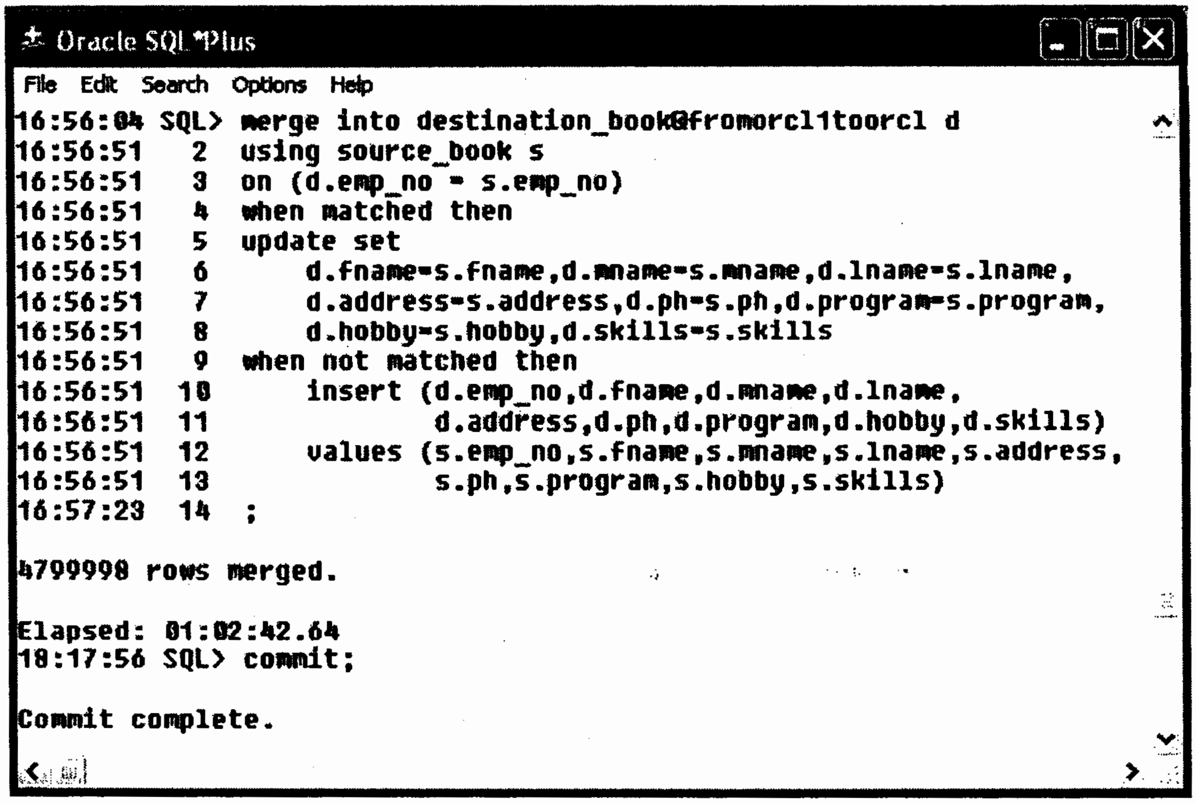




```
16:55:59 SQL> merge into destination_teacher@fromorcl1toorcl d
16:56:17   2  using source_teacher s
16:56:17   3  on (d.emp_no = s.emp_no)
16:56:17   4  when matched then
16:56:17   5  update set
16:56:17   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:56:17   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:56:17   8      d.hobby=s.hobby,d.skills=s.skills
16:56:17   9  when not matched then
16:56:17  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:56:17  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:56:17  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
16:56:18  13              s.ph,s.program,s.hobby,s.skills)
16:57:38  14  ;

4799998 rows merged.

Elapsed: 00:56:44.96
18:22:23 SQL> commit;

Commit complete.
```



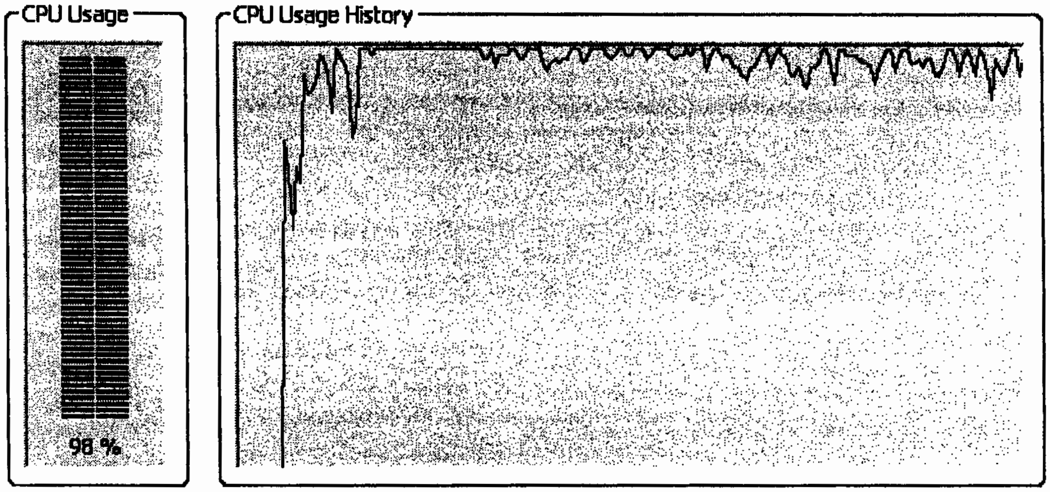
```
16:53:47 SQL> merge into destination_student@fromorcl1toorcl d
16:56:35   2  using source_student s
16:56:35   3  on (d.emp_no = s.emp_no)
16:56:35   4  when matched then
16:56:35   5  update set
16:56:35   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:56:35   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:56:35   8      d.hobby=s.hobby,d.skills=s.skills
16:56:35   9  when not matched then
16:56:35  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:56:35  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:56:35  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
16:56:35  13              s.ph,s.program,s.hobby,s.skills)
16:57:33  14  ;

4799998 rows merged.

Elapsed: 00:53:01.03
17:50:35 SQL>
```

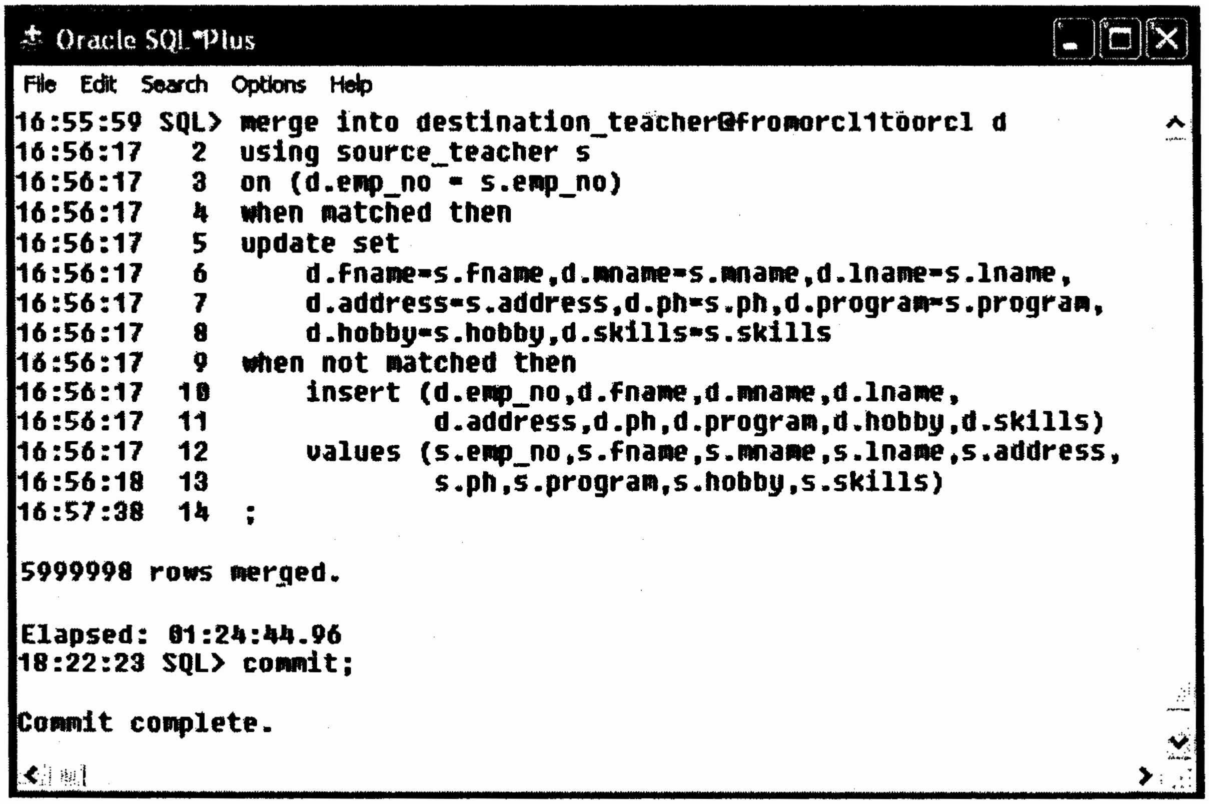


```
16:56:04 SQL> merge into destination_book@fromorcl1toorcl d
16:56:51   2  using source_book s
16:56:51   3  on (d.emp_no = s.emp_no)
16:56:51   4  when matched then
16:56:51   5  update set
16:56:51   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:56:51   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:56:51   8      d.hobby=s.hobby,d.skills=s.skills
16:56:51   9  when not matched then
16:56:51  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:56:51  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:56:51  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
16:56:51  13              s.ph,s.program,s.hobby,s.skills)
16:57:23  14  ;

4799998 rows merged.

Elapsed: 01:02:42.64
18:17:56 SQL> commit;

Commit complete.
```

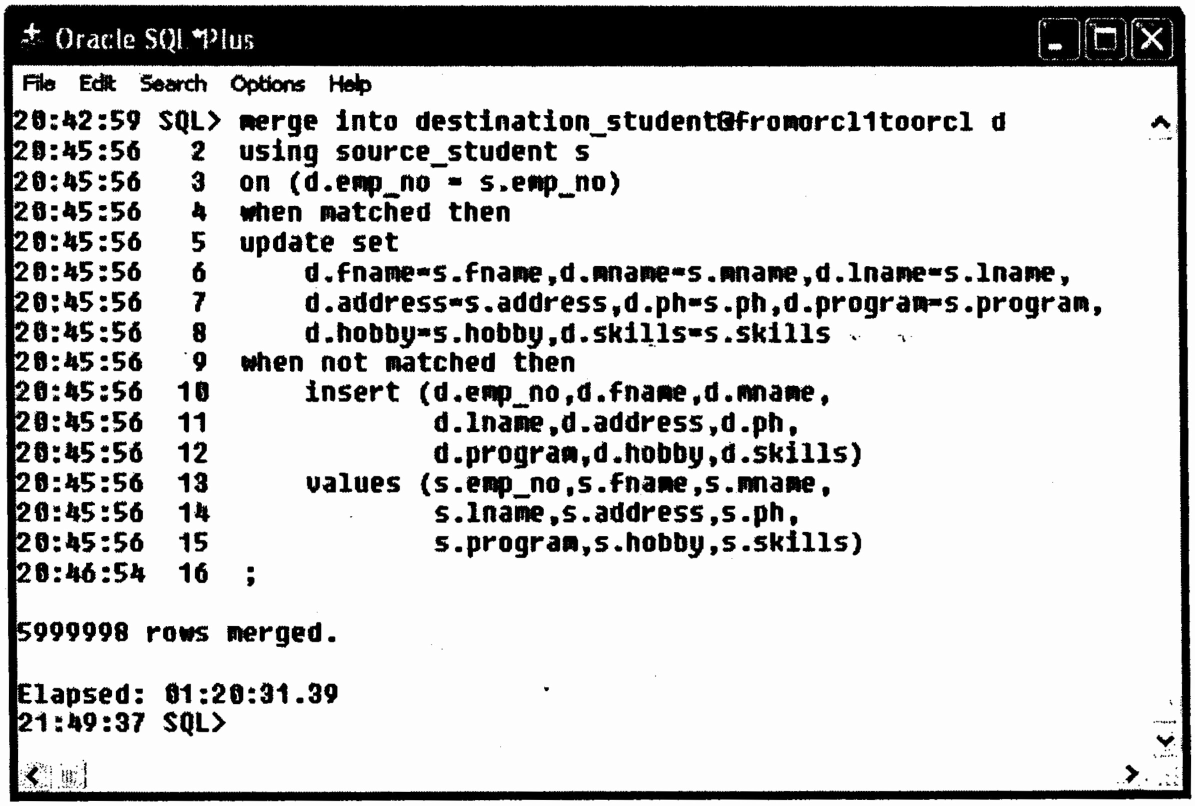
- **Day Five Data:**

The CPU usage shows that the source system is fully busy up to 98% when whole data is being extracted and move to DSA.

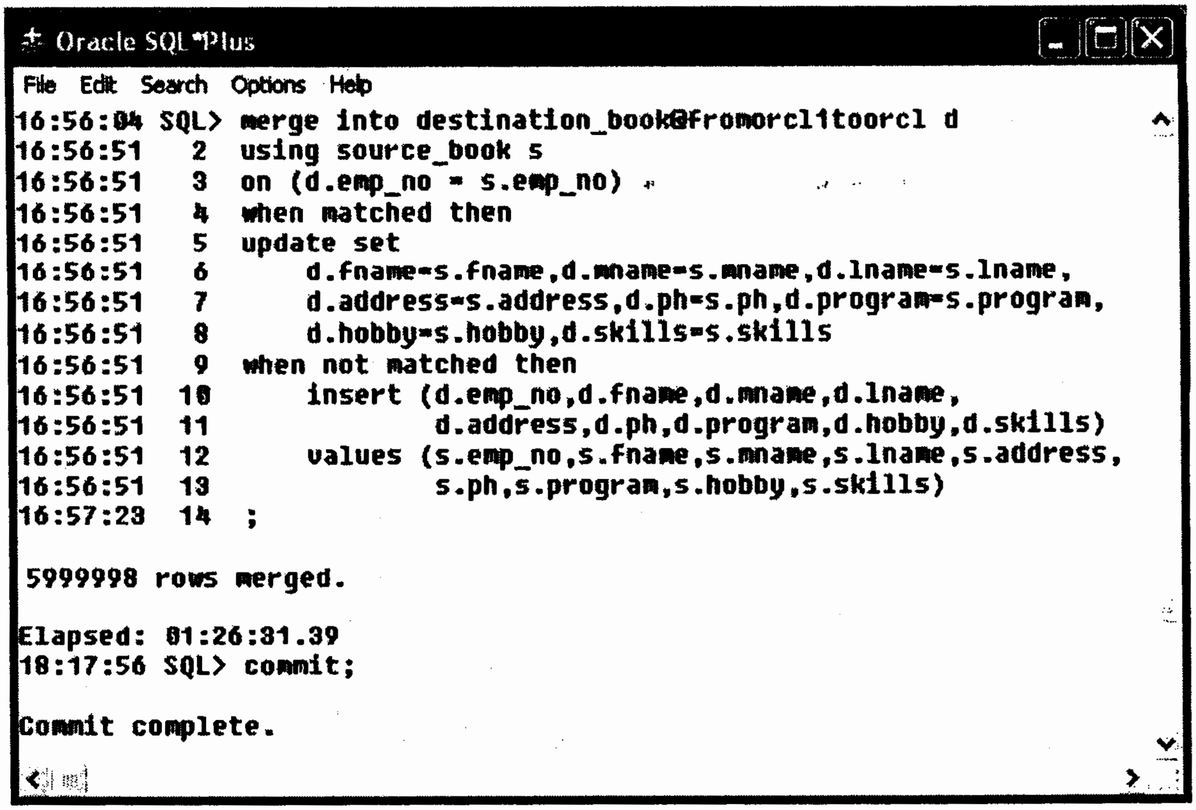




```
16:55:59 SQL> merge into destination_teacher@fromorcl1toorcl d
16:56:17   2  using source_teacher s
16:56:17   3  on (d.emp_no = s.emp_no)
16:56:17   4  when matched then
16:56:17   5  update set
16:56:17   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:56:17   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:56:17   8      d.hobby=s.hobby,d.skills=s.skills
16:56:17   9  when not matched then
16:56:17  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:56:17  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:56:17  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
16:56:18  13              s.ph,s.program,s.hobby,s.skills)
16:57:38  14  ;

5999998 rows merged.

Elapsed: 01:24:44.96
18:22:23 SQL> commit;

Commit complete.
```



```
20:42:59 SQL> merge into destination_student@fromorcl1toorcl d
20:45:56   2  using source_student s
20:45:56   3  on (d.emp_no = s.emp_no)
20:45:56   4  when matched then
20:45:56   5  update set
20:45:56   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
20:45:56   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
20:45:56   8      d.hobby=s.hobby,d.skills=s.skills
20:45:56   9  when not matched then
20:45:56  10      insert (d.emp_no,d.fname,d.mname,
20:45:56  11              d.lname,d.address,d.ph,
20:45:56  12              d.program,d.hobby,d.skills)
20:45:56  13      values (s.emp_no,s.fname,s.mname,
20:45:56  14              s.lname,s.address,s.ph,
20:45:56  15              s.program,s.hobby,s.skills)
20:46:54  16  ;

5999998 rows merged.

Elapsed: 01:20:31.39
21:49:37 SQL>
```

