Detecting and Representing Relevant Web Deltas in WHOWEDA

Project Report p-1210

Submitted in partial fulfillment of the
Requirement for the award of the degree of the

Bachelor of Engineering in Computer Science and Engineering
Of
Bharathiar University, Coimbatore.

Submitted by

S. Muralidharan
A.Raju

Under the guidance of

Mrs. V.Vanitha M.E.,
Senior Lecturer,
Department of Computer Science and Engineering

Department Of Computer Science and Engineering
KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE –641 006

MARCH 2004.

KUMARAGURU COLLEGE OF TECHNOLOGY
(Affiliated to Bharathiar University, Coimbatore)

Department Of Computer Science and Engineering

CERTIFICATE

This is to certify that the project entitled

Detecting and Representing Relevant Web Deltas in WHOWEDA

is done by

S. Muralidharan
(0027k0184)

A.Raju
(0027k0194)

In partial fulfillment of the
requirements for the award of the degree of

Bachelor of Computer Science and Engineering
Bharathiar University, Coimbatore
During the academic year 2003-2004

S. Thangasamy

Professor & Head of the Department
(Dr.S.Thangasamy)

22/3/04

Guide
(Mrs.V.Vanitha)

Certified that the candidates were examined by us in the project
Viva-voce examination held on 24.03.04

24/3/04

Internal Examiner

External Examiner

DECLARATION

Declaration

We,

S.Muralidharan	**0027K0184**
A.Raju	**0027k0194**

declare that the project entitled "**Detecting and Representing Relevant Web Deltas in WHOWEDA**", is done by us and to the best of our knowledge, a similar work has not been submitted earlier to the Bharathiar University or any other institution, for fulfillment of the requirement of the course study.

This project report is submitted on the partial fulfillment of the requirement for all awards of the degree of Bachelor of Computer Science and Engineering of Bharathiar University.

S Muralidharan

Place: Coimbatore. **S. Muralidharan**

Date : 22.03.04

A. Raju

A.Raju

Countersigned by Staff- in- charge,

V Vtha 22/3/04

Mrs.V.Vanitha M.E.,
Senior Lecturer,
Department of Computer Science and Engineering,
Kumaraguru College of Technology.

DEDICATED TO
OUR BELOVED
PARENTS AND FRIENDS

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We take this opportunity to thank the management of our college for having provided us excellent facilities to work with. We express our deep gratitude to our Principal **Dr.K.K.Padmanabhan B.Sc (Engg), M.Tech.**, for ushering us in the path of triumph.

We are always thankful to our beloved Professor and the Head, Department of Computer Science and Engineering, **Dr.S.Thangasamy B.E.(HONS).**, whose consistent support and enthusiastic involvement helped us a great deal.

We are greatly indebted to our beloved guide **Mrs.V.Vanitha M.E.**, Senior Lecturer, Department of Computer Science and Engineering, for her excellent guidance and timely support during the course of this project.

We also thank our project coordinator **Mrs.S.Chandrakala M.E.**, Senior Lecturer, Department Of Computer Science and Engineering, and our beloved class advisor **Mrs.Hema B.E.**, Lecturer, Department of Computer Science and Engineering, for their invaluable assistance.

We feel proud to pay our respectful thanks to our Parents for their enthusiasm and encouragement and also we thank our friends who have associated themselves to bring out this project successfully.

SYNOPSIS

SYNOPSIS

The project entitled "Detecting and representing Relevant Web Deltas in WHOWEDA" is a mechanism for detecting and representing changes, given the old and new versions of a set of interlinked Web documents, retrieved in response to a user's query. In particular, we show how to detect and represent Web deltas, i.e., changes in the Web documents that are relevant to user's query in context of our Web warehousing system called WHOWEDA (Warehouse of Web Data).

The Web offers access to large amounts of heterogeneous information and allows this information to change at any time and in any way. These rapid and often unpredictable changes to the information create a new problem of detecting and representing changes. This is a challenging problem because the information in the Web are autonomous and typical database approaches to detect changes based on triggering mechanisms are not usable.

In WHOWEDA, Web information is materialized views stored in Web tables in the form of Web tuples. These Web tuples, represented as directed graphs, can be manipulated using the Web algebraic operator.

In this project, we present a mechanism to detect relevant Web deltas using Web algebraic operator such as Web join. The Web join operator with the change detection criterion as input and outputs only those Web tuples that are new or modified or deleted tables. We also show how to represent these changes using the delta Web tables.

CONTENTS

CONTENTS

INTRODUCTION

INTRODUCTION

Existing System and Limitations:

AT&T Internet Difference Engine (AIDE) is a system that finds and displays changes to pages on the World Wide Web. AIDE also supports recursive tracking and differencing of a page and its descendants. A user may specify a number of operations in AIDE which includes registering a URL including the degree of recursion through links to other pages, view textual differences between a pair of versions, and view a graph showing the structure of a page, etc.

WebGUIDE (Web Graphical User Interface to a Difference Engine) is another tool that supports recursive document comparison: user may explore the difference between the pages with respect to two dates.

The AIDE and WebGUIDE have certain limitations. First, the recursive specification is restrictive, i.e., it selects all the children's of a specified document, however in reality, a user may often be interested in only some of those links. On top of that, the user may wish to track changes of successive interlinked documents, satisfying some hyperlinked structure. Such constraints cannot be specified in AIDE. Second, AIDE displays all of the changes in the documents. In reality, we may not be interested in all of these changes. Hence it is necessary to be able to query these changes rather than browsing them to find the relevant changes. This is useful when the number of documents monitored is large.

