DB Analyzer

By

Nikkolus Wortham

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Engineering Technology

University of Cincinnati
College of Applied Science

March 2003

DB Analyzer

By

Nikkolus Wortham

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements
for
the Degree of Bachelor of Science
in Information Engineering Technology

_____	_____
Nikkolus Wortham	Date
_____	_____
Prof. Robert Schlemmer, Faculty Advisor	Date
_____	_____
James F. Sullivan, Department Head	Date

Table of Contents

Section **Page**

List of Figures

Figure Number **Page**

Abstract

DB Analyzer is an administration application that is designed to aid database administrators with specific management and planning tasks on an *Oracle* database. Certain tasks like determining if a table space needs to be coalesced and finding percentages of free space in a table may take several hours to perform manually, but by using *DB Analyzer* these tasks can be performed in a matter of seconds.

Built with VB.Net, this application includes three helpful tabs. The first tab contains a clickable graph that displays every free and used space in a selected table space. By viewing the logical location of every free and used space in the database, administrators are able to determine if coalescing or defragmenting is necessary. The second tab includes a list of the administrators most used SQL Scripts that can be saved, edited, and deleted using *DB Analyzer*. The final tab generates several different crystal reports, making it possible to assess the health of the entire database. Overall, this application will allow any database administrator using *Oracle* to perform daily, weekly, or monthly tasks quickly and effectively.

DB Analyzer

1. Statement of Problem

This project was conceived after discussions with Eric Steele, Database Administrator (DBA) at 3M Precision Optics. Mr. Steele explained some of his management and planning tasks performed on the company's database. Since he is the only DBA in the division he has other duties to perform and some of his database tasks can be very time consuming. Tasks like the following can take several hours to perform manually:

- Determining if table spaces need coalescing
- Determining if and which tables are fragmented
- Determining all free and used spaces in a table
- Determining what percentage of a table is used
- Determining how many extents a table is spread across

Mr. Steel had an idea for a GUI application that would aid him in completing these tasks in a very timely manner.

2. Description of Solution

Since a great deal of company's focus on time management for employees, DB Analyzer had to be a quick and easy to use application that allows Mr. Steele to complete some of his DBA duties and start on another project. After several meetings with Mr. Steele, we decided that the application should include the following:

- A graph section
- A logon screen able to access any Oracle database

- A clear and easy to read graph
- Accurate and easy to print reports

2.1 User Profile

DB Analyzer was designed for any Database Administrator. The DBA is responsible for procuring/maintaining database software and related documents and tools, optimizing performance, monitoring backups and maintaining the overall health of the database. Administrators should have a Bachelor's Degree in computer science, information science, system analysis, software engineering, or a closely related field and three years experience as a database administrator, specialist or database designer and programmer.

2.2 Design Protocols

Visual Studio.Net was the integrated development environment (IDE) used to write all VB.Net code. Although Visual Basic 6 is the current coding language at 3M Precision Optics, I chose VB.Net because it is the lasted version of Visual Basics. With features including a drag-and-drop form designer, automatic docking/anchoring, and a "code-behind" form's model, Visual Studio.Net was a great tool to use on this project.

Microsoft's .Net Provider for Oracle was used to connect to the Oracle 8i and 9i databases used for this project. Although the OLE DB data classes are more then adequate to use, I chose the .Net Oracle provider because of the enhanced database access mechanism. With the performance boost and the similarity to the SQL data provider, the .Net Oracle Provider was found to be a good choice for this project.

All reports created in the Report tab were written in Crystal Reports for Visual Studio.Net. Crystal Reports has the ability to do the following:

- Export reports to four different file types
- Zoom reports
- Search text
- Print

Although Crystal Reports for Visual Studio.Net was not the easiest tool to use, it was a good utility to create the reports needed by the application.

The images used on the login screen were created using POV-Ray 3.5. POV-Ray is a free tool used to create three-dimensional, photo-realistic images using a rendering technique called ray-tracing. POV reads a text file containing lighting, camera, and object information then generates the scene accordingly. The DB Analyzer logo and graph legend on the main tab was created in Adobe Photoshop 7.0.

3. Deliverables

During Senior Design II, I specified the following deliverables for this project:

- XML configuration file able to hold SQL scripts and other criteria needed for DB Analyzer to function.
- Secure log-on user interface able to access any Oracle database that the user has permissions too.
- Three alert indicators that will alert user if selected table is a certain percentage full, if a table is nearing maximum number of extents, and if a selected table needs to be defragmented.
- Graph tab that consists of a legend and a clickable graph that displays table, table space and file id information.
- Script tab that consists of SQL scripts to perform defragmenting and coalescing functions.

- Report tab that will display several types of reports to print and review.

4. Design and Development

The next sections describe the project's budget, timeline, and hardware/software specifications.

4.1 Budget

The budget for this project is shown below (See Figure 1).