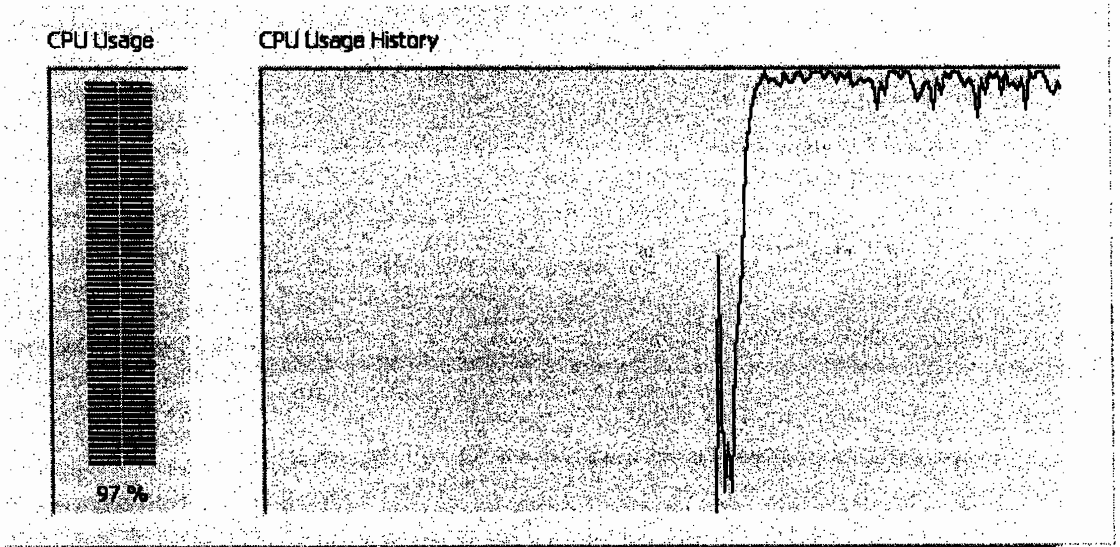


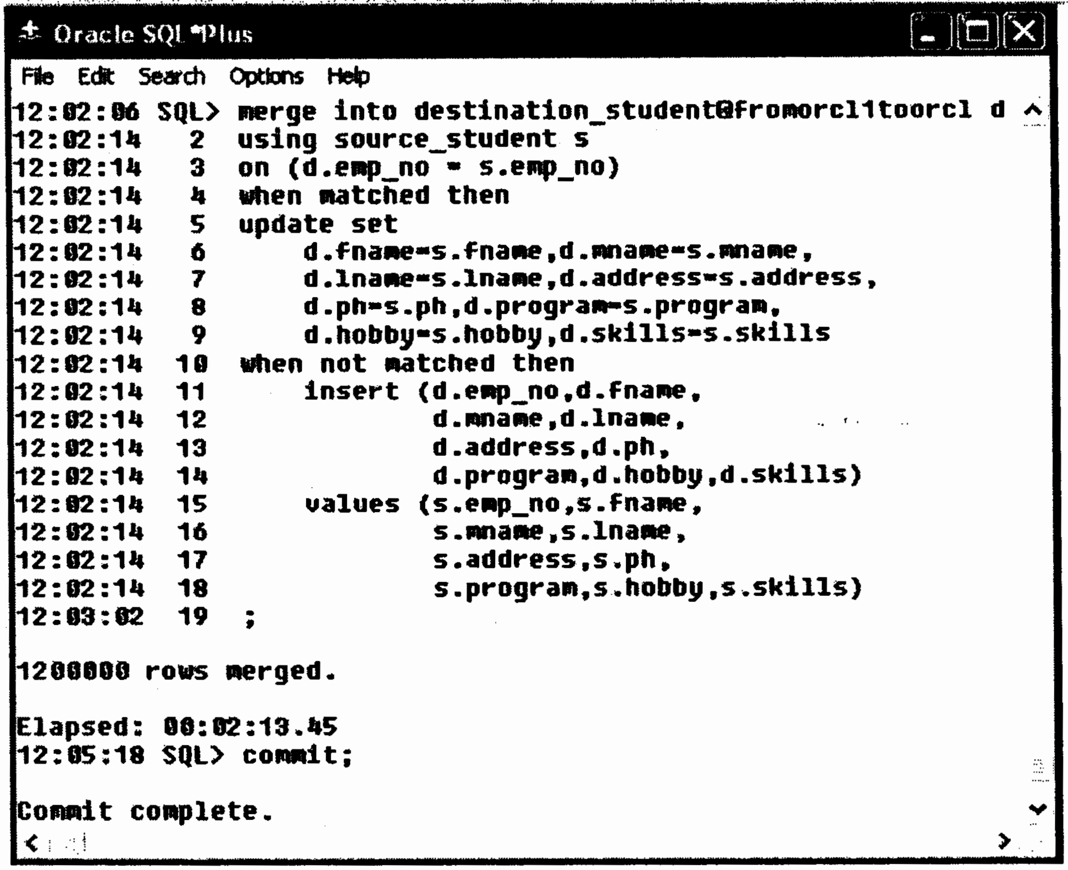
```
16:56:04 SQL> merge into destination_book@fromorcl1toorcl d
16:56:51   2  using source_book s
16:56:51   3  on (d.emp_no = s.emp_no)
16:56:51   4  when matched then
16:56:51   5  update set
16:56:51   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:56:51   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:56:51   8      d.hobby=s.hobby,d.skills=s.skills
16:56:51   9  when not matched then
16:56:51  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:56:51  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:56:51  12      values (s.emp_no,s.fname,s.mname,s.lname,s.address,
16:56:51  13              s.ph,s.program,s.hobby,s.skills)
16:57:23  14  ;

 5999998 rows merged.

Elapsed: 01:26:31.39
18:17:56 SQL> commit;

Commit complete.
```

4.8. Extraction of only Critical Data during Peak Hours

- **Day One Data:**

The CPU usage shows that the source system is fully busy up to 97% when only critical data is being extracted and move to DSA.

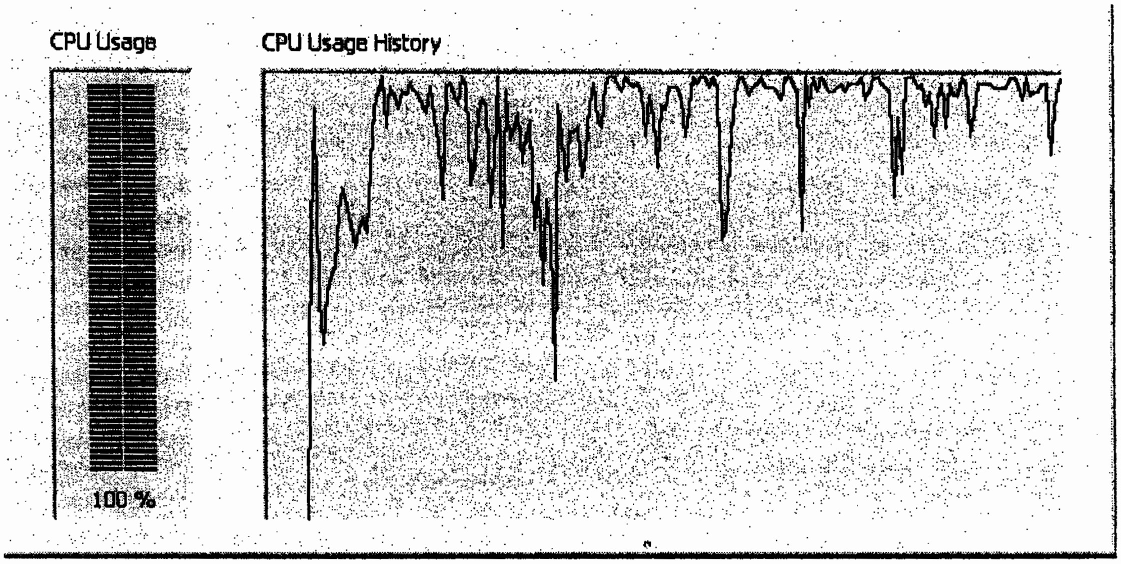


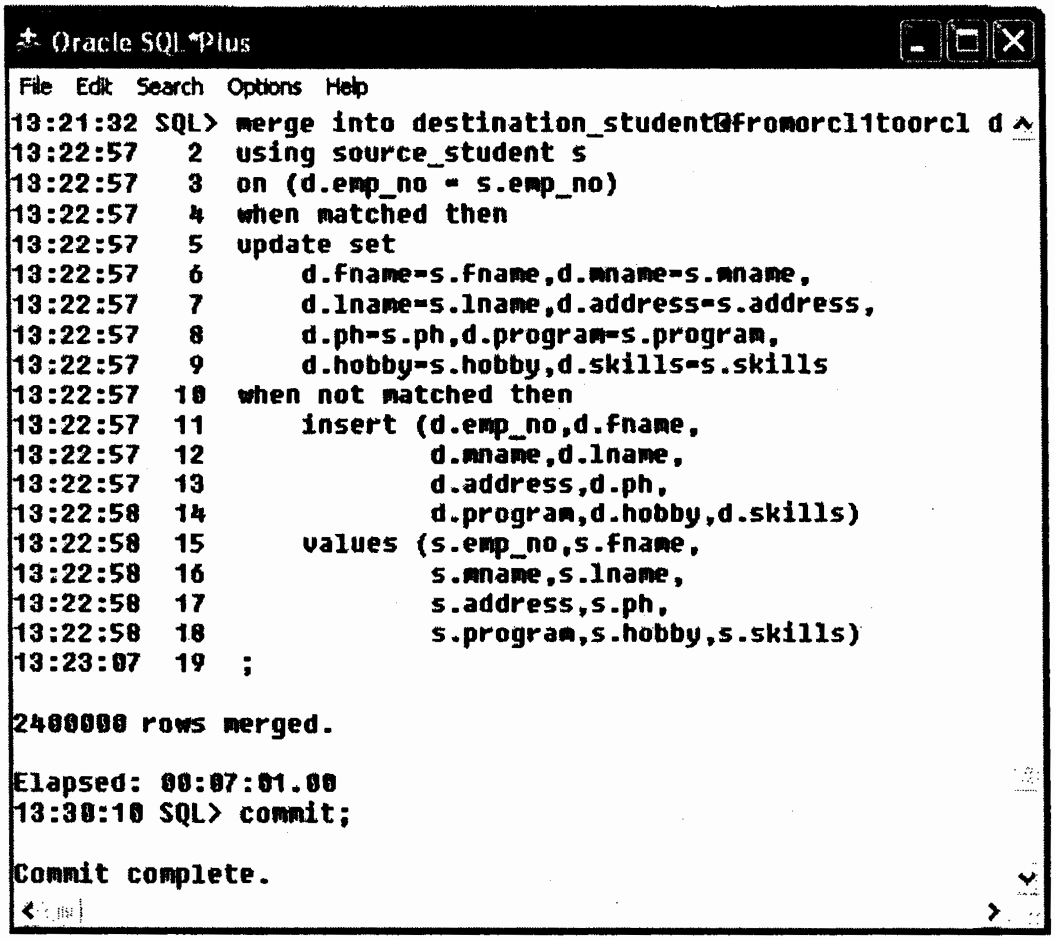


```
12:02:06 SQL> merge into destination_student@fromorcl1toorcl d
12:02:14   2  using source_student s
12:02:14   3  on (d.emp_no = s.emp_no)
12:02:14   4  when matched then
12:02:14   5  update set
12:02:14   6      d.fname=s.fname,d.mname=s.mname,
12:02:14   7      d.lname=s.lname,d.address=s.address,
12:02:14   8      d.ph=s.ph,d.program=s.program,
12:02:14   9      d.hobby=s.hobby,d.skills=s.skills
12:02:14  10  when not matched then
12:02:14  11      insert (d.emp_no,d.fname,
12:02:14  12              d.mname,d.lname,
12:02:14  13              d.address,d.ph,
12:02:14  14              d.program,d.hobby,d.skills)
12:02:14  15      values (s.emp_no,s.fname,
12:02:14  16              s.mname,s.lname,
12:02:14  17              s.address,s.ph,
12:02:14  18              s.program,s.hobby,s.skills)
12:03:02  19  ;

1200000 rows merged.

Elapsed: 00:02:13.45
12:05:18 SQL> commit;

Commit complete.
```

- **Day Two Data:**

The CPU usage shows that the source system is fully busy up to 100% when only critical data is being extracted and move to DSA.

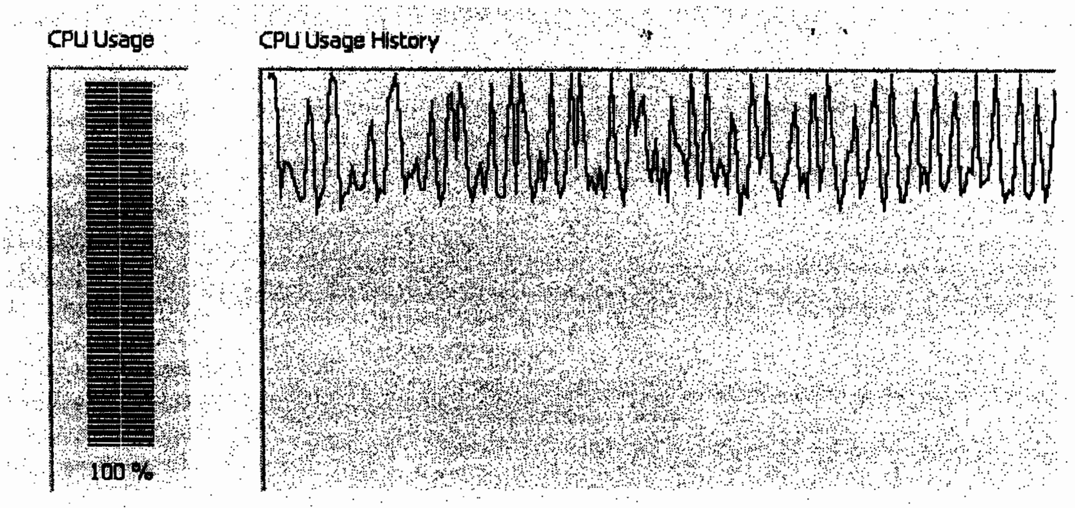


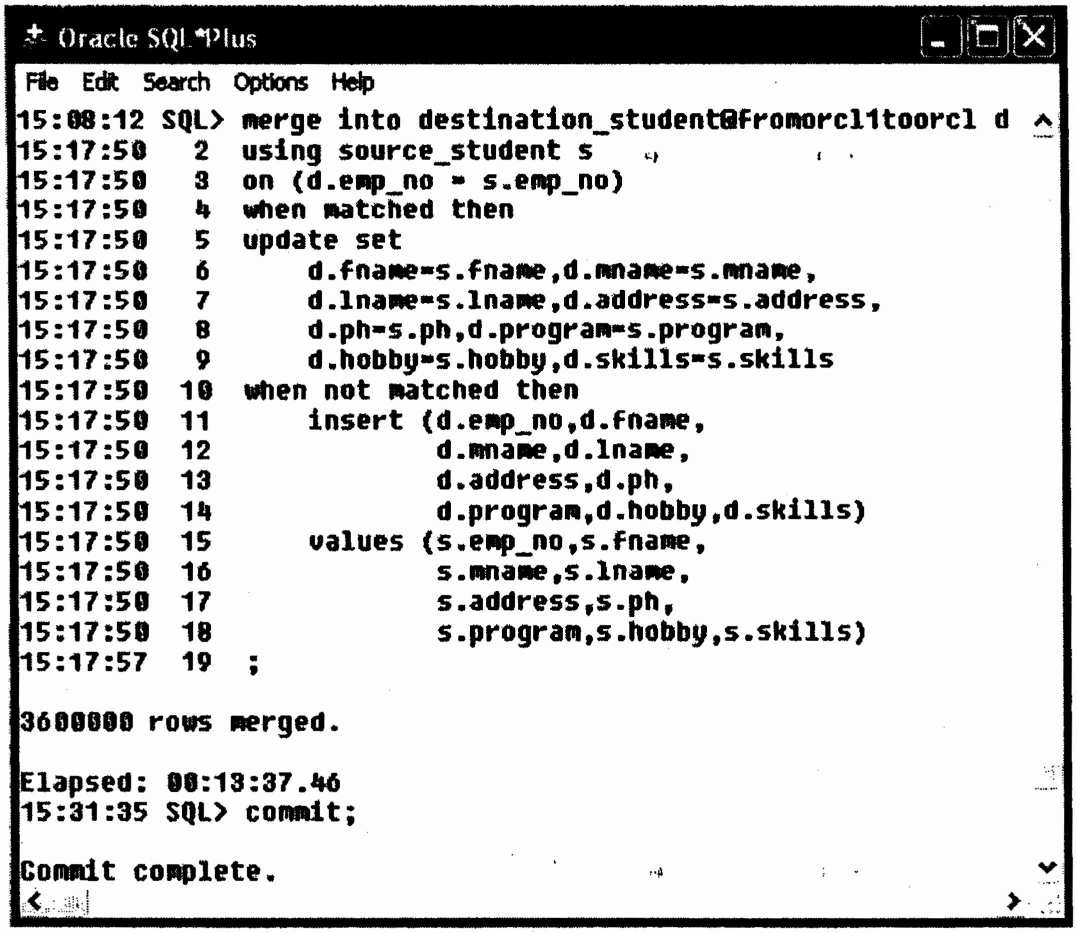


```
13:21:32 SQL> merge into destination_student@fromorcl1toorcl d
13:22:57   2  using source_student s
13:22:57   3  on (d.emp_no = s.emp_no)
13:22:57   4  when matched then
13:22:57   5  update set
13:22:57   6      d.fname=s.fname,d.mname=s.mname,
13:22:57   7      d.lname=s.lname,d.address=s.address,
13:22:57   8      d.ph=s.ph,d.program=s.program,
13:22:57   9      d.hobby=s.hobby,d.skills=s.skills
13:22:57  10  when not matched then
13:22:57  11      insert (d.emp_no,d.fname,
13:22:57  12              d.mname,d.lname,
13:22:57  13              d.address,d.ph,
13:22:58  14              d.program,d.hobby,d.skills)
13:22:58  15      values (s.emp_no,s.fname,
13:22:58  16              s.mname,s.lname,
13:22:58  17              s.address,s.ph,
13:22:58  18              s.program,s.hobby,s.skills)
13:23:07  19  ;

2400000 rows merged.

Elapsed: 00:07:01.00
13:30:10 SQL> commit;

Commit complete.
```

- **Day Three Data:**

The CPU usage shows that the source system is fully busy up to 100% when only critical data is being extracted and move to DSA.