Proposed System and Advantage:

Here we present a mechanism for detecting and representing changes, given the older and new versions of a set of interlinked web documents, retrieved in response to user's query. In particular we have demonstrated how to detect and represent changes in the web data using a set of Web algebraic operator in the context of WHOWEDA (Warehouse of Web Data), a data warehousing system for managing and manipulating relevant data extracted from the web. To facilitate manipulation of web data stored in the web tables we have identified a set of web algebraic operators such as web coupling, web join and web select.

Web Coupling-It enables a user to retrieve a set of interlinked documents satisfying a *Web query*, regardless of the locations of the documents in the web. This operator τ, takes in a coupling query G and returns a web tables W={Z,S,T} containing a set of Web tuples T extracted from the World Wide Web satisfying the query and a set of web schemas S generated from G and T.

Web Join-This operator is used to combine identical data residing in two Web tables (say W1 and W2) containing joinable nodes (nodes participating in Web Join operation) are concatenated into a single joined web tuple that can be materialized in a Web table.

Web Select-This operator enables the user to query the delta web tables for required information's in the requested format.

The proposed mechanism is useful for the following type of users,

Web site administrator – By scanning the changes administrator will be sure whether the changes are consistent with any policies for content or format without having to review the entire set of pages at the same level of details.

Customers of E-commerce web site- A user may wish to monitor new products, services, or auctions on E-commerce web site.

Wireless users- The ability to download or highlight only changes instead of complete web page can be a very desirable feature for wireless user using handheld devices.

Analysts for gathering competitive intelligence- Companies can monitor evolution of their competitor's web site to discover their new directions or offering over a period of time that may influence their market positions.

SOFTWARE REQUIREMENT ANALYSIS

SOFTWARE REQUIREMENT ANALYSIS

Scope

To develop a system that detects any changes in a web document given two snap shots of same web document.

Product Overview

The Web offers access to large amount of heterogeneous information and allows this information to change at any time and in any way. These rapid and often unpredictable changes to the information create a new problem of detecting and representing changes. The system proposed detects and represents changes in Web document (referred to as WebDelta) which are relevant to user's query using two *Web algebraic operators*, i.e., *Web join* and *Web Select* in the context of our Web Warehousing system called WHOWEDA.

Development and Operating Environment

Hardware Specification

Processor	Pentium III
RAM	128 MB
Hard Disk	10 GB
Floppy Drive	1.44 FDD

Software Specification

Operating System	Windows xp
Front End	Visual Basic .NET
Back End	MS SQL 7.0

Implementation of Proposed System

The entire system is decomposed into three modules as mentioned,

Module I- **Identifying Relevant Websites**

The web site with its set of interlinked documents which satisfy the user's query has to be analyzed to identify the required set of links that contains information of user's interest. Here we have programmed for the web site in which the selection is based on two strategies-

Web Site with,

Abundant Information

Frequent Updations

Abundant Information:

This program, we have done to demonstrate the efficient working of the parser with such a lot data. The Web site we have chosen is www.made-in-china.com an official Web site of Chinese Government that materializes the information's about the products manufactured in China. The product are categorized and sub-categorized through a sequence of interlinks. For this Website, we have designed the database as follows, the tables in the made-in-china Website is: Category table, Subcategory table, Manufacture table and the Product table.

The database is designed according to the fields that are present in Website chosen. So the identification of suitable websites plays a vital role in our project. The Category table contains the main product lists that are available in the made-in-china Website. For each main category lists there may be some Subcategories available. These subcategories are present in the table Subcategory.

The Manufacture table contains all the details about the manufacturer whose is manufacturing the product. Some of the details about the manufacturer that are available in the made-in-china Website are as follows: the manufacture name, manufacture profile, manufactures address, city, country etc.. The Product table contains all the required information about the products that are available in the Website. Some of the details about the products are: product code, product name, product trademark, product model, standard, etc, Our aim here is to watch for any new products in the Chinese market. For the bdjobs Website, the database contains the table as Details. This table contains all the information about the companies, their products etc.

Frequent Updation:

This program will demonstrate the overall performance of the system. The necessity assumed here is that the user wants to know any new entry either in the main category list or in the subcategory list and also the user wants to know any changes that are made in product list in the made-in-china Website. In the bdjobs Website the user wants to know the company list in which any new company has available as a new entry and also for new products. So websites with frequent updations are chosen for our project.

Module II- **Parsing**

The Parser module performs the extraction of information from the Web documents that are retrieved on the basis of users query for that information. This module also plays a vital role in our project, because based on this retrieval of these data's only we are going to compare and represent the changes according to the user's query. The working of parser begins with passing a HTML query to the Web site seeking the source code of the desired Web page. The parser works on this source code and outputs the extracted data.

The working is as follows: Initially the source code will be stored in a string variable. Then this variable is run through a series of substring match that searches for the match of necessary hyperlinks and extracts the associated information's. The information that is required for our project are extracted from the source code of the Web page. The information's in the form of tables are also extracted by searching for the table tag in the source code.
This retrieval operation is called the Web Coupling Operation. These extracted information's are then inserted into the appropriate Web Tables that are designed for the Websites in WHOWEDA.

Module III- **Database Manipulation**

In this module the major operation of finding the changes in the Website is done. The mechanism for Detection and Representation of Changes problem is handled here through two major operations. They are:

Web Join Operation
Web Select Operation

Web Join operation:

As changes in relevant Web data are reflected on a web table, we can address the problem of detecting and representing changes to web data in the context of such Web tables. Let the user specify a polling times t1 and t2 of a particular Web document between which there might be possible changes in the contents which is to be detected and represented. For example for the bdjobs Website, at t1 Web coupling operator retrieves a set of interlinked documents and materializes them in the form of a Web table 'Details' . At t2 the same set of interlinked documents will be in the Web table 'Detailsnext'.

Given two such Web tables containing the snapshots of two versions of relevant Web data, the problem of change detection is to find the set of Web tuples containing nodes which are inserted into or deleted from 'Details' or those nodes which are modified in 'Details' to transform it into 'Detailsnext'.
Note that these Web tuples will denote the changes to the Web document that are relevant to the user. The Webjoin operator takes the Web tables *t1* and *t2* of two times respectively along with the change detection criterion as input and outputs only those Web tuples that are new or modified or deleted.
These tuples are materialized in a separate Web table for further query. The *Web Select* operation takes care of providing the requested information from the resultant table to the user.

The Changes to the Web tables are reflected on the individual Web tuples in WHOWEDA. Consequently, the different types of change operations in WHOWEDA can be defined in terms of the following:

Insert Node: Intuitively, the Insert Node operation creates a set of nodes *N* in a Web tuple in the Web table *W*. The nodes must be new i.e., *N* must not occur in *W* before. Observe that *N* can be a new Web tuple added to *W* or it can be a set of node inserted into an existing Web tuple in *W*.

Delete Node: This operation is the inverse of the Insert Node operation. It removes a set of nodes fro *W.*

Update Node: The operation Update Node modifies the content of the node, we mean the textual contents or structure of the node may change or the attributes of the links embedded in the node may change.

Insert Link: Intuitively, the Insert Link operation creates a set of links *L* in a Web tuple in the Web table *W*. The links must be new i.e., *L* must not occur in *W* before. Observe that, in a Web table, a new link can occur in two ways.

First, a new link can connect an existing node to a new node. In this case, The *Insert Link* results in an *Insert Node* operation. Second, a new link may connect to existing nodes in a Web tuple.
In this case, the *Insert Link* operation is equivalent to the *Update Node* operation as the source node of the link has to be modified in order to incorporate the new link. So, we can express the *Insert Link* operation by the *Insert Node* or *Update Node* operation.

Delete Link: It removes a set of links from *W*. Similar to *Insert Link,* in a Web table, deletion of a link may occur based on two cases. First, removal of a link may remove the only link to an existing node. In this case, it is equivalent to the *Delete Node* operation. Second, a node may be deleted between two nodes which are connected by more than one link. In this case, this operation is essentially the *Update Node* operation. Therefore, we can express the *Delete Link* operation by the *Delete Node* or *Update Node* operation.

Representing Changes

The problem of detecting and representing changes can now be decomposed into two parts: First is the construction of joined and inner joined Web tables from the two tables. Next step by using the joined and inner joined Web tables generate a set of Web tables i.e., containing the Web deltas. We define a structure called the delta Web table for representing Web deltas. Delta Web tables encapsulate the relevant changes that have occurred in the Web with respect to a user's query. The Webselect operator is used to represent the changes that are available in the joined webtable according to the user's query.

Exception Handling

This proposed system can handle only consistent changes in the web document namely the change in the format of document will render the system useless.

RESULTS OBTAINED

RESULTS OBTAINED

The Purpose of the Testing is to verify that a correct system is being built and is performed with the intent of finding faults in the system and to produce the correct results. So, to verify whether our system is working properly or not it is essential to check our system in parallel with all the stages of system development, starting with requirement specification itself.

Testing of results for the consistency of the procured data plays vital role in determining the reliability of the system. Here we have to be sure of that the information's gathered are the necessary one and are enough to satisfy any user queries. The project consists of three modules out of which the first module, Web site analysis is a General Study Module. The remaining two involves direct coding and works on the data.

In this project, the results are verified from starting of the parsing module itself. Because it is very essential to extract only the necessary data's from the Webpage for the comparison operations and also it is essential to check the system for redundancy. Because once the parsing is started, the parser extracts the necessary data from the webpage. In case of any disturbances like power failure, disconnection of the net, etc., we need to start the parsing from starting stage itself. At these circumstances there may be a chance for the repetations of the data's. So we also check for not any repetation of same data.

The Parser module involves retrieving required HTML document and extracting necessary information's. The downloaded source code should here be checked whether that was the requested page. This we did by comparing the two files one containing the code obtained through the browser and other through our program. Again the information that has been extracted through the Parser has to be checked for consistency which we did manually comparing the data in the HTML document with those that were recently inserted into the Web table both of

same time snap. The problem encountered here was the redundancy in the nodes. For example in the program for www.made-in-china.com we are in the perspective for new products in the market, the Web table containing the Company Details associated with this program was found redundant as that particular company manufactures more than one product. So here during each entry of Company Details it should be checked for non-redundancy. Similarly if we are programming for the active companies then there might be a redundancy in the Product Details Web table since there be many companies producing same good. For the first time of parsing all the required data's are stored in the tables that are defined in the database. For example for the Bangladeshi website, for first time of parsing the data's are stored in the table Details. At certain time if interval again the parsing is done for same Website. But this time, the data's are stored in a new table called the Detailsnext that is to be defined in the same database of that Website. For the made-in-china Website the necessary data's are stored in the tables that are defined for that Website.

The third module, Database Manipulation involves certain operations like Join and Inner Join, used to combine two tables containing joinable nodes. These joinable nodes are the values that are to be checked for any changes just described. An interesting bug that was uncovered here was that some nodes that were not subjected to changes were also able to get through the join operation. Later it was found to be due to the joinable nodes containing NULL values. The cause was the corresponding information field in the web document was left unfilled. This NULL value can't be compared and hence those nodes containing the NULL value at the joinable node created the effect of changes in that field. We overcame this problem by inserting blank space instead of NULL value for the corresponding fields.