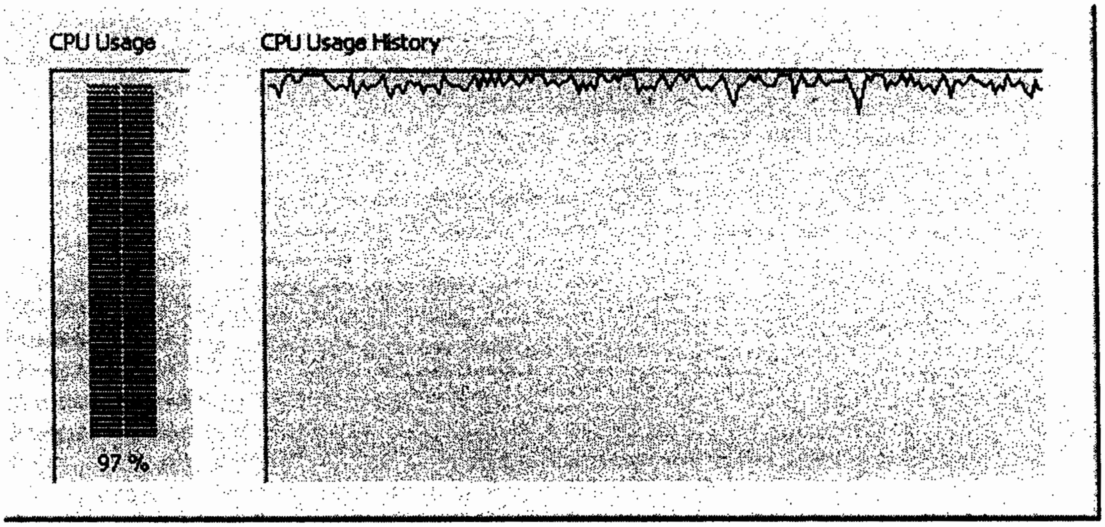


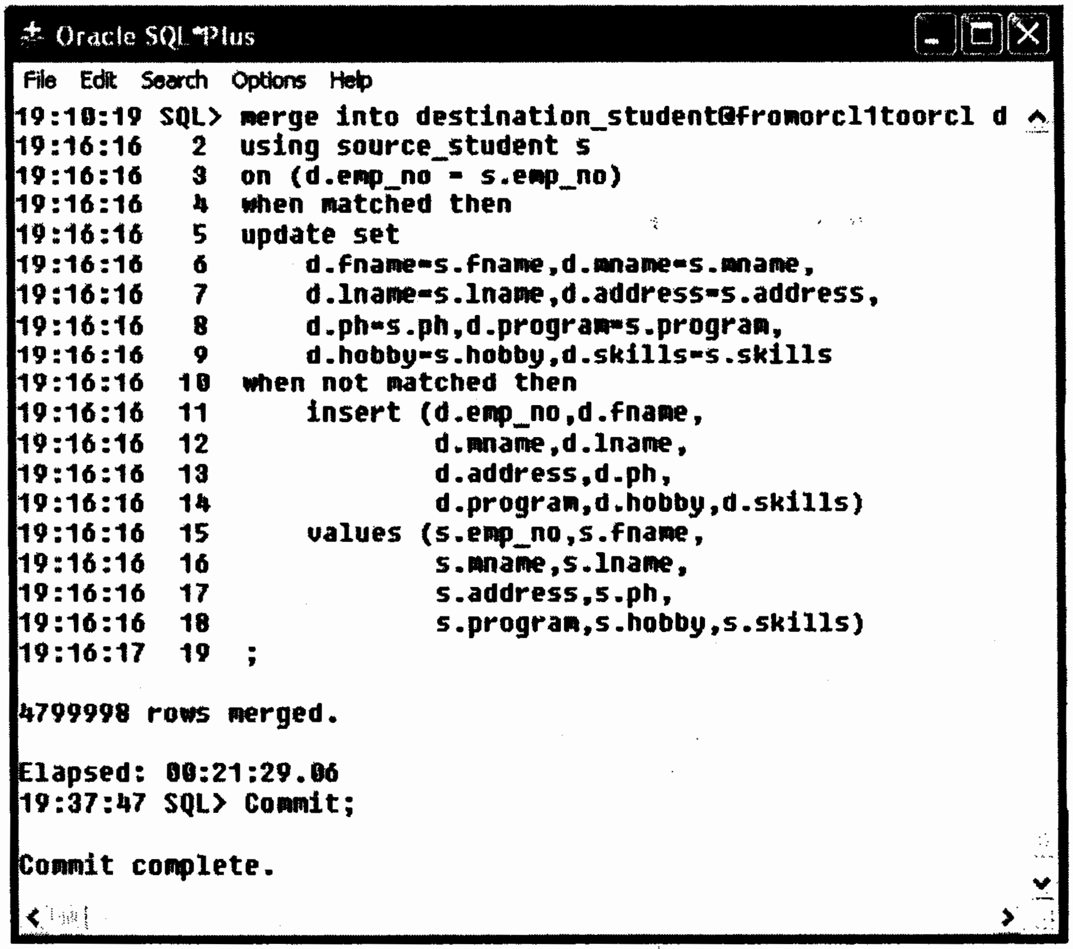


```
15:08:12 SQL> merge into destination_student@fromorcl1toorcl d
15:17:50   2  using source_student s
15:17:50   3  on (d.emp_no = s.emp_no)
15:17:50   4  when matched then
15:17:50   5  update set
15:17:50   6      d.fname=s.fname,d.mname=s.mname,
15:17:50   7      d.lname=s.lname,d.address=s.address,
15:17:50   8      d.ph=s.ph,d.program=s.program,
15:17:50   9      d.hobby=s.hobby,d.skills=s.skills
15:17:50  10  when not matched then
15:17:50  11      insert (d.emp_no,d.fname,
15:17:50  12              d.mname,d.lname,
15:17:50  13              d.address,d.ph,
15:17:50  14              d.program,d.hobby,d.skills)
15:17:50  15      values (s.emp_no,s.fname,
15:17:50  16              s.mname,s.lname,
15:17:50  17              s.address,s.ph,
15:17:50  18              s.program,s.hobby,s.skills)
15:17:57  19  ;

3600000 rows merged.

Elapsed: 00:13:37.46
15:31:35 SQL> commit;

Commit complete.
```

- **Day Four Data:**

The CPU usage shows that the source system is fully busy up to 97% when only critical data is being extracted and move to DSA.

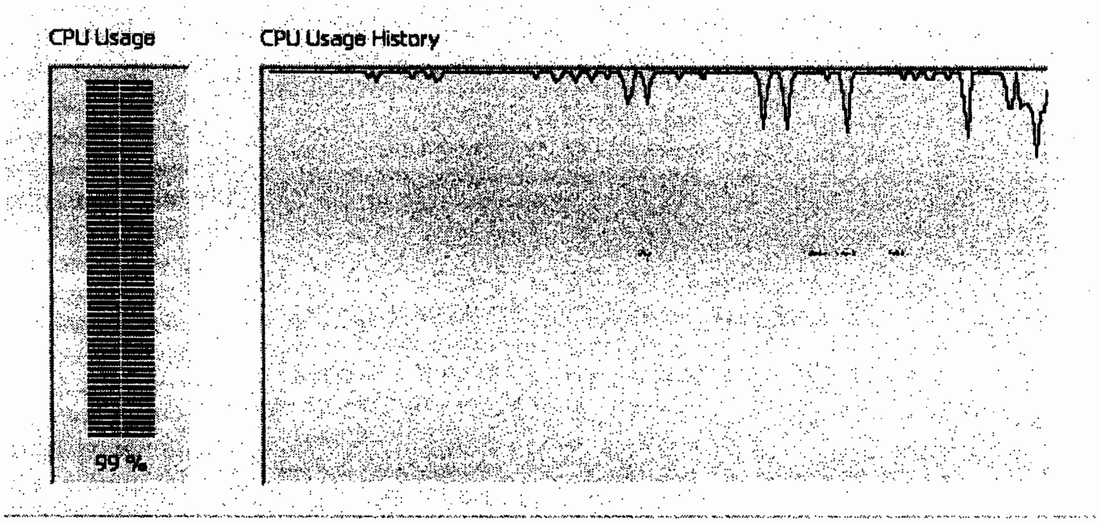


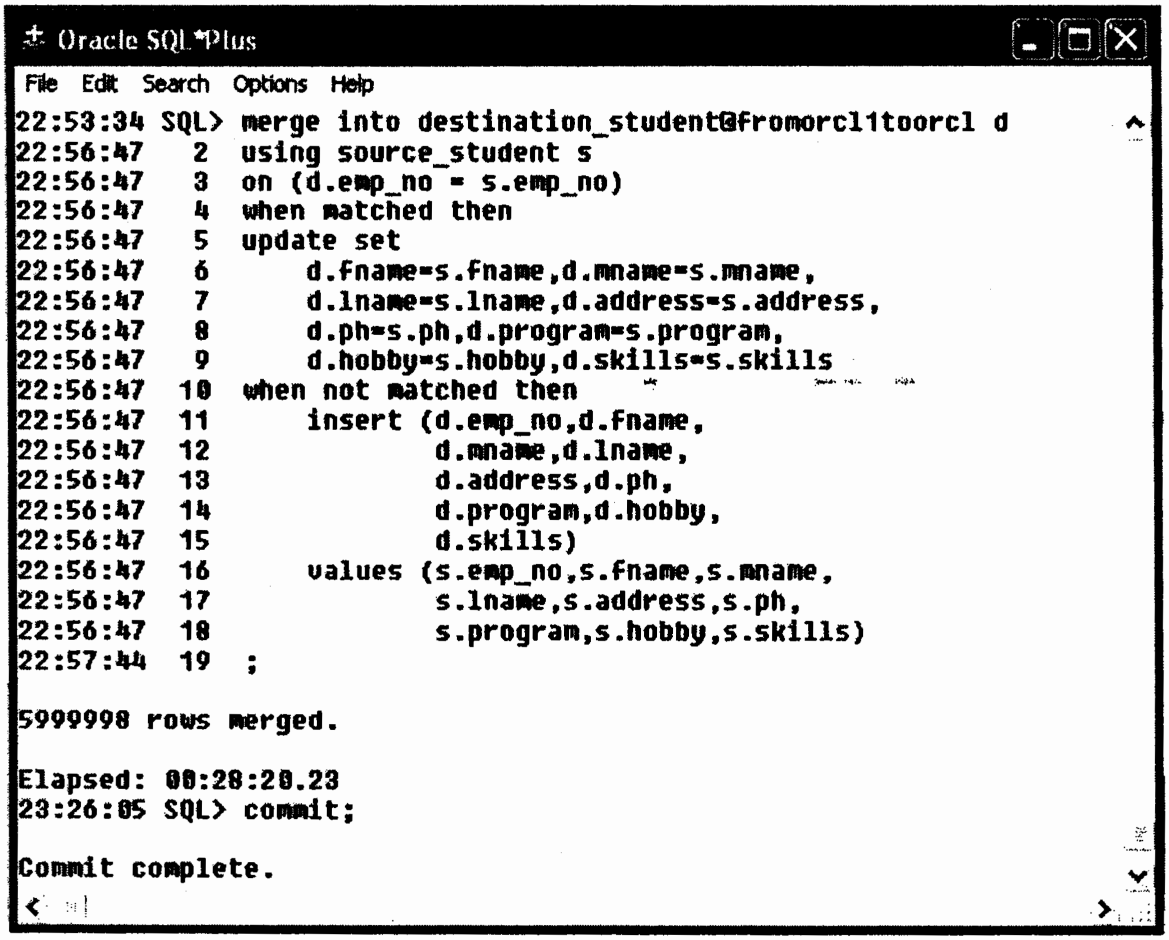


```
19:10:19 SQL> merge into destination_student@fromorcl1toorcl d
19:16:16   2  using source_student s
19:16:16   3  on (d.emp_no = s.emp_no)
19:16:16   4  when matched then
19:16:16   5  update set
19:16:16   6      d.fname=s.fname,d.mname=s.mname,
19:16:16   7      d.lname=s.lname,d.address=s.address,
19:16:16   8      d.ph=s.ph,d.program=s.program,
19:16:16   9      d.hobby=s.hobby,d.skills=s.skills
19:16:16  10  when not matched then
19:16:16  11      insert (d.emp_no,d.fname,
19:16:16  12              d.mname,d.lname,
19:16:16  13              d.address,d.ph,
19:16:16  14              d.program,d.hobby,d.skills)
19:16:16  15      values (s.emp_no,s.fname,
19:16:16  16              s.mname,s.lname,
19:16:16  17              s.address,s.ph,
19:16:16  18              s.program,s.hobby,s.skills)
19:16:17  19  ;

4799998 rows merged.

Elapsed: 00:21:29.06
19:37:47 SQL> Commit;

Commit complete.
```

- **Day Five Data:**

The CPU usage shows that the source system is fully busy up to 99% when only critical data is being extracted and move to DSA.



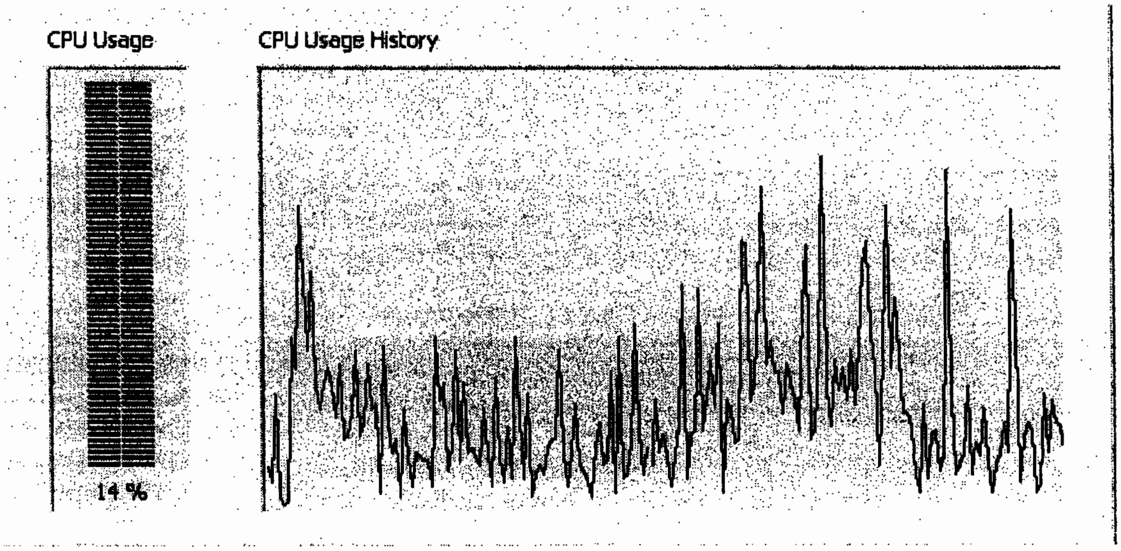


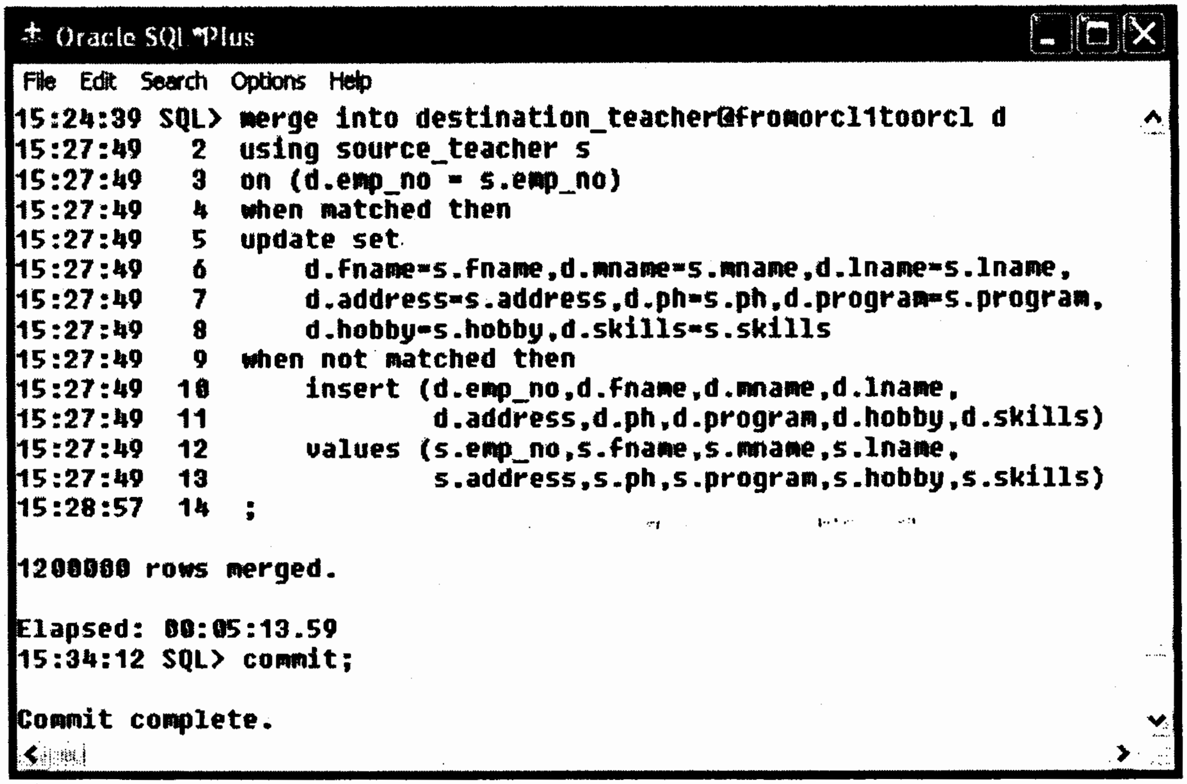
```
22:53:34 SQL> merge into destination_student@fromorcl1toorcl d
22:56:47   2  using source_student s
22:56:47   3  on (d.emp_no = s.emp_no)
22:56:47   4  when matched then
22:56:47   5  update set
22:56:47   6      d.fname=s.fname,d.mname=s.mname,
22:56:47   7      d.lname=s.lname,d.address=s.address,
22:56:47   8      d.ph=s.ph,d.program=s.program,
22:56:47   9      d.hobby=s.hobby,d.skills=s.skills
22:56:47  10  when not matched then
22:56:47  11      insert (d.emp_no,d.fname,
22:56:47  12              d.mname,d.lname,
22:56:47  13              d.address,d.ph,
22:56:47  14              d.program,d.hobby,
22:56:47  15              d.skills)
22:56:47  16      values (s.emp_no,s.fname,s.mname,
22:56:47  17              s.lname,s.address,s.ph,
22:56:47  18              s.program,s.hobby,s.skills)
22:57:44  19  ;

5999998 rows merged.

Elapsed: 00:28:20.23
23:26:05 SQL> commit;

Commit complete.
```

4.9. Extraction of whole Data during off Peak Hours

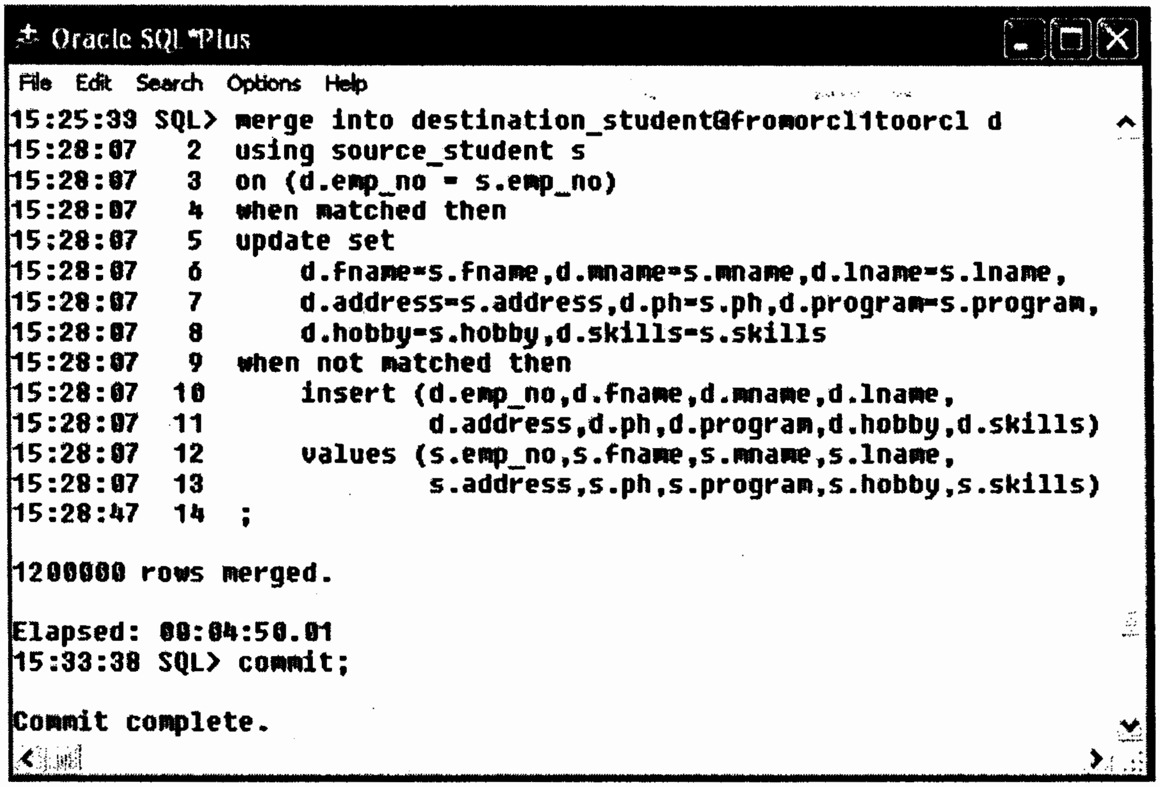
- **Day One Data:**

The CPU usage shows that the source system is busy up to 14% when whole data is being extracted and move to DSA.

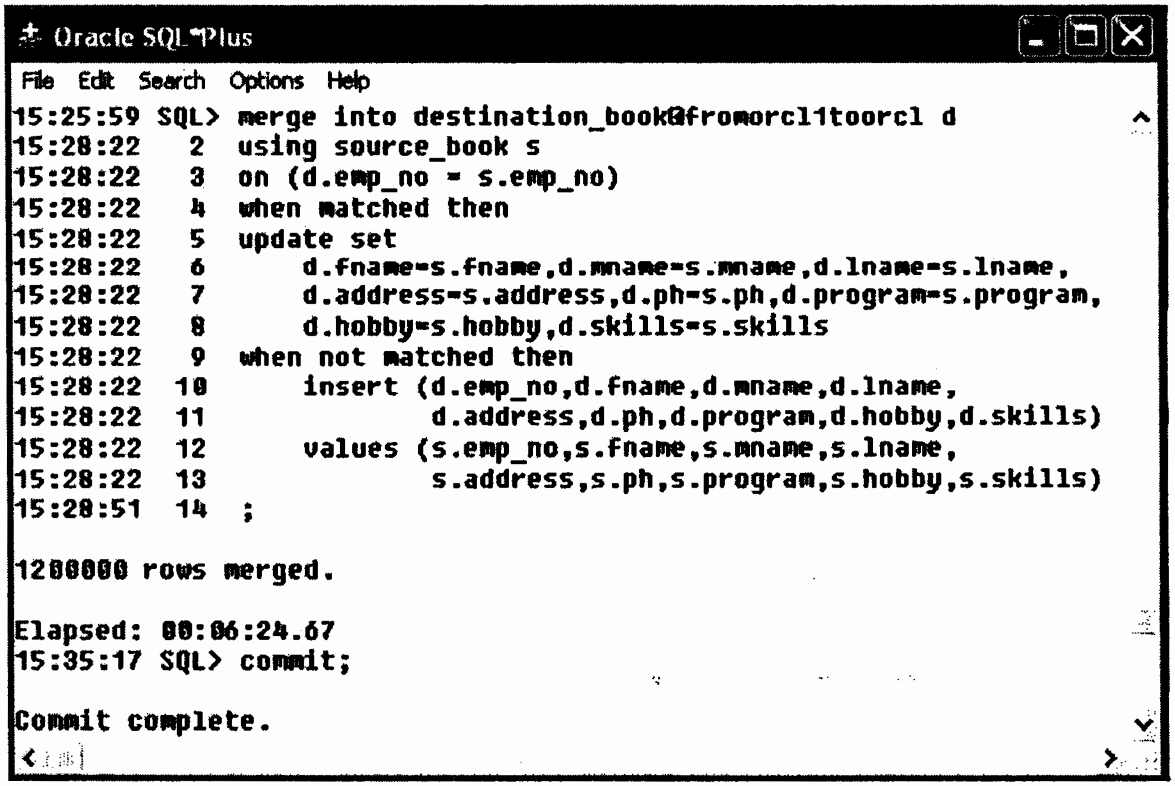




```
15:24:39 SQL> merge into destination_teacher@fromorcl1toorcl d
15:27:49   2  using source_teacher s
15:27:49   3  on (d.emp_no = s.emp_no)
15:27:49   4  when matched then
15:27:49   5  update set
15:27:49   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
15:27:49   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
15:27:49   8      d.hobby=s.hobby,d.skills=s.skills
15:27:49   9  when not matched then
15:27:49  10      insert (d.emp_no,d.fname,d.mname,d.lname,
15:27:49  11              d.address,d.ph,d.program,d.hobby,d.skills)
15:27:49  12      values (s.emp_no,s.fname,s.mname,s.lname,
15:27:49  13              s.address,s.ph,s.program,s.hobby,s.skills)
15:28:57  14  ;

1200000 rows merged.

Elapsed: 00:05:13.59
15:34:12 SQL> commit;

Commit complete.
```



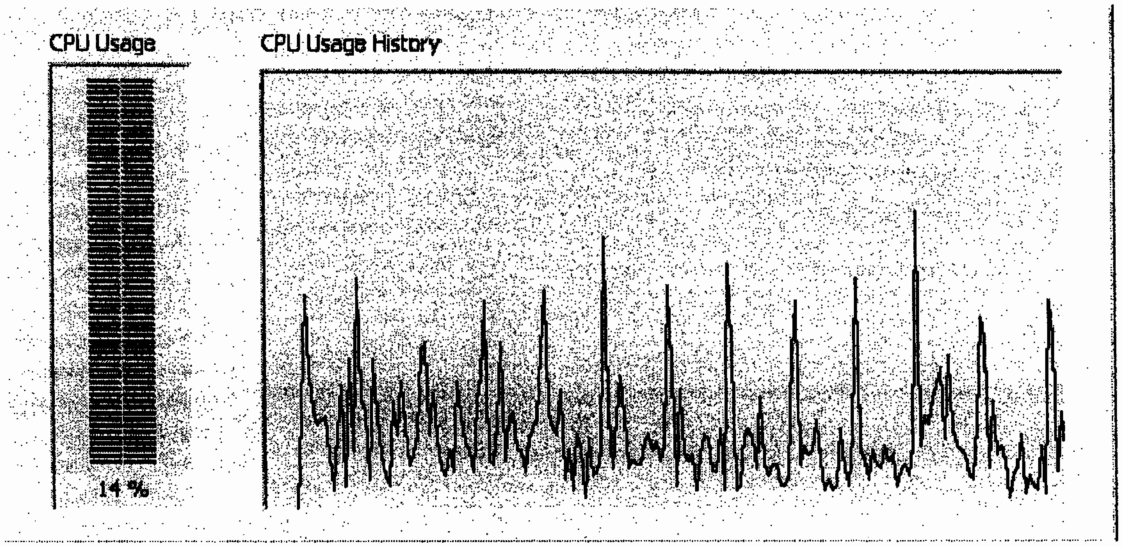
```
15:25:39 SQL> merge into destination_student@fromorcl1toorcl d
15:28:07   2  using source_student s
15:28:07   3  on (d.emp_no = s.emp_no)
15:28:07   4  when matched then
15:28:07   5  update set
15:28:07   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
15:28:07   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
15:28:07   8      d.hobby=s.hobby,d.skills=s.skills
15:28:07   9  when not matched then
15:28:07  10      insert (d.emp_no,d.fname,d.mname,d.lname,
15:28:07  11              d.address,d.ph,d.program,d.hobby,d.skills)
15:28:07  12      values (s.emp_no,s.fname,s.mname,s.lname,
15:28:07  13              s.address,s.ph,s.program,s.hobby,s.skills)
15:28:47  14  ;

1200000 rows merged.

Elapsed: 00:04:50.01
15:33:38 SQL> commit;

Commit complete.
```

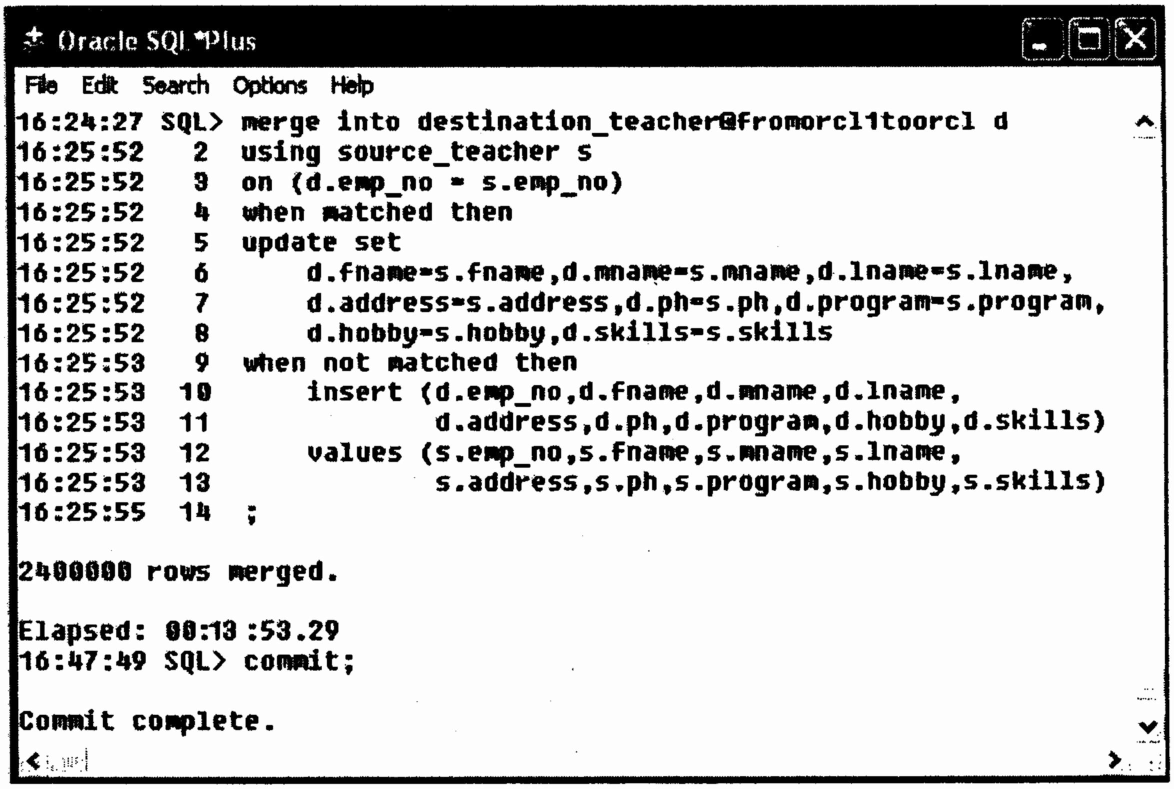


```
15:25:59 SQL> merge into destination_book@fromorcl1toorcl d
15:28:22   2  using source_book s
15:28:22   3  on (d.emp_no = s.emp_no)
15:28:22   4  when matched then
15:28:22   5  update set
15:28:22   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
15:28:22   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
15:28:22   8      d.hobby=s.hobby,d.skills=s.skills
15:28:22   9  when not matched then
15:28:22  10      insert (d.emp_no,d.fname,d.mname,d.lname,
15:28:22  11              d.address,d.ph,d.program,d.hobby,d.skills)
15:28:22  12      values (s.emp_no,s.fname,s.mname,s.lname,
15:28:22  13              s.address,s.ph,s.program,s.hobby,s.skills)
15:28:51  14  ;

1200000 rows merged.

Elapsed: 00:06:24.67
15:35:17 SQL> commit;

Commit complete.
```