Next step is to represent the new entry data's that are available in the Website. The changes may happen at any level, i.e., there may be some changes in the main category itself and there are also changes available in the subcategory level. So it is essential to detect and represent those changes. For the Bangladeshi garment Website the changes are reflected based on the following condition: Here, the same company name with different address will be represented as a new entry and also the company with new name is also made as a new entry. So to represent the changes in this Website instead of comparing only the company name and company city location, it is necessary to compare the address of the company also. The changes that are present in the Website's are detected and represented according to the user's query.

FUTURE ENHANCEMENTS

FUTURE ENHANCEMENTS

[1] Currently the Web table contains tuples where only some of the nodes represent the insertion, deletion, or update operation during the transition. This is because we wish to show how these nodes are related to one another and to other nodes which have remained unchanged during the transition. Therefore, we need a mechanism to distinguish between the modified, new, or deleted nodes in each delta web tables.

[2] Design an event condition-action trigger language for WHOWEDA based on the ideas from change detection system.

CONCLUSION

CONCLUSION

Here we have defined the challenging problem of detecting and representing time based changes in the web document. To solve the problem we have identified three algebraic operators namely web coupling, web join and web select operators which detects the changes in the web delta and present them against the user's query. We have presented web deltas in the form of web tables.

REFERENCES

REFERENCES

[1] S. Abiteboul, D. Quass, I. MeHugh, I. Widom "The Query language for Semi structured Data". Apr 1997.

[2] S. S Bhowmick "WHOM, A Data Model and Algebra for Web Warehouse", 2001.

[3] Tony Martin and Dominic Selly, 2002, "Visual Basic .NET Projects". WILEY-dreamtech India Pvt, Ltd.

Web Sites Visited:

[1] URL Minder Web site http://www.netmind.com/URL-minder/URL-minder.html

[2] www.ieee.org

[3] www.made-in-china.com

[4] www.bdjobs.com

[5] www.alibaba.com

APPENDIX

SAMPLE CODE

SAMPLE SOURCE CODE

Parsing code:

```
Imports System.Data
Public Class Form1
    Inherits System.Windows.Forms.Form
#Region " Windows Form Designer generated code "

    Public Sub New()
        MyBase.New()
                InitializeComponent()
    End Sub

    Protected Overloads Overrides Sub Dispose(ByVal disposing As
Boolean)
        If disposing Then
            If Not (components Is Nothing) Then
                components.Dispose()
            End If
        End If
        MyBase.Dispose(disposing)
    End Sub
    Dim Conn As New OleDb.OleDbConnection()
    Dim cmdExec As OleDb.OleDbCommand
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        cmdExec = Conn.CreateCommand
        Conn.ConnectionString = "Provider=SQLOLEDB.1;Persist Security
Info=False;User ID=sa;pwd=;Initial Catalog=bdjobs;Data Source=."
    End Sub

Private Sub startcmd_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles startcmd.Click
        Dim strSource As String
        Dim strCategoryAnchor As String
        Dim strCompanyName As String
        Dim strCompanyUrl As String
        Dim intFirstIdx As Int64
        Dim intSecondIdx As Int64
        Dim intLength As Int64
        Dim intFirstTDIdx As Integer
        Dim intSecondTDIdx As Integer
        Dim intThirdTDIdx As Integer
        Dim intFourthTDIdx As Integer
        Dim strProductlistTD As String
        Dim strProductlist As String
        Dim strProductcapacityTD As String
        Dim strProductcapacity As String

        Call
setURL("Http://www.bdjobs.com/bgmea/search_alpha.asp?char=B")
        strSource = Trim(getHMTLSource(getUrl()))
        intFirstIdx = 1
        intFirstIdx = InStr(intFirstIdx, strSource,
"view_company.asp?id", CompareMethod.Text)
```

```
        While intFirstIdx > 0

            intSecondIdx = InStr(intFirstIdx, strSource, "</a>",
CompareMethod.Text) + 4
            intLength = intSecondIdx - intFirstIdx
            strCategoryAnchor = removeDoubleQuotes(Mid(strSource,
intFirstIdx, intLength))

            strCompanyUrl = getHyperLink(strCategoryAnchor)
            strCompanyName =
removeAmp(getHyperLink_Text(strCategoryAnchor))

            intFirstTDIdx = InStr(intFirstIdx, strSource, "<td",
CompareMethod.Text)
            intSecondTDIdx = InStr(intFirstTDIdx + 3, strSource,
"</td>", CompareMethod.Text)
            strProductlistTD = Mid(strSource, intFirstTDIdx,
intSecondTDIdx - intFirstTDIdx)
            strProductlist =
databaseModule.gettproductdetails(strProductlistTD)
            strProductlist = strProductlist.Replace(" ", "")

            intThirdTDIdx = InStr(intSecondTDIdx, strSource, "<td",
CompareMethod.Text)
            intFourthTDIdx = InStr(intThirdTDIdx + 3, strSource,
"</td>", CompareMethod.Text)
            strProductcapacityTD = Mid(strSource, intThirdTDIdx,
intFourthTDIdx - intThirdTDIdx)
            strProductcapacity =
databaseModule.gettproductdetails(strProductcapacityTD)
            strProductcapacity = strProductcapacity.Replace(" ",
"")
            Call getcompanydetails(strCompanyUrl, strCompanyName)
            intFirstIdx = InStr(intSecondIdx, strSource,
"view_company.asp?id", CompareMethod.Text)

        End While
    End Sub

    Public Function getcompanydetails(ByVal strcompanyurl As String,
ByVal strcompanyname As String)

        Dim strCompanySource As String
        Dim intcompanyfirstidx As Integer
        Dim intcompanysecondidx As Integer
        Dim strtablerow As String
        Dim strannul As String
        Dim straddress As String
        Dim strContactPerson As String
        Dim strCity As String
        Dim strDesignation As String
        Dim stremail As String
        Dim strIndustryType As String
        Dim strPhone As String
        Dim strOfficialWeb As String
        Dim strProduct As String
        Dim strFactoryLocation As String
```

```
        Dim strduplicompany As String = "select count(*) from details
where companyname='" & checkQuotation(strcompanyname) & "'"
        Dim cmdduplicompany As New OleDb.OleDbCommand(strduplicompany,
Conn)
        Dim intrecheckid As Integer
        Dim cmdrecheckidquery As New OleDb.OleDbCommand("select
isnull(max(companyid),0) from details;", Conn)
        Dim strrecheckname As String
        Dim cmdrechecknamequery As New OleDb.OleDbCommand("select
companyname from details where companyid = '" & intrecheckid & "' ;",
Conn)
        System.Threading.Thread.Sleep(1000)
        Dim strtemp As String
        Dim strerr As String
        strtemp = "http://www.bdjobs.com/bgmea/" & strcompanyurl
        Try
            strCompanySource =
Trim(getHMTLSource("http://www.bdjobs.com/bgmea/" & strcompanyurl))
        Catch err As Exception
         Dim strerrquery As String
            strerrquery = "insert into details"
            strerrquery = strerrquery & "(COMPANYNAME)"
            strerrquery = strerrquery & "values ('" &
checkQuotation(strcompanyname) & "')"
            cmdExec.CommandText = strerrquery
            Conn.Open()
            cmdExec.Connection = Conn
            intrecheckid = cmdrecheckidquery.ExecuteScalar
            strrecheckname = cmdrechecknamequery.ExecuteScalar
            If strrecheckname = strcompanyname Then
                If (cmdduplicompany.ExecuteScalar = 0) Then
                    Try
                        cmdExec.ExecuteNonQuery()
                    Catch erro As Exception
                        MsgBox(erro.Message)
                    End Try
                End If
            Else
                Dim cmdCheckExist As OleDb.OleDbCommand = New
OleDb.OleDbCommand("select isnull(max(companyid),0) from details where
companyname='" & checkQuotation(strcompanyname) & "'", Conn)
                If (cmdCheckExist.ExecuteScalar < 1) Then
                    If (cmdduplicompany.ExecuteScalar = 0) Then
                        Try
                            cmdExec.ExecuteNonQuery()
                        Catch erro As Exception
                            MsgBox(erro.Message)
                        End Try
                    End If
                End If
            End If
            Conn.Close()
        End Try
       If InStr(1, strCompanySource, strcompanyname,
CompareMethod.Text) > 1 Then
```

```
            Dim intfirstTD As Integer
            Dim intsecondTD As Integer
            Dim strTD As String
            intfirstTD = InStr(1, strCompanySource, strcompanyname,
CompareMethod.Text)
            intfirstTD = InStr(intfirstTD, strCompanySource, "<tr",
CompareMethod.Text)
            If intfirstTD > 0 Then
                intsecondTD = InStr(intfirstTD, strCompanySource,
"</tr>", CompareMethod.Text)
                strTD = Mid(strCompanySource, intfirstTD, intsecondTD -
intfirstTD)
                straddress = RetriveTDDetails(strTD)
          End If
        End If

        intcompanyfirstidx = InStr(1, strCompanySource, strcompanyname,
CompareMethod.Text)
        While intcompanyfirstidx > 0

            intcompanyfirstidx = InStr(intcompanyfirstidx,
strCompanySource, "<tr", CompareMethod.Text) + 3
            intcompanysecondidx = InStr(intcompanyfirstidx,
strCompanySource, "</tr>", CompareMethod.Text)

            strtablerow = Mid(strCompanySource, intcompanyfirstidx,
intcompanysecondidx - intcompanyfirstidx)

            If InStr(1, strtablerow, "Annul", CompareMethod.Text) > 1
Then

                strannul = retrievemanufactureTD(strtablerow, "Annul")
                strannul = Trim(RetriveTDDetails(strannul))
            End If


            If InStr(1, strtablerow, "Contact", CompareMethod.Text) > 1
Then
                strContactPerson = retrievemanufactureTD(strtablerow,
"Contact")
                strContactPerson =
Trim(RetriveTDDetails(strContactPerson))
             End If

             If InStr(1, strtablerow, "City", CompareMethod.Text) > 1
Then

                strCity = retrievemanufactureTD(strtablerow, "City")
                strCity = Trim(RetriveTDDetails(strCity))
            End If

             If InStr(1, strtablerow, "Designation", CompareMethod.Text)
> 1 Then
                strDesignation = retrievemanufactureTD(strtablerow,
"Designation")
                strDesignation = Trim(RetriveTDDetails(strDesignation))
          End If
```

```
        If InStr(1, strtablerow, "Phone", CompareMethod.Text) > 1 Then

                strPhone = retrievemanufactureTD(strtablerow, "Phone")
                strPhone = Trim(RetriveTDDetails(strPhone))

            End If

            If InStr(1, strtablerow, "Official Web",
CompareMethod.Text) > 1 Then
                strOfficialWeb = retrievemanufactureTD(strtablerow,
"Official Web")
                strOfficialWeb = Trim(RetriveTDDetails(strOfficialWeb))
            End If

            If InStr(1, strtablerow, "E-mail", CompareMethod.Text) > 1
Then
                stremail = retrievemanufactureTD(strtablerow, "E-mail")
                stremail = Trim(RetriveTDDetails(stremail))
                Dim intemailstart As Integer
                Dim intemailend As Integer
                intemailstart = InStr(1, stremail, "<a",
CompareMethod.Text)
                If intemailstart > 0 Then
                    intemailstart = InStr(intemailstart + 2, stremail,
">", CompareMethod.Text)
                    intemailend = InStr(intemailstart + 1, stremail,
"</a>", CompareMethod.Text)
                    stremail = Mid(stremail, intemailstart, intemailend
- intemailstart)
                    stremail = stremail.Replace(">", "")
                End If
            End If

            If InStr(1, strtablerow, "Industry Type",
CompareMethod.Text) > 1 Then
                strIndustryType = retrievemanufactureTD(strtablerow,
"Industry Type")
                strIndustryType =
Trim(RetriveTDDetails(strIndustryType))
            End If

            If InStr(1, strtablerow, "Products", CompareMethod.Text) >
1 Then
                strProduct = retrievemanufactureTD(strtablerow,
"Product")
                strProduct = Trim(RetriveTDDetails(strProduct))
           End If
            If InStr(1, strtablerow, "Factory Location",
CompareMethod.Text) > 1 Then
               strFactoryLocation = retrievemanufactureTD(strtablerow,
"Factory Location")
                strFactoryLocation =
Trim(RetriveTDDetails(strFactoryLocation))
                Exit While
            End If
            intcompanyfirstidx = intcompanysecondidx
        End While
```

```
        Dim strQuery As String
        strQuery = "insert into details"
        strQuery = strQuery &
"(COMPANYNAME,ADDRESS,ANNUL,CONTACTPERSON,CITY,DESIGNATION,PHONE,OFFICI
ALWEB,EMAIL,INDUSTRYTYPE,PRODUCTS,FACTORYLOCATION)"
        strQuery = strQuery & "values ('" &
checkQuotation(strcompanyname) & "','" & checkQuotation(straddress) &
"','" & checkQuotation(strannul) & "','" &
checkQuotation(strContactPerson) & "','" & checkQuotation(strCity) &
"','" & checkQuotation(strDesignation) & "','" &
checkQuotation(strPhone) & "','" & checkQuotation(strOfficialWeb) &
"','" & checkQuotation(stremail) & "','" &
checkQuotation(strIndustryType) & "','" & checkQuotation(strProduct) &
"','" & checkQuotation(strFactoryLocation) & "')"
        cmdExec.CommandText = strQuery
        Conn.Open()
        cmdExec.Connection = Conn
        strrecheckname = cmdrechecknamequery.ExecuteScalar

        If strrecheckname = strcompanyname Then
            If (cmdduplicompany.ExecuteScalar = 0) Then
                Try
                   cmdExec.ExecuteNonQuery()
                Catch err As Exception
                    MsgBox(err.Message)
                End Try
            End If
        Else

            Dim cmdCheckExist As OleDb.OleDbCommand = New
OleDb.OleDbCommand("select isnull(max(companyid),0) from details where
companyname='" & checkQuotation(strcompanyname) & "'", Conn)
           If (cmdCheckExist.ExecuteScalar < 1) Then
               If (cmdduplicompany.ExecuteScalar = 0) Then
                    Try
                        cmdExec.ExecuteNonQuery()
                    Catch err As Exception
                        MsgBox(err.Message)
                    End Try
                End If
            End If
        End If
       Conn.Close()
    End Function


    Public Function retrievemanufactureTD(ByVal strtable As String,
ByVal strsearchname As String)
        Dim intfirstTD As Integer
        Dim intsecondTD As Integer
        Dim strTD As String
        intfirstTD = InStr(1, strtable, strsearchname,
CompareMethod.Text)
        intfirstTD = InStr(intfirstTD, strtable, "<td",
CompareMethod.Text)
```

```
        If intfirstTD > 0 Then
            intsecondTD = InStr(intfirstTD, strtable, "</td>",
CompareMethod.Text)
            strTD = Mid(strtable, intfirstTD, intsecondTD - intfirstTD)
            retrievemanufactureTD = strTD
        End If
    End Function

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim f As New Form2()
        f.Show()

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim df As New datacompare()
        df.Show()

    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        Dim rf As New Resultdata()
        rf.Show()

    End Sub
End Class
```