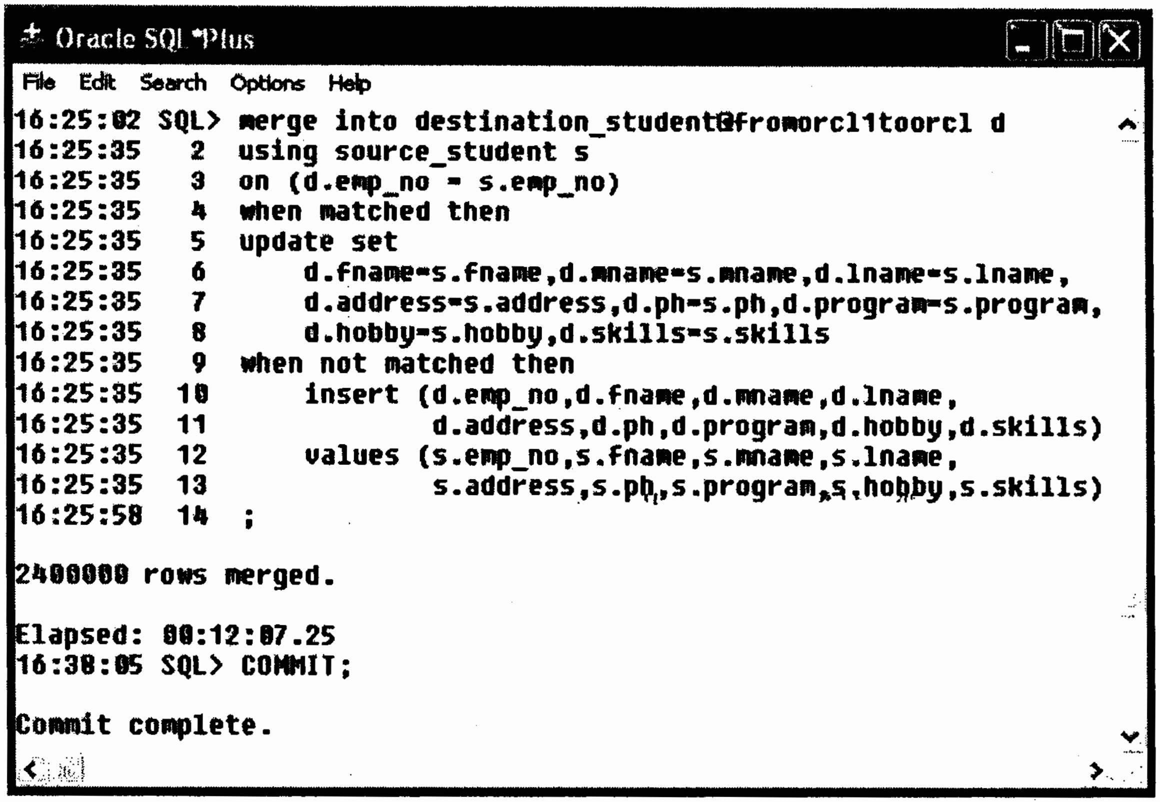
- **Day Two Data:**

The CPU usage shows that the source system is busy up to 14% when whole data is being extracted and move to DSA.

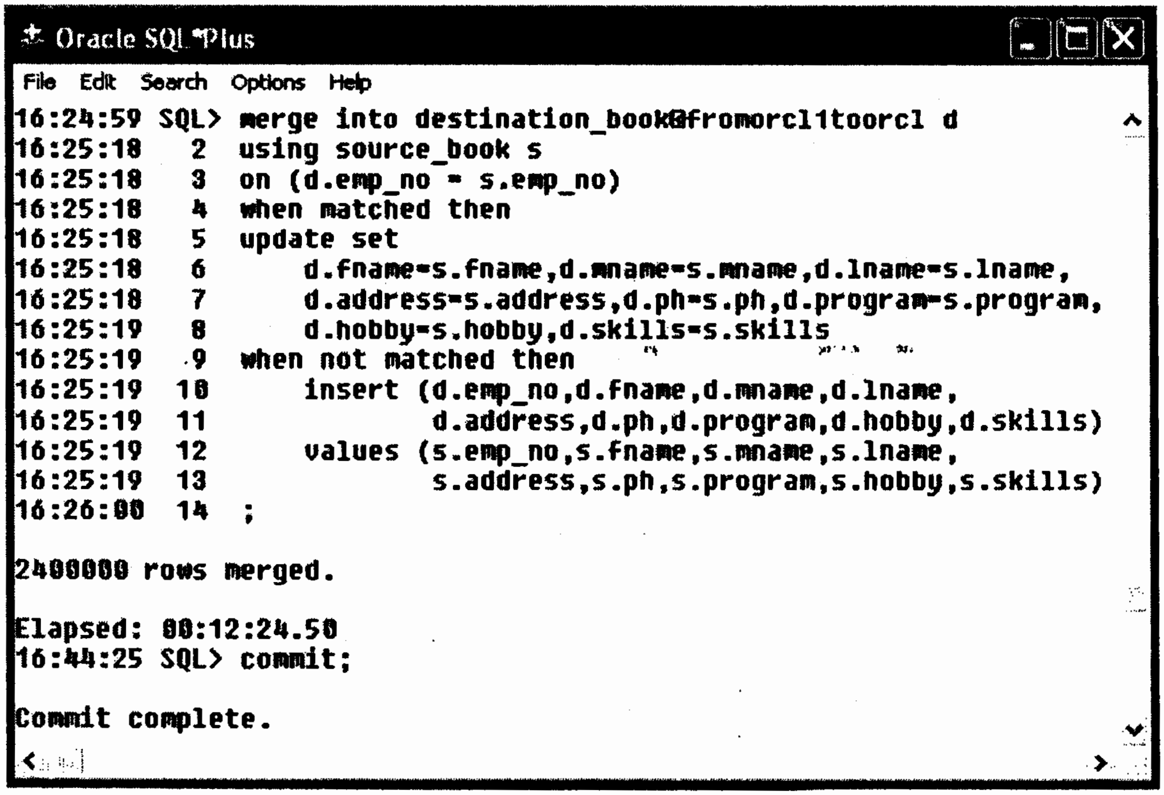




```
16:24:27 SQL> merge into destination_teacher@fromorcl1toorcl d
16:25:52   2  using source_teacher s
16:25:52   3  on (d.emp_no = s.emp_no)
16:25:52   4  when matched then
16:25:52   5  update set
16:25:52   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:25:52   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:25:52   8      d.hobby=s.hobby,d.skills=s.skills
16:25:53   9  when not matched then
16:25:53  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:25:53  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:25:53  12      values (s.emp_no,s.fname,s.mname,s.lname,
16:25:53  13              s.address,s.ph,s.program,s.hobby,s.skills)
16:25:55  14  ;

2400000 rows merged.

Elapsed: 00:13:53.29
16:47:49 SQL> commit;

Commit complete.
```



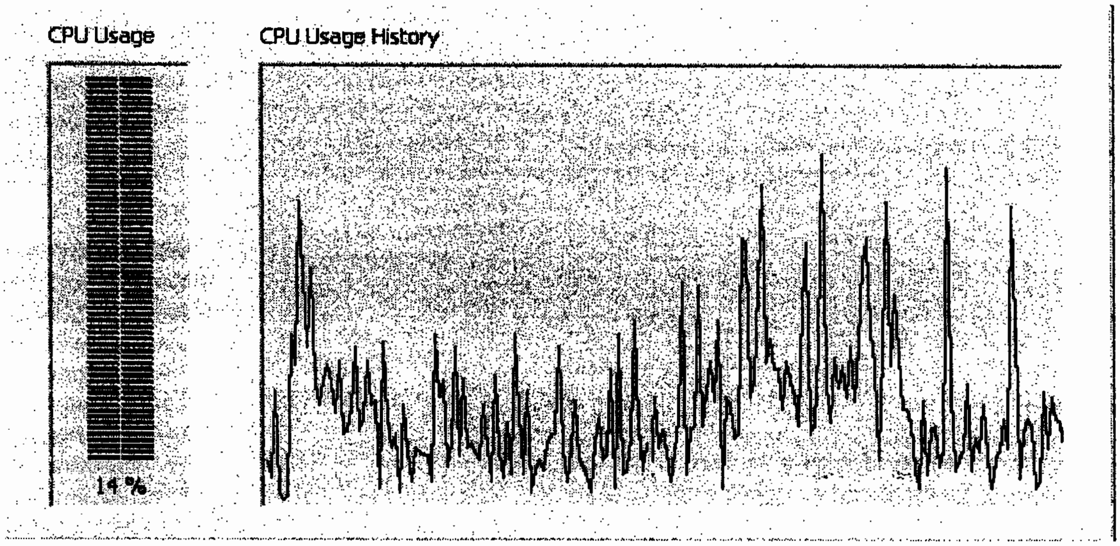
```
16:25:02 SQL> merge into destination_student@fromorcl1toorcl d
16:25:35   2  using source_student s
16:25:35   3  on (d.emp_no = s.emp_no)
16:25:35   4  when matched then
16:25:35   5  update set
16:25:35   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:25:35   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:25:35   8      d.hobby=s.hobby,d.skills=s.skills
16:25:35   9  when not matched then
16:25:35  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:25:35  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:25:35  12      values (s.emp_no,s.fname,s.mname,s.lname,
16:25:35  13              s.address,s.ph,s.program,s.hobby,s.skills)
16:25:58  14  ;

2400000 rows merged.

Elapsed: 00:12:07.25
16:38:05 SQL> COMMIT;

Commit complete.
```

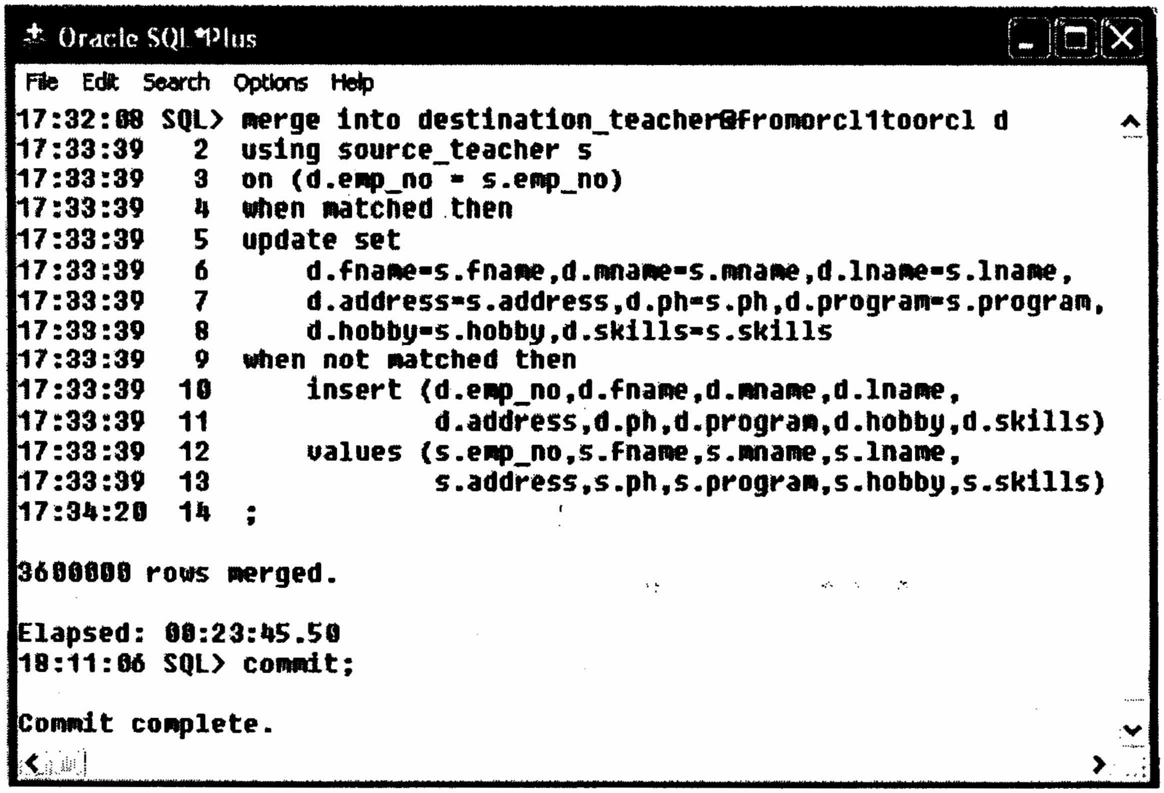


```
16:24:59 SQL> merge into destination_book@fromorcl1toorcl d
16:25:18   2  using source_book s
16:25:18   3  on (d.emp_no = s.emp_no)
16:25:18   4  when matched then
16:25:18   5  update set
16:25:18   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
16:25:18   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
16:25:19   8      d.hobby=s.hobby,d.skills=s.skills
16:25:19   9  when not matched then
16:25:19  10      insert (d.emp_no,d.fname,d.mname,d.lname,
16:25:19  11              d.address,d.ph,d.program,d.hobby,d.skills)
16:25:19  12      values (s.emp_no,s.fname,s.mname,s.lname,
16:25:19  13              s.address,s.ph,s.program,s.hobby,s.skills)
16:26:00  14  ;

2400000 rows merged.

Elapsed: 00:12:24.50
16:44:25 SQL> commit;

Commit complete.
```

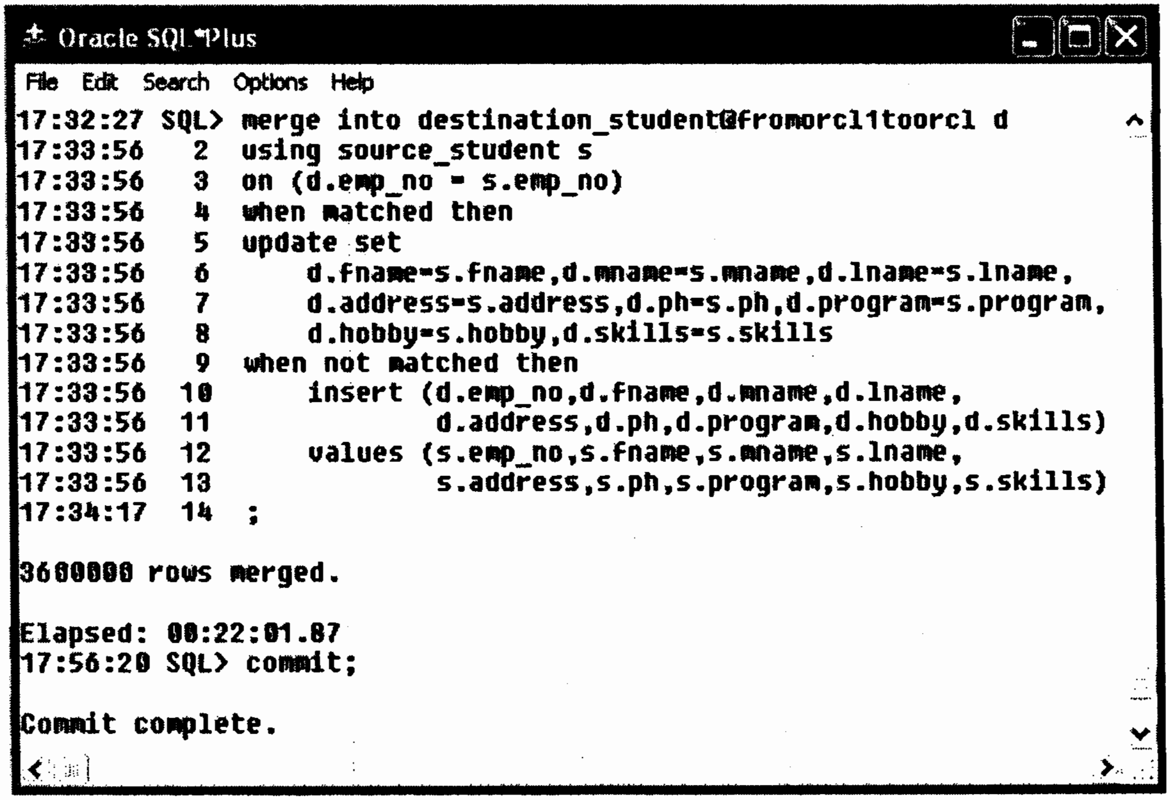
- **Day Three Data:**

The CPU usage shows that the source system is busy up to 14% when whole data is being extracted and move to DSA.