Snapshot1:

```
Public Class Form2
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim strconn As New ADODB.Connection()
        Dim rst As New ADODB.Recordset()
        Dim Query As String
        strconn.ConnectionString = "Provider=SQLOLEDB.1;Persist
Security Info=False;User ID=sa;Initial Catalog=bdjobs"
        strconn.ConnectionTimeout = 30
        strconn.Open()
        Query = "select * from details"
        GridHeadDisplay()
        rst.Open(Query, strconn, ADODB.CursorTypeEnum.adOpenStatic,
ADODB.LockTypeEnum.adLockReadOnly)
        If Not rst.EOF Then
            While Not rst.EOF
                Grid.Rows = Grid.Rows + 1
                Grid.Row = Grid.Rows - 1 : Grid.Col = 0
```

```
                Grid.Text = IIf(IsDBNull(rst("companyname").Value), "",
rst("companyname").Value)
                Grid.Row = Grid.Rows - 1 : Grid.Col = 1
                Grid.Text = IIf(IsDBNull(rst("city").Value), "",
rst("city").Value)
                rst.MoveNext()


            End While
        End If
        rst.Close()
        strconn.Close()
    End Sub

    Private Sub GridHeadDisplay()
        Grid.Rows = 1
        Grid.Cols = 2
        Grid.set_ColWidth(0, 4000)
        Grid.set_ColWidth(1, 2500)
        Grid.Row = 0 : Grid.Col = 0
        Grid.Text = "Company"
        Grid.CellAlignment = 3

        Grid.Row = 0 : Grid.Col = 1
        Grid.Text = "City"
        Grid.CellAlignment = 3

    End Sub

    Private Sub Grid_Enter(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Grid.Enter

    End Sub
End Class
```