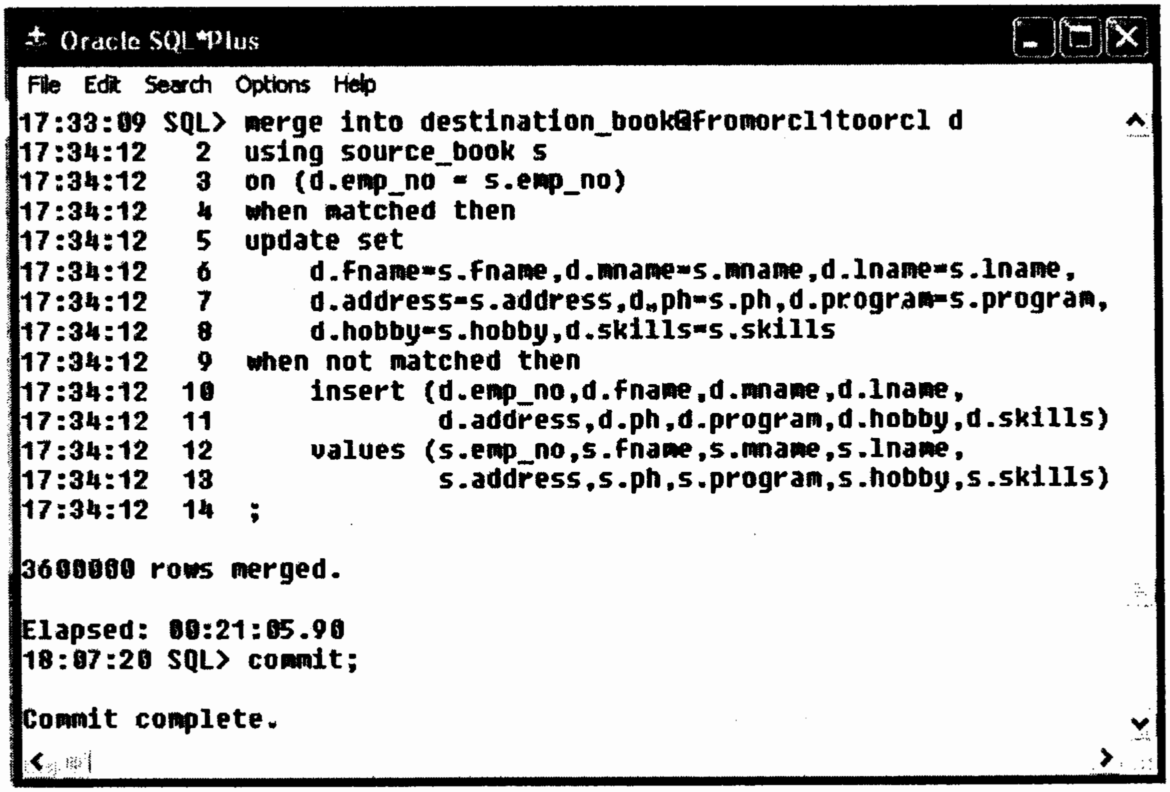




```
17:32:08 SQL> merge into destination_teacher@fromorcl1toorcl d
17:33:39   2  using source_teacher s
17:33:39   3  on (d.emp_no = s.emp_no)
17:33:39   4  when matched then
17:33:39   5  update set
17:33:39   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
17:33:39   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
17:33:39   8      d.hobby=s.hobby,d.skills=s.skills
17:33:39   9  when not matched then
17:33:39  10      insert (d.emp_no,d.fname,d.mname,d.lname,
17:33:39  11              d.address,d.ph,d.program,d.hobby,d.skills)
17:33:39  12      values (s.emp_no,s.fname,s.mname,s.lname,
17:33:39  13              s.address,s.ph,s.program,s.hobby,s.skills)
17:34:20  14  ;

3600000 rows merged.

Elapsed: 00:23:45.50
18:11:06 SQL> commit;

Commit complete.
```



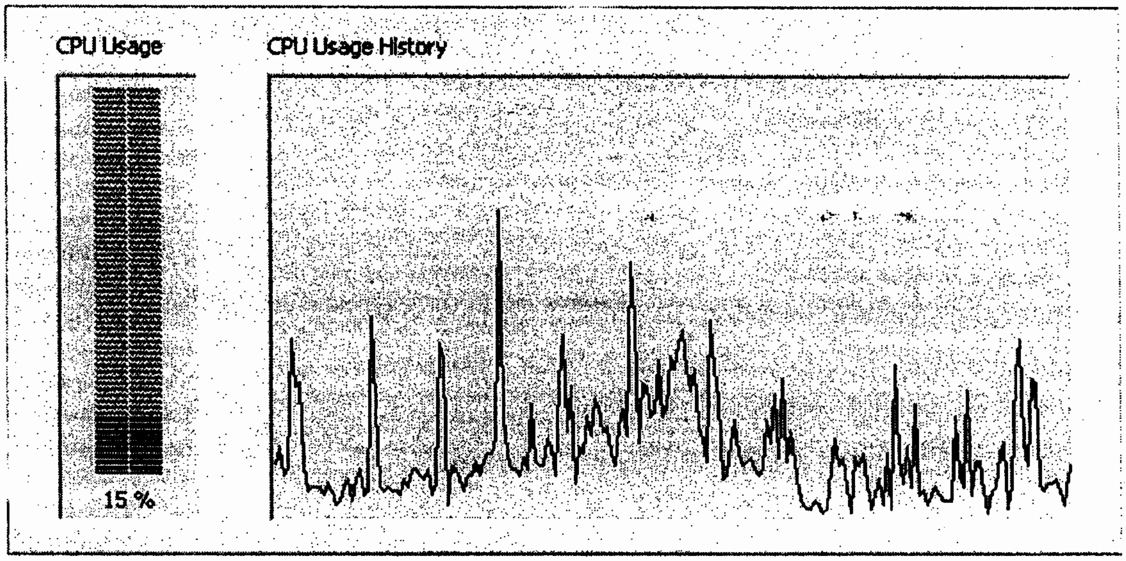
```
17:32:27 SQL> merge into destination_student@fromorcl1toorcl d
17:33:56   2  using source_student s
17:33:56   3  on (d.emp_no = s.emp_no)
17:33:56   4  when matched then
17:33:56   5  update set
17:33:56   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
17:33:56   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
17:33:56   8      d.hobby=s.hobby,d.skills=s.skills
17:33:56   9  when not matched then
17:33:56  10      insert (d.emp_no,d.fname,d.mname,d.lname,
17:33:56  11              d.address,d.ph,d.program,d.hobby,d.skills)
17:33:56  12      values (s.emp_no,s.fname,s.mname,s.lname,
17:33:56  13              s.address,s.ph,s.program,s.hobby,s.skills)
17:34:17  14  ;

3600000 rows merged.

Elapsed: 00:22:01.87
17:56:20 SQL> commit;

Commit complete.
```

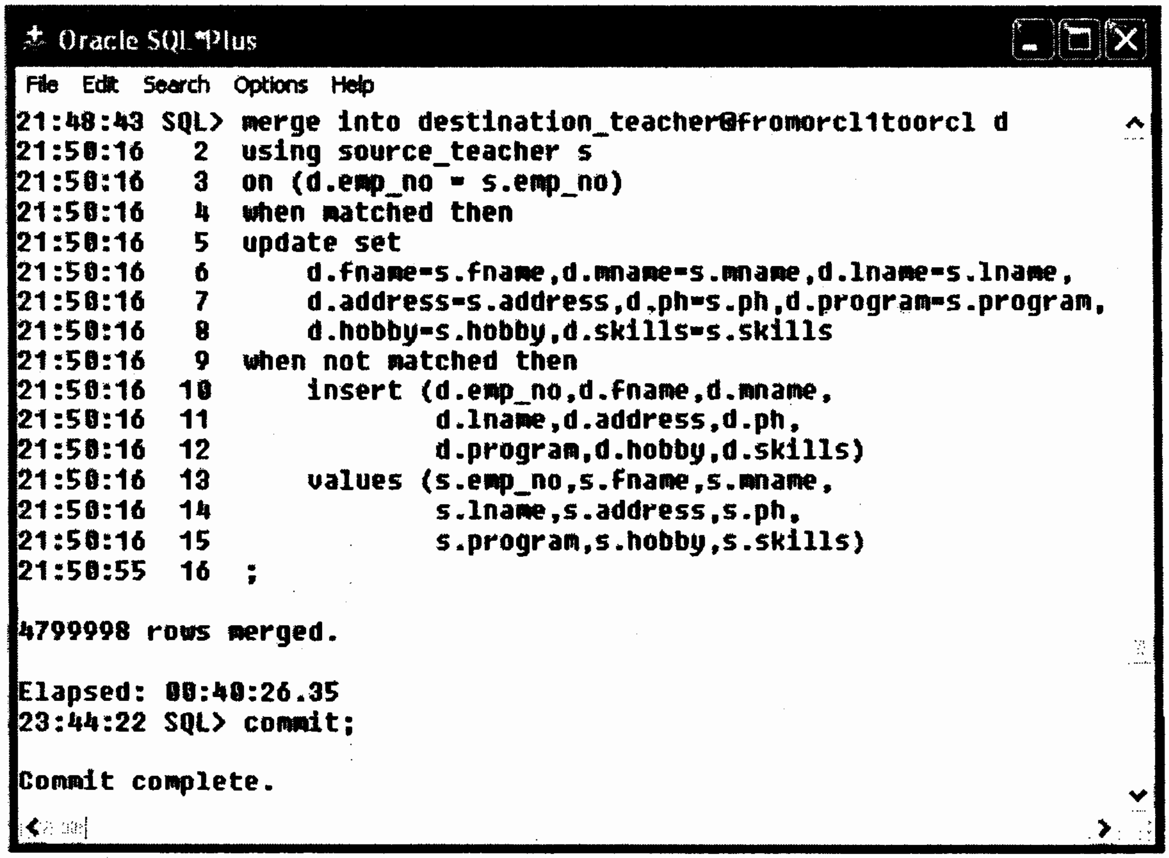


```
17:33:09 SQL> merge into destination_book@fromorcl1toorcl d
17:34:12   2  using source_book s
17:34:12   3  on (d.emp_no = s.emp_no)
17:34:12   4  when matched then
17:34:12   5  update set
17:34:12   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
17:34:12   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
17:34:12   8      d.hobby=s.hobby,d.skills=s.skills
17:34:12   9  when not matched then
17:34:12  10      insert (d.emp_no,d.fname,d.mname,d.lname,
17:34:12  11              d.address,d.ph,d.program,d.hobby,d.skills)
17:34:12  12      values (s.emp_no,s.fname,s.mname,s.lname,
17:34:12  13              s.address,s.ph,s.program,s.hobby,s.skills)
17:34:12  14  ;

3600000 rows merged.

Elapsed: 00:21:05.90
18:07:20 SQL> commit;

Commit complete.
```

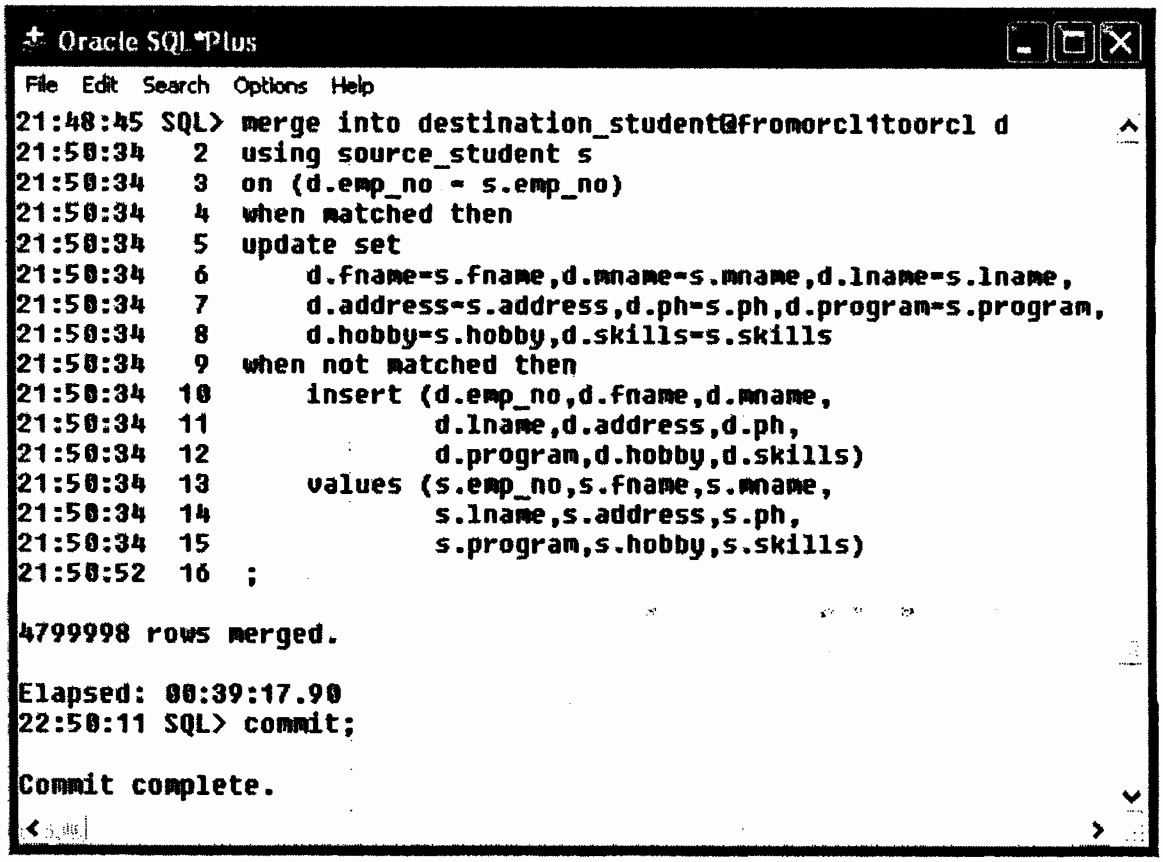
- **Day Four Data:**

The CPU usage shows that the source system is busy up to 15% when whole data is being extracted and move to DSA.