Snapshot2:

```
ublic Class datacompare
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

        Private Sub datacompare_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
         Dim strconn As New ADODB.Connection()
        Dim rst As New ADODB.Recordset()
        Dim Query As String
        strconn.ConnectionString = "Provider=SQLOLEDB.1;Persist
Security Info=False;User ID=sa;Initial Catalog=bdjobs"
        strconn.ConnectionTimeout = 30
        strconn.Open()
        Query = "select * from detailsnext"
        FetchCTRHeadDisplay()
        rst.Open(Query, strconn, ADODB.CursorTypeEnum.adOpenStatic,
ADODB.LockTypeEnum.adLockReadOnly)
        If Not rst.EOF Then
```

```
            While Not rst.EOF
                FetchCTR.Rows = FetchCTR.Rows + 1
                FetchCTR.Row = FetchCTR.Rows - 1 : FetchCTR.Col = 0
                FetchCTR.Text = IIf(IsDBNull(rst("companyname").Value),
"", rst("companyname").Value)
                FetchCTR.Row = FetchCTR.Rows - 1 : FetchCTR.Col = 1
                FetchCTR.Text = IIf(IsDBNull(rst("city").Value), "",
rst("city").Value)
                rst.MoveNext()
            End While
        End If
        rst.Close()
        strconn.Close()
    End Sub

    Private Sub FetchCTRHeadDisplay()
        FetchCTR.Rows = 1
        FetchCTR.Cols = 2
        FetchCTR.set_ColWidth(0, 4000)
        FetchCTR.set_ColWidth(1, 2500)
        FetchCTR.Row = 0 : FetchCTR.Col = 0
        FetchCTR.Text = "Company"
        FetchCTR.CellAlignment = 3
        FetchCTR.Row = 0 : FetchCTR.Col = 1
        FetchCTR.Text = "City"
        FetchCTR.CellAlignment = 3

    End Sub

    Private Sub FetchCTR_Enter(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles FetchCTR.Enter

    End Sub
End Class
```

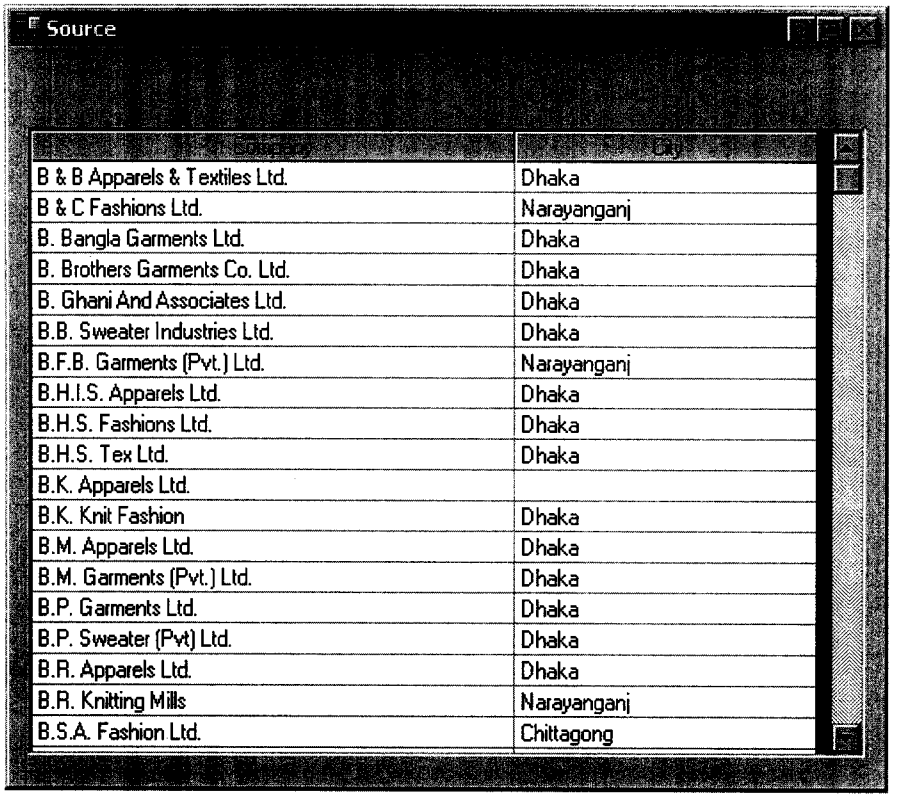
Comparision:

```
Public Class Resultdata
    Inherits System.Windows.Forms.Form

    Private Sub Resultdata_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load

        Dim strconn As New ADODB.Connection()
        Dim rst As New ADODB.Recordset()
        Dim Query As String
        strconn.ConnectionString = "Provider=SQLOLEDB.1;Persist
Security Info=False;User ID=sa;pwd=;Initial Catalog=bdjobs"
        strconn.ConnectionTimeout = 30
        strconn.Open()
        Query = "select dn.* from detailsnext dn where  dn.COMPANYID
not in (select detailsnext.COMPANYID from detailsnext inner join
details d1 on d1.companyname = detailsnext.companyname inner join
details d2 on d2.address like detailsnext.address )"
        FetchresultHeadDisplay()
```

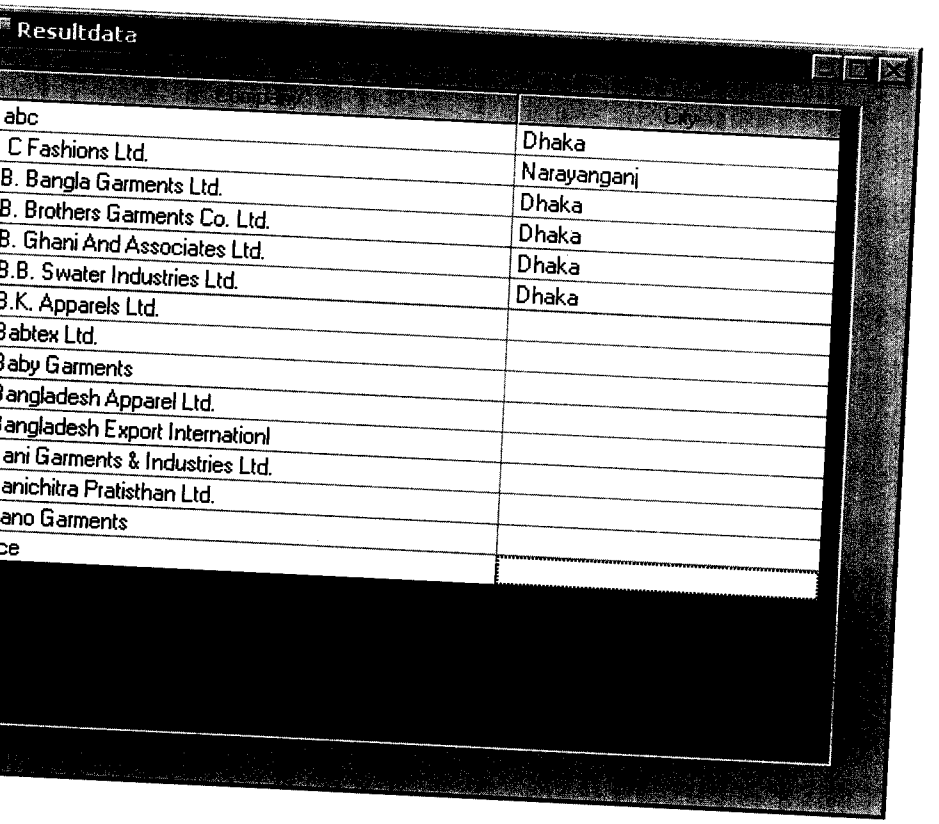
```
        rst.Open(Query, strconn, ADODB.CursorTypeEnum.adOpenStatic,
ADODB.LockTypeEnum.adLockReadOnly)

        If Not rst.EOF Then
            While Not rst.EOF
                Fechresult.Rows = Fetchresult.Rows + 1
                Fetchresult.Row = Fetchresult.Rows - 1 :
Fetchresult.Col = 0
                Fetchresult.Text =
IIf(IsDBNull(rst("companyname").Value), "", rst("companyname").Value)
                Fetchresult.Row = Fetchresult.Rows - 1 :
Fetchresult.Col = 1
                Fetchresult.Text = IIf(IsDBNull(rst("city").Value), "",
rst("city").Value)
                rst.MoveNext()
             End While
        End If
        rst.Close()
        strconn.Close()
    End Sub
   Private Sub FetchresultHeadDisplay()
        Fetchresult.Rows = 1
        Fetchresult.Cols = 2
        Fetchresult.set_ColWidth(0, 4000)
        Fetchresult.set_ColWidth(1, 2500)
        Fetchresult.Row = 0 : Fetchresult.Col = 0
        Fetchresult.Text = "Company"
        Fetchresult.CellAlignment = 3
        Fetchresult.Row = 0 : Fetchresult.Col = 1
        Fetchresult.Text = "City"
        Fetchresult.CellAlignment = 3
    End Sub

 Private Sub Fetchresult_Enter(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Fetchresult.Enter
    End Sub

End Class
```

SAMPLE OUTPUT

Snapshot 1


Source
B & B Apparels & Textiles Ltd. Dhaka
B & C Fashions Ltd. Narayanganj
B. Bangla Garments Ltd. Dhaka
B. Brothers Garments Co. Ltd. Dhaka
B. Ghani And Associates Ltd. Dhaka
B.B. Sweater Industries Ltd. Dhaka
B.F.B. Garments (Pvt.) Ltd. Narayanganj
B.H.I.S. Apparels Ltd. Dhaka
B.H.S. Fashions Ltd. Dhaka
B.H.S. Tex Ltd. Dhaka
B.K. Apparels Ltd.
B.K. Knit Fashion Dhaka
B.M. Apparels Ltd. Dhaka
B.M. Garments (Pvt.) Ltd. Dhaka
B.P. Garments Ltd. Dhaka
B.P. Sweater (Pvt) Ltd. Dhaka
B.R. Apparels Ltd. Dhaka
B.R. Knitting Mills Narayanganj
B.S.A. Fashion Ltd. Chittagong

Result


Resultdata
abc
C Fashions Ltd.
B. Bangla Garments Ltd.
B. Brothers Garments Co. Ltd.
B. Ghani And Associates Ltd.
B.B. Swater Industries Ltd.
B.K. Apparels Ltd.
Babtex Ltd.
Baby Garments
Bangladesh Apparel Ltd.
Bangladesh Export Internationl
Bani Garments & Industries Ltd.
Banichitra Pratisthan Ltd.
Bano Garments
Dhaka
Narayanganj
Dhaka
Dhaka
Dhaka
Dhaka