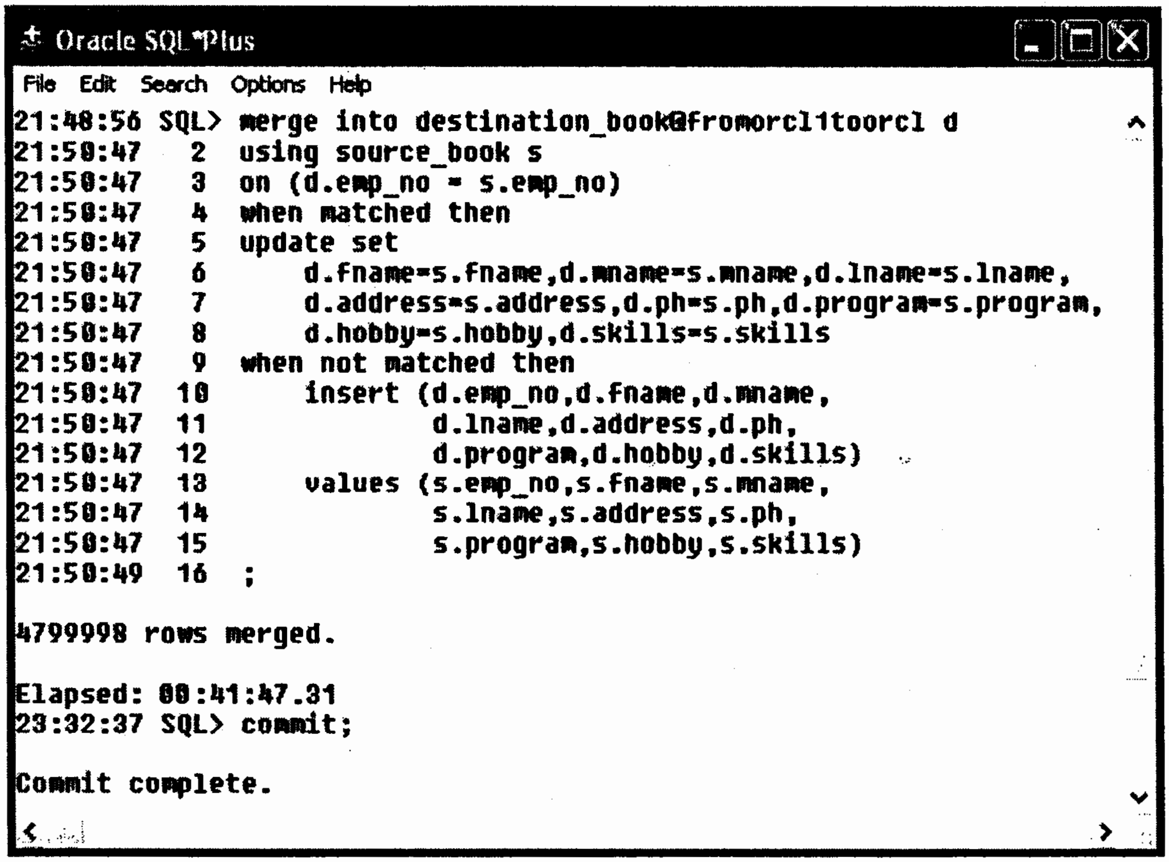




```
21:48:43 SQL> merge into destination_teacher@fromorcl1toorcl d
21:50:16   2  using source_teacher s
21:50:16   3  on (d.emp_no = s.emp_no)
21:50:16   4  when matched then
21:50:16   5  update set
21:50:16   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
21:50:16   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
21:50:16   8      d.hobby=s.hobby,d.skills=s.skills
21:50:16   9  when not matched then
21:50:16  10      insert (d.emp_no,d.fname,d.mname,
21:50:16  11              d.lname,d.address,d.ph,
21:50:16  12              d.program,d.hobby,d.skills)
21:50:16  13      values (s.emp_no,s.fname,s.mname,
21:50:16  14              s.lname,s.address,s.ph,
21:50:16  15              s.program,s.hobby,s.skills)
21:50:55  16  ;

4799998 rows merged.

Elapsed: 00:40:26.35
23:44:22 SQL> commit;

Commit complete.
```



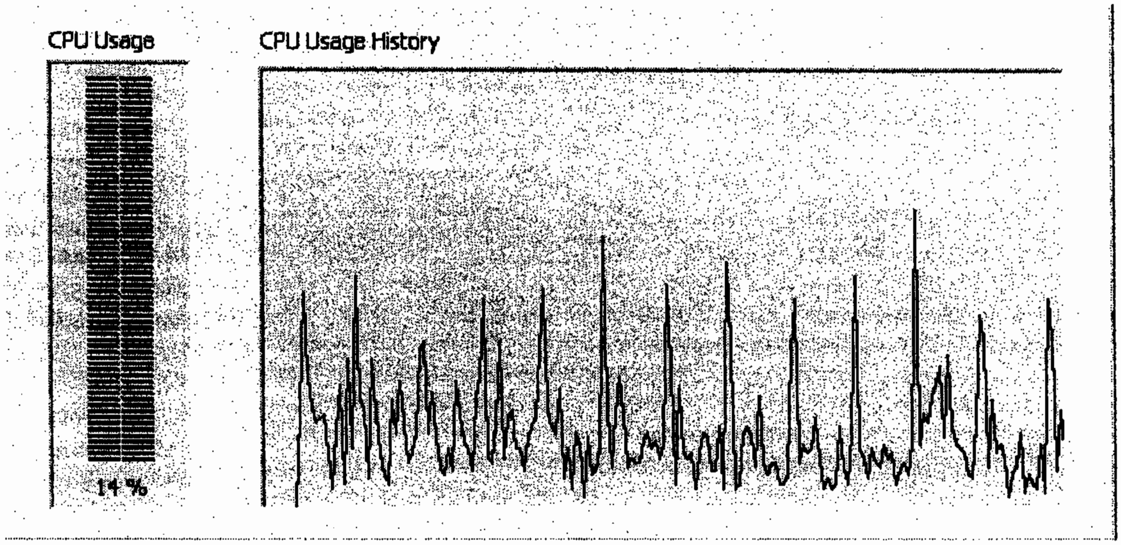
```
21:48:45 SQL> merge into destination_student@fromorcl1toorcl d
21:50:34   2  using source_student s
21:50:34   3  on (d.emp_no = s.emp_no)
21:50:34   4  when matched then
21:50:34   5  update set
21:50:34   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
21:50:34   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
21:50:34   8      d.hobby=s.hobby,d.skills=s.skills
21:50:34   9  when not matched then
21:50:34  10      insert (d.emp_no,d.fname,d.mname,
21:50:34  11              d.lname,d.address,d.ph,
21:50:34  12              d.program,d.hobby,d.skills)
21:50:34  13      values (s.emp_no,s.fname,s.mname,
21:50:34  14              s.lname,s.address,s.ph,
21:50:34  15              s.program,s.hobby,s.skills)
21:50:52  16  ;

4799998 rows merged.

Elapsed: 00:39:17.90
22:50:11 SQL> commit;

Commit complete.
```

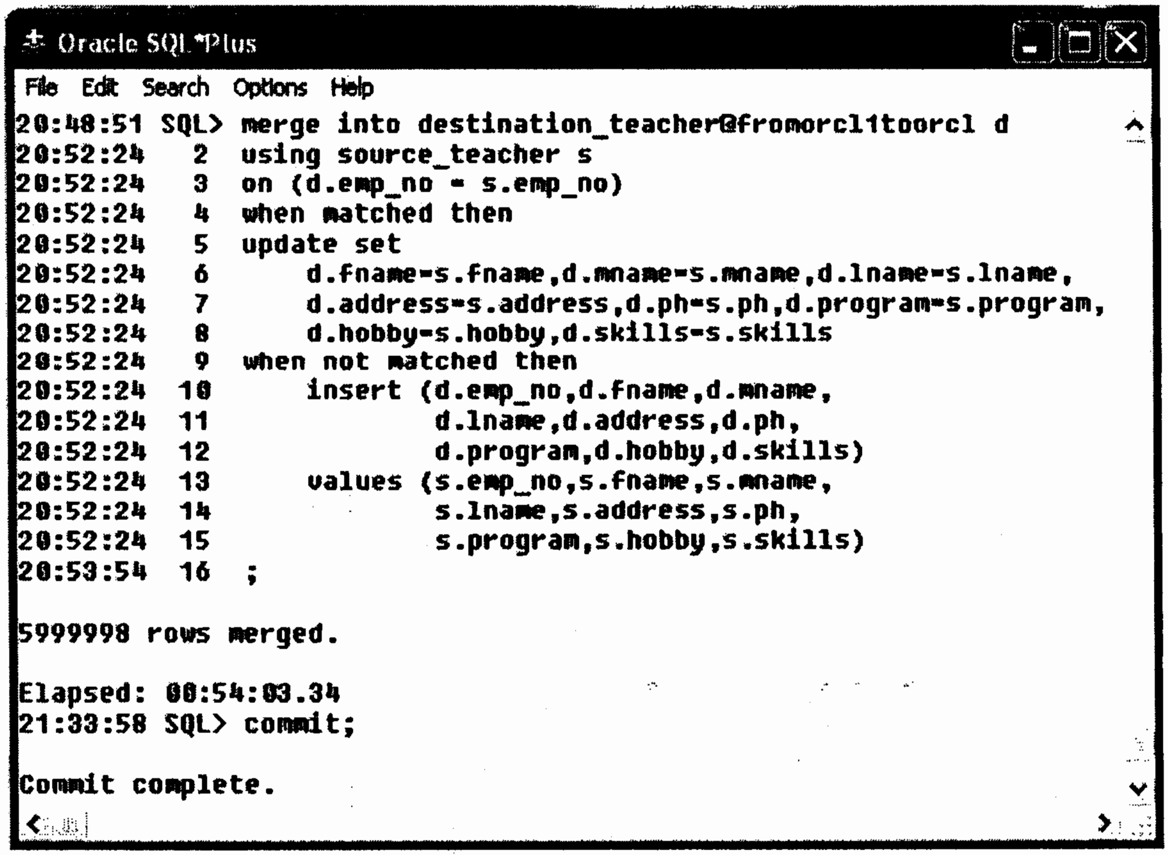


```
21:48:56 SQL> merge into destination_book@fromorcl1toorcl d
21:50:47   2  using source_book s
21:50:47   3  on (d.emp_no = s.emp_no)
21:50:47   4  when matched then
21:50:47   5  update set
21:50:47   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
21:50:47   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
21:50:47   8      d.hobby=s.hobby,d.skills=s.skills
21:50:47   9  when not matched then
21:50:47  10      insert (d.emp_no,d.fname,d.mname,
21:50:47  11              d.lname,d.address,d.ph,
21:50:47  12              d.program,d.hobby,d.skills)
21:50:47  13      values (s.emp_no,s.fname,s.mname,
21:50:47  14              s.lname,s.address,s.ph,
21:50:47  15              s.program,s.hobby,s.skills)
21:50:49  16  ;

4799998 rows merged.

Elapsed: 00:41:47.31
23:32:37 SQL> commit;

Commit complete.
```

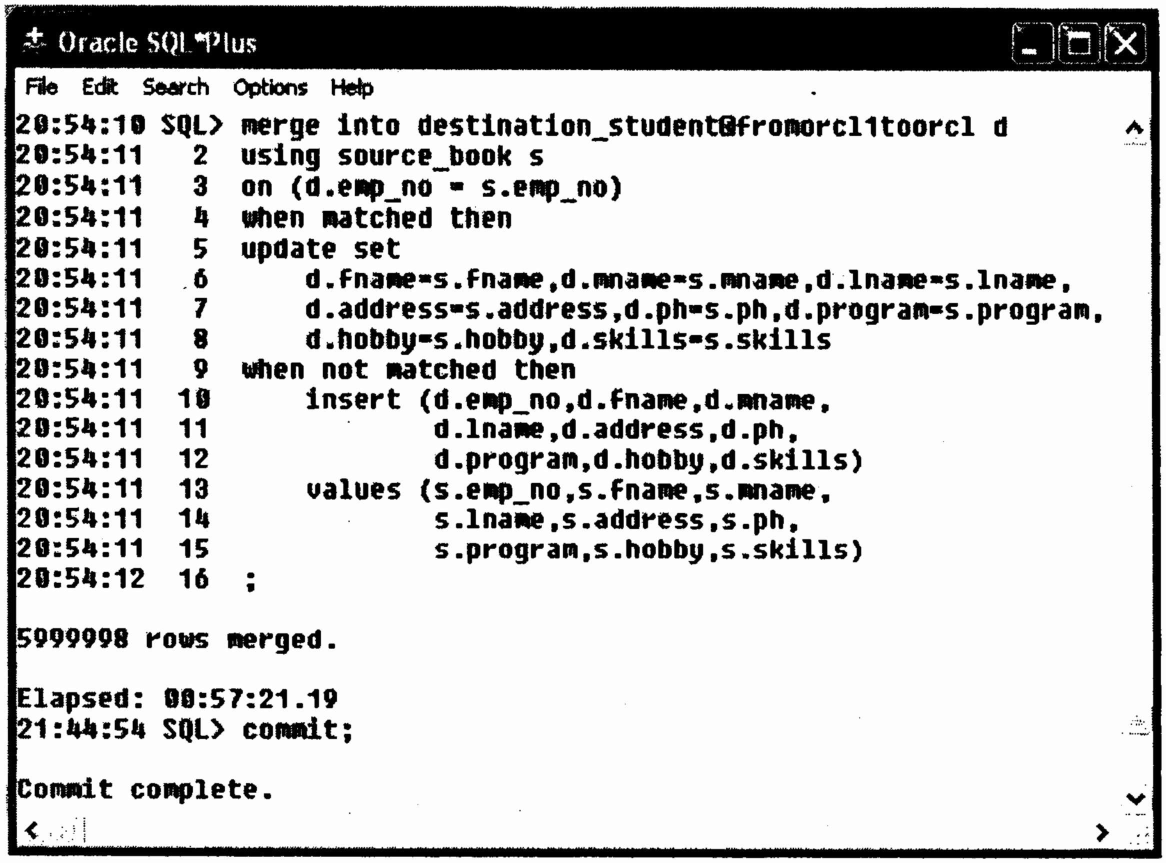
- **Day Five Data:**

The CPU usage shows that the source system is busy up to 14% when whole data is being extracted and move to DSA.

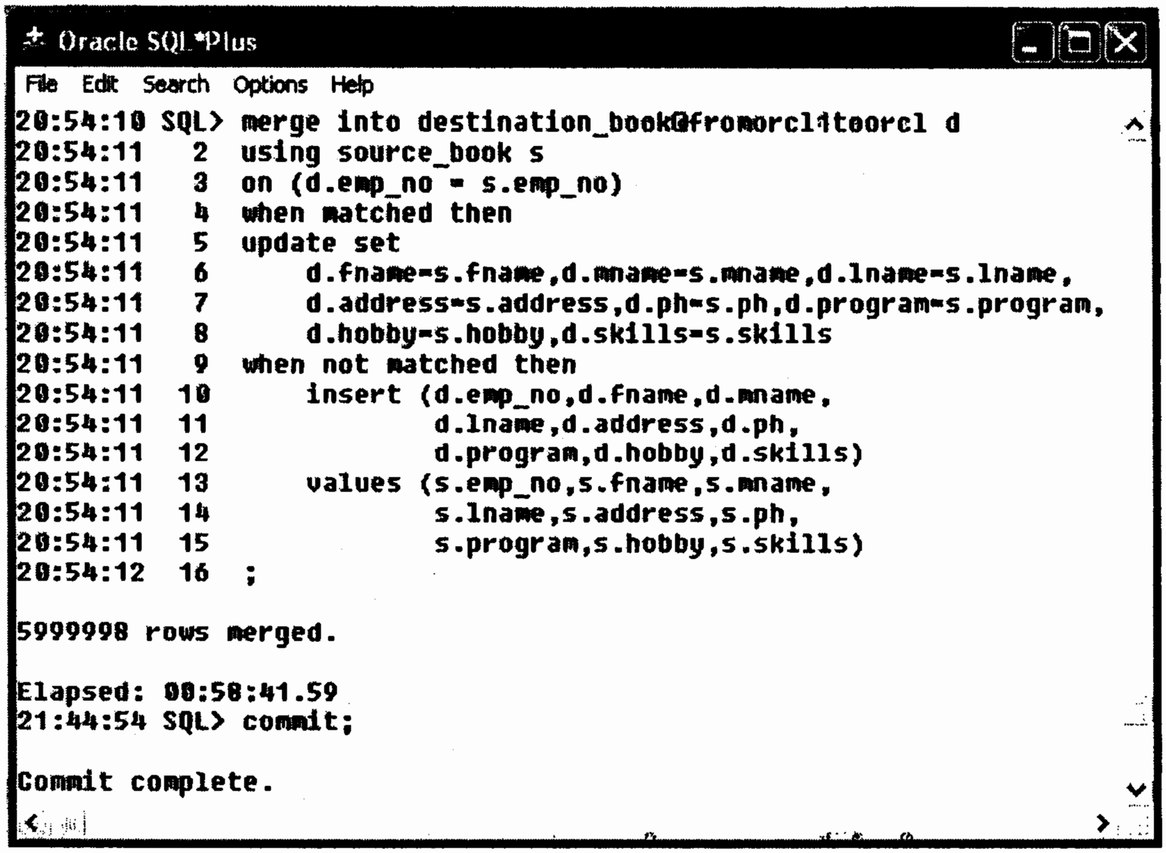




```
20:48:51 SQL> merge into destination_teacher@fromorcl1toorcl d
20:52:24   2  using source_teacher s
20:52:24   3  on (d.emp_no = s.emp_no)
20:52:24   4  when matched then
20:52:24   5  update set
20:52:24   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
20:52:24   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
20:52:24   8      d.hobby=s.hobby,d.skills=s.skills
20:52:24   9  when not matched then
20:52:24  10      insert (d.emp_no,d.fname,d.mname,
20:52:24  11              d.lname,d.address,d.ph,
20:52:24  12              d.program,d.hobby,d.skills)
20:52:24  13      values (s.emp_no,s.fname,s.mname,
20:52:24  14              s.lname,s.address,s.ph,
20:52:24  15              s.program,s.hobby,s.skills)
20:53:54  16  ;

5999998 rows merged.

Elapsed: 00:54:03.34
21:33:58 SQL> commit;

Commit complete.
```



```
20:54:10 SQL> merge into destination_student@fromorcl1toorcl d
20:54:11   2  using source_book s
20:54:11   3  on (d.emp_no = s.emp_no)
20:54:11   4  when matched then
20:54:11   5  update set
20:54:11   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
20:54:11   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
20:54:11   8      d.hobby=s.hobby,d.skills=s.skills
20:54:11   9  when not matched then
20:54:11  10      insert (d.emp_no,d.fname,d.mname,
20:54:11  11              d.lname,d.address,d.ph,
20:54:11  12              d.program,d.hobby,d.skills)
20:54:11  13      values (s.emp_no,s.fname,s.mname,
20:54:11  14              s.lname,s.address,s.ph,
20:54:11  15              s.program,s.hobby,s.skills)
20:54:12  16  ;

5999998 rows merged.

Elapsed: 00:57:21.19
21:44:54 SQL> commit;

Commit complete.
```



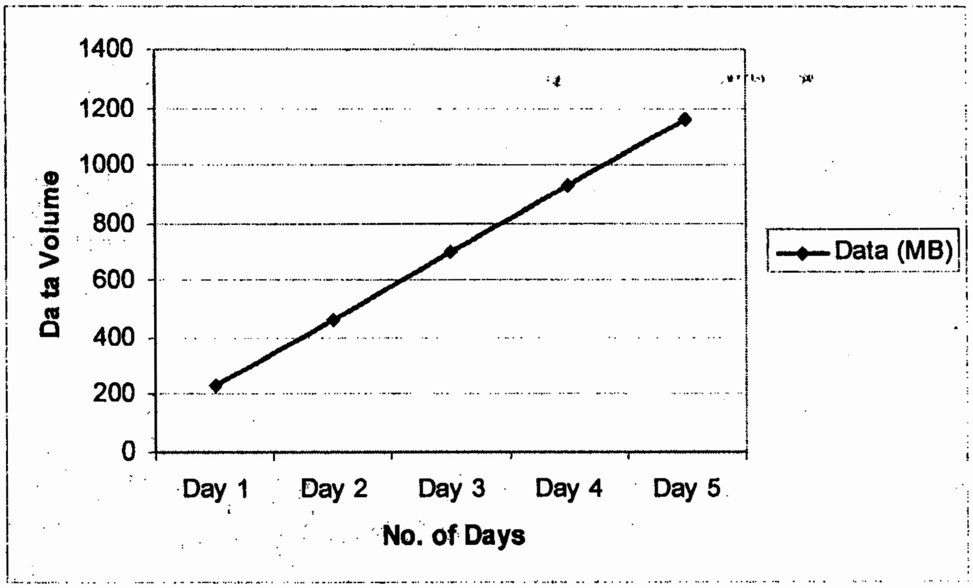
```
20:54:10 SQL> merge into destination_book@fromorcl1toorcl d
20:54:11   2  using source_book s
20:54:11   3  on (d.emp_no = s.emp_no)
20:54:11   4  when matched then
20:54:11   5  update set
20:54:11   6      d.fname=s.fname,d.mname=s.mname,d.lname=s.lname,
20:54:11   7      d.address=s.address,d.ph=s.ph,d.program=s.program,
20:54:11   8      d.hobby=s.hobby,d.skills=s.skills
20:54:11   9  when not matched then
20:54:11  10      insert (d.emp_no,d.fname,d.mname,
20:54:11  11              d.lname,d.address,d.ph,
20:54:11  12              d.program,d.hobby,d.skills)
20:54:11  13      values (s.emp_no,s.fname,s.mname,
20:54:11  14              s.lname,s.address,s.ph,
20:54:11  15              s.program,s.hobby,s.skills)
20:54:12  16  ;

5999998 rows merged.

Elapsed: 00:58:41.59
21:44:54 SQL> commit;

Commit complete.
```

4.10. Performance Evaluation:

The following table shows that the source data increase per day, so time will also be automatically increased for extraction. It has been also showed with the help of graph. So if we extract whole data during peak hours then the source system will be slowed down which will not only slow the performance of foundation system but also the process of extraction and this is not acceptable for customer.

Table: Data per day

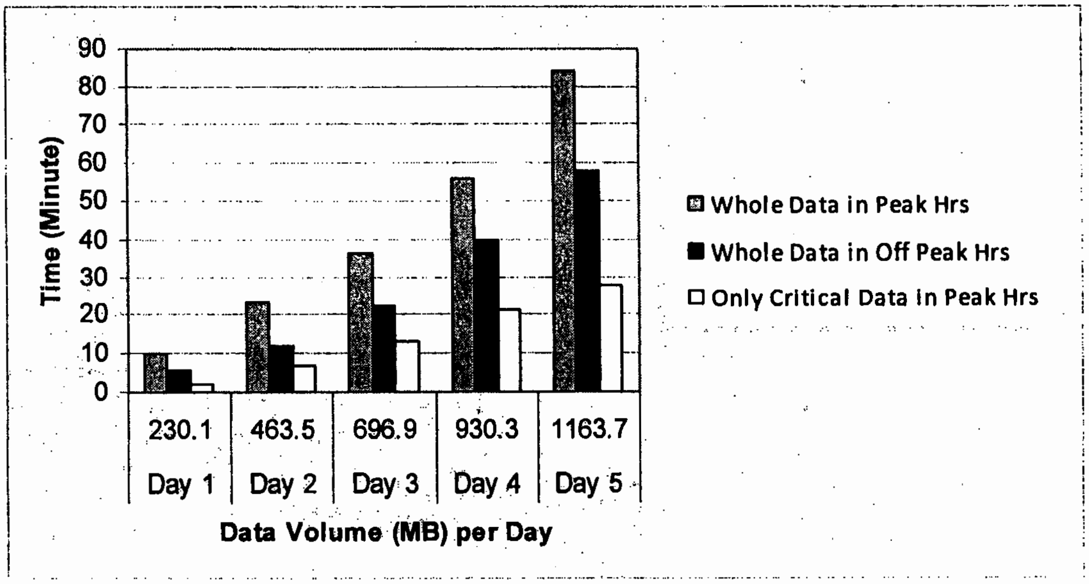
No. of Days	Data (MB)
[redacted]	[redacted]

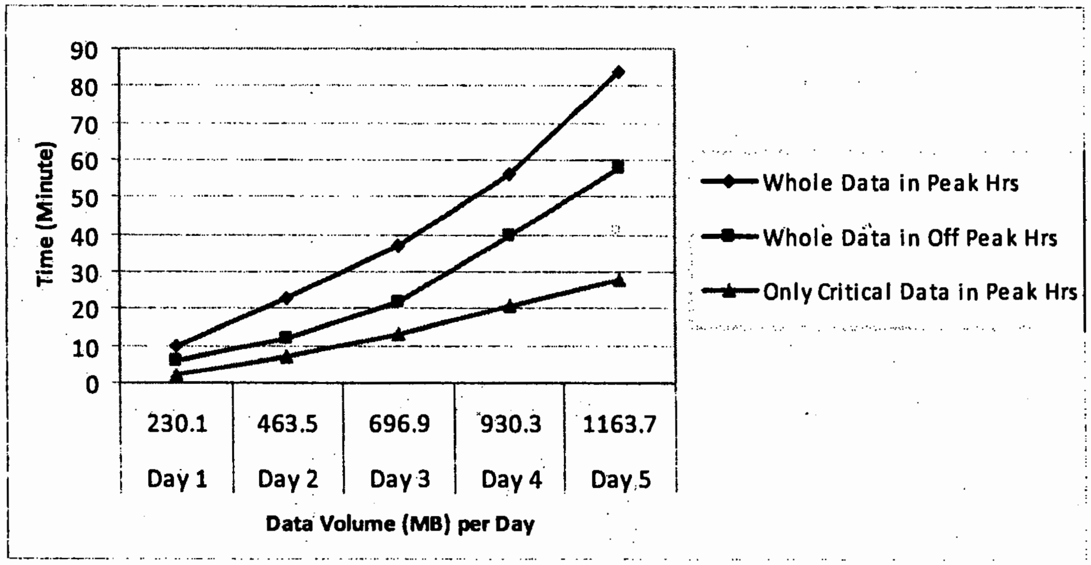


The following table shows the time in minutes taken by extraction of whole data in peak hrs and in off peak hrs as well as extraction of only critical data in peak hrs. The purpose of extraction of only critical data is to support decision process while extracting only required data during peak hrs which support to update the DWH on right time for taking a good and quick decision.

Table: Results of the proposed framework

		Time (Minute)		
No. of Days	Data (MB)	[illegible]	[illegible]	[illegible]
Day 1	[illegible]	[illegible]	[illegible]	[illegible]
Day 2	[illegible]	[illegible]	[illegible]	[illegible]
Day 3	[illegible]	[illegible]	[illegible]	[illegible]
Day 4	[illegible]	[illegible]	[illegible]	[illegible]
Day 5	[illegible]	[illegible]	[illegible]	[illegible]




90
80
70
60
50
40
30
20
10
0
Time (Minute)
230.1
463.5
696.9
930.3
1163.7
Day 1
Day 2
Day 3
Day 4
Day 5
Data Volume (MB) per Day
Whole Data in Peak Hrs
Whole Data in Off Peak Hrs
Only Critical Data in Peak Hrs

Chapter #5
Conclusion

5.1. Conclusion

It is concluded that DWH could be refreshed on near real-time or at right time according to the business rules defined. It is also concluded that near real-time data refresh can play an important role for supporting good decision making process. It is observed that the picking of only critical volume of data produces significant reduction of extraction time. The main aim of this research was to develop a technique to select only critical volume of data according to the business rules defined for extraction which can support the decision making process at right time.

Near real-time propagation of workflow audit data is very critical, because it allows workflow participants and process analysts an early detection of weaknesses and problems in the process execution. In this research, it has been discussed the challenging issues of integrating workflow audit trail data into data warehouse systems. It has been presented a new model in this research for that enables a near real-time data integration and provides services for the extraction, parsing and translation of workflow.

Appendix A
References

6. References

[1] Youchan Zhu, Lei An, Shuangxi Liu, "Data Updating and Query in Real-time Data Warehouse System", IEEE International Conference on Computer Science and Software Engineering, pp. 1295-1297, 2008.

[2] Alexandros Karakasidis, Panos Vassiliadis, Evaggelia Pitoura "ETL Queues for Active Data Warehousing", ACM, Baltimore, MD, USA, pp. 28-39. June 2005.

[3] M. Asif Naeem, Gillian Dobbie, Gerald Weber "An Event-Based Near Real-Time Data Integration Architecture" IEEE, 2008.

[4] Li Chen, Wenny Rahayu, David Taniar "Towards Near Real-Time Data Warehousing" in 24th IEEE International Conference on Advanced Information Networking and Applications 2010.

[5] JinGang Shi, YuBin Bao, FangLing Leng, Ge Yu "Study on Log-Based Change Data Capture and Handling Mechanism in Real-Time Data Warehouse" IEEE International Conference on Computer Science and Software Engineering, pp. 478-481, 2008.

[6] Dr.Muhammad Younus Javed, Asim Nawaz "Data Load Distribution by Semi Real Time Data Warehouse", IEEE, 2nd International Conference on Computer and Network Technology, pp. 556-560, 2010.

[7] Mohammad Rifaie Keivan Kianmehr Reda Alhajj Mick J. Ridley "Data Warehouse Architecture and Design", pp 58-63, IEEE Las Vegas, Nevada, USA, 2008.

[8] J.M. Artz, How Good is that Data in the Warehouse?, The Data Base for Advances in Information Systems – Summer 1997 (Vol. 28, No. 3).

[9] Steven K. Ball, Do You Need a Data Warehouse Layer in Your Business Intelligence Architecture?, DM Direct, September 2005.

[10] Frank Bensberg, Controlling the Data Warehouse – A Balanced Scorecard Approach, In the Proceedings of the 25th International Conference on Information Technology Interfaces ITI 2003, June 16-19, 2003, Cavtat, Croatia.

[11] Jorge Bernardino and Henrique Madeira, Experimental Evaluation of a New Distributed Partitioning Technique for Data Warehouses, In proceedings of the International Database Engineering & Applications Symposium, 2001.

[12] Michele Bokun and Carmen Taglienti, Incremental Data Warehouse Updates, DM Review, May 1998.

[13] Stephen Brobst, The Future of Data Warehousing. Available online at http://www.ncr.com/, Spring 2004.

[14] S. Chaudhuri and U. Dayal, An Overview of Data Warehousing and OLAP Technology. SIGMOD Record 26(1), March 1997.

[15] Kemal A. Delic, Laurent Douillet and Umeshwar Dayal, Towards an Architecture for Real-Time Decision Support Systems: Challenges and Solutions, In proceedings of the International Database Engineering & Applications Symposium (IDEAS '01), 2001.

[16] Jim Ericson, The Right "Real-Time", Business Intelligence Report, September 2005.

[17] Forrester Research, Inc., Real-Time Data Warehousing: The Hype and The Reality, Available online at http://www.forrester.com/Research/Document/Excerpt/0,7211,36076,00.html, December 15, 2004.

[18] Gerri Furlow, The Case for Building a Data Warehouse, IT Professional, vol. 03, no. 4, pp. 31-34, July/August, 2001.

[19] Stephen R. Gardner, Building the Data Warehouse, Communications of the ACM, Volume 41, Number 9, September, 1998.

[20] Richard Hackathorn, Current Practices in Active Data Warehousing, Bolder Technology, Inc., Available online at http://www.teradata.com/t/pdf.aspx?a=83673&b=86843, 2002.

[21] J.H. Hanson and M.J. Willshire, Modeling a faster data warehouse, International Database Engineering and Applications Symposium (IDEAS '97).

[22] Lynn Hedegard, Teradata's Real-Time Enterprise Reference Architecture, http://www.teradata.com/t/pdf.aspx?a=83673&b=132245, January 2005.

[23] Hired Brains, Inc., Exploring the Business Imperative of Real-Time Analytics, http://www.informationweek.com/story/shotArticle.jhtml?articleID=1280297, 2003.

[24] Intelligent Solutions, Inc., Active Data Warehousing – the Ultimate Fulfillment of the Operational Data Store, 2001, Retrieved from http://www.teradata.com/t/pdf.aspx?a=83673&b=86856 on December 14, 2005.

[25] Alexandros Karakasidis, Panos Vassiliadis, Evaggelia Pitoura, ETL Queues for Active Data Warehousing, 2nd International Workshop on Information Quality in Information Systems, IQIS 2005, June 17, 2005, Baltimore, MD, USA.

[26] Wilburt Labio, Jun Yang, Yingwei Cui, Hector Garcia-Molina, Jennifer Widom, Performance issues in Incremental Warehouse Maintenance, In proceedings of the VLDB, Cairo, Egypt, September 2000.

[27] Oracle Corporation White Paper, On-Time Data Warehousing with Oracle10g - Information at the Speed of your Business, 2003. Retrieved from http://www.oracle.com/technology/products/bi/pdf/10gr1_twp_bi_ontime_etl.pdf on December 14, 2005.

[28] Nayem Rahman, Intelligent Metadata Model in a Teradata Warehousing Environment, Annual Teradata PARTNERS User Group Conference and Expo, September 18-22, 2005, Walt Disney World Swan/ Dolphin Resort – Orlando, FL, USA.

[29] Rifaieh Rami, Benharkat Aicha Nabila, Query-based Data Warehousing Tool, ACM Fifth International Workshop on Data Warehousing and OLAP (DOLAP 2002) McLean, VA, USA November 8, 2002.

[30] Holger Schwarz, Ralf Wagner, Bernhard Mitschang, Improving the Processing of Decision Support Queries: The Case for a DSS Optimizer, In proceedings of the International Database Engineering & Applications Symposium. (IDEAS '01), 2001.

[31] Bongsik Shin, An Exploratory Investigation of System Success Factors in Data Warehousing, Journal of the Association for Information Systems (Volume 4, 2003) pp. 141-170.

[32] Alkis Simitsis, Panos Vassiliadis, Timos Sellis, Optimizing ETL Processes in Data Warehouses, Proceedings of the 21st International Conference on Data Engineering (ICDE'05), 2005.

[33] Sunopsis, ETL: The Emerging Standard Eclipses ETL, Retrieved from www.sunopsis.com on December 14, 2005.

[34] A. Vavouras, S. Gatziu, K.R. Dittrich, Modeling and Executing the Data Warehouse Refreshment Process, International Symposium on Database Applications in Non-Traditional Environments (DANTE'99).

[35] Hugh J. Watson, Ceila Fuller, Thilini Ariyachandra, Data Warehouse Governance: Best Practices at Blue Cross and Blue Shield of North Carolina, Decision Support Systems 38 (2004) pp. 435-450.

[36] J. Widom, Research Problems in Data Warehousing, In proceedings of the 4th Int'l Conference on Information and Knowledge Management (CIKM), November 1995.

[37] Shuigeng Zhou, Aoying Zhou, Xiaopeng Tao, Yunfa Hu, Hierarchically Distributed Data Warehouse, In proceedings of the Fourth International Conference on High-Performance Computing in the Asia-Pacific Region-Volume 2 - Volume 2, 2000.

[38] Gartner, ETL Magic Quadrant Update: Market Pressure Increases. Retrieved from Google on December 20, 2005.

[39] http://data-warehouses.net/.

[40] http://youngcow.net/doc/oracle10g/server.102/b14223/extract.htm

[41] http://forum.kimballgroup.com/deployment-and-maintenance-f8/real-time-datawarehousing-t122.htm