Software	
Item	**Cost**
Oracle 8i Personal Ed. **	$ 800.00
Visual Studio.Net ***	$ 1,079.00
Adobe Photoshop 7	$ 649.00
POV- Ray	Free
Hardware	
Item	**Cost**
Dimension 2350 – Intel Pentium 4 processor at 2.0GHz *	$ 699.00
PROJECT TOTAL	**$ 3,227.00**

Figure 1. Budget

4.2 Timeline

Below are the achievements of my Senior Design and Co-op sequences.

4.2.1 Senior Design I Achievements

During Senior Design I, I accomplished the following:

- Researched possible project
- Drafted proposal
- Finalized and presented project proposal

4.2.2 Spring Co-op Achievements

During my Spring Co-op quarter the following was accomplished:

- Developed GUI design and layout
- Designed application graphics
- Started research on basic GDI+ functions
- Started collections and classes for the Main Tab of the application

4.2.3 Senior Design II Achievements

During Senior Design II, I accomplished the following:

- Started collections and classes for the Script Tab of the application
- Drafted design freeze documentation
- Completed prototype
- Finalized and presented design freeze

4.2.4 Fall Co-op Achievements

During my Fall Co-op quarter the following was accomplished:

- Developed XML configuration file editor
- Integrated configuration file with Script Tab of main application

4.2.5 Senior Design III Achievements

During Senior Design II, I accomplished the following:

- Started coding for the Report Tab of the application
- Drafted final report documentation
- Finalized coding and testing
- Finalized and presented final documentation

5. Proof of Design

In this section, I will convey how the deliverables of the project were met and what challenges I encountered.

5.1 Secure Login

When the application is started, user will see the login screen (See Figure 2). Users must establish the username, password and server name of the Oracle database they wish to access. DB Analyzer queries Oracle to make sure the entered user has access to the desired database. If any of the entered criteria does not match Oracle standards, DB analyzer will display an error message conveying that to the user (See Figure 3).

Figure 2. Login Screen

Figure 3. Invalid Error Message

5.2 Graph Tab

Once authenticated, users will have the option of selecting three different tabs. The first tab is the Graph tab where users will have to press the "Go" button. The "Go" button retrieves all table space names in the entire database. Once these names have been retrieved, users must select a table space name (See Figure 4).

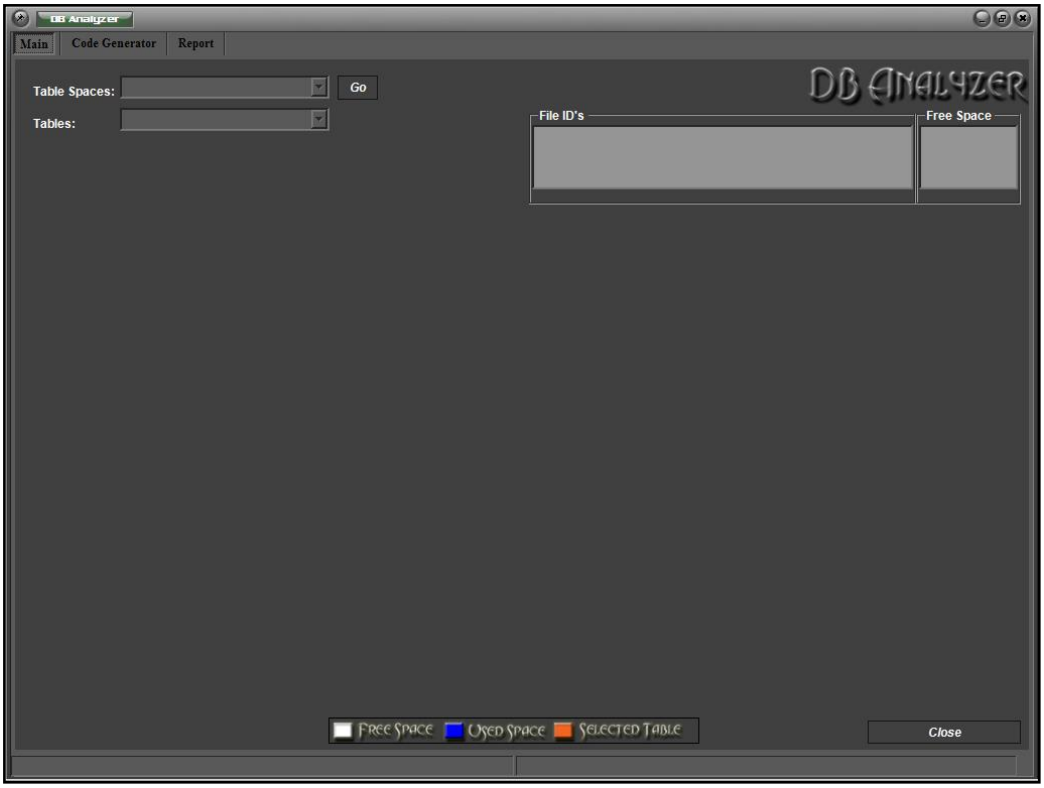


Figure 4. Graph Tab

After users have selected the preceding, a graph will appear on the screen that maps the logical locations of every free and used space in the selected table space (See Figure 5). Once the graph appears, users will be able to assess the following about the selected table space:

- The five largest free spaces
- The size of the table and the number of extents it is spread across

- The percentage of free space in the table space

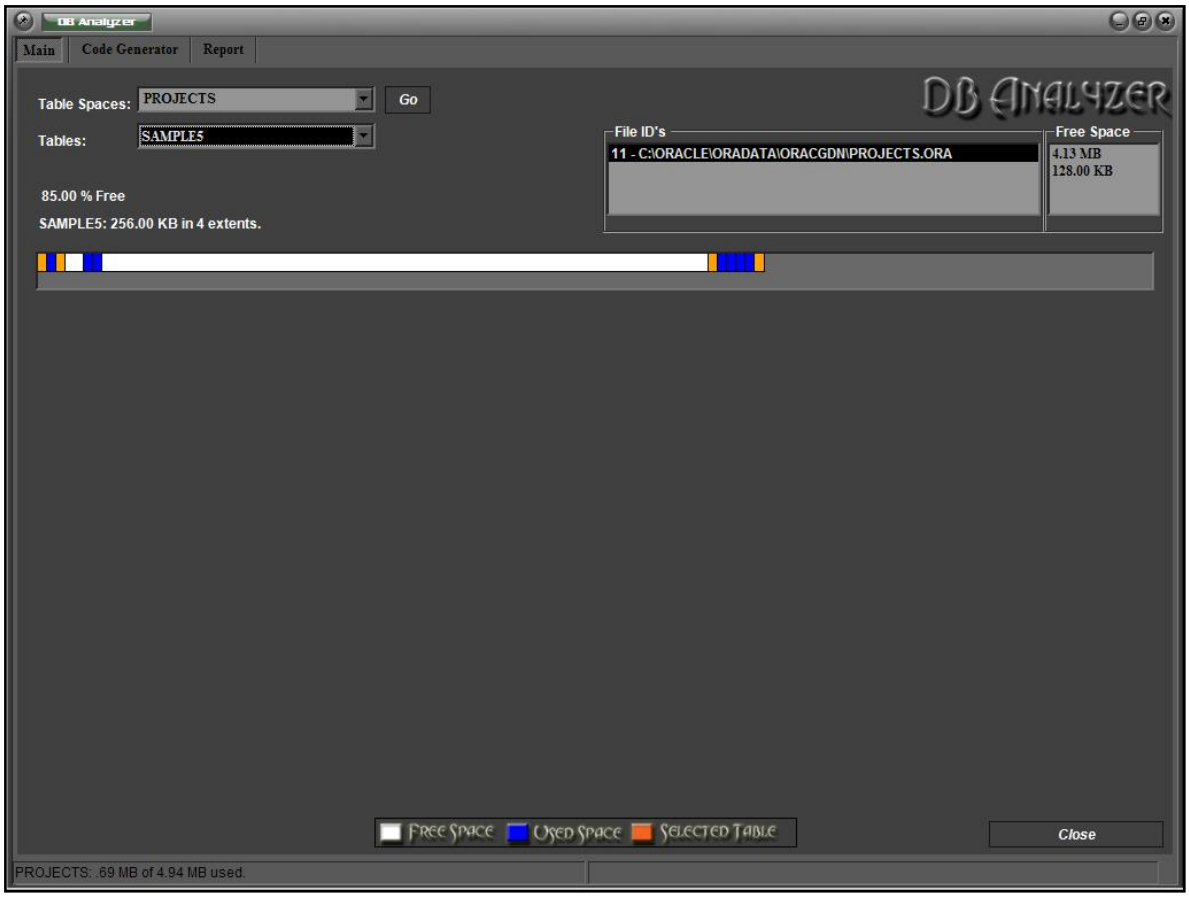


Figure 5. Graph

Users can also click on any space of the graph and find out the name and the size of the space. Every space on the graph is designated by one of three colors. White for free space, blue for used space, and orange for the selected table.

5.3 Script Tab and Editor

The Script tab is the next tab in the application (See Figure 6). This tab was originally intended to allow users to select from several predefined scripts to copy and paste into Oracle SQL Plus. After starting the code for this tab I decided to give the user more control over what scripts DB Analyzer would display. Since this was a new addition to the project, I had to create an editor to allow users make changes (See

Figure 7). The Configuration Editor is a separate application that will allow user to add, delete, and update the scripts to be displayed in DB Analyzer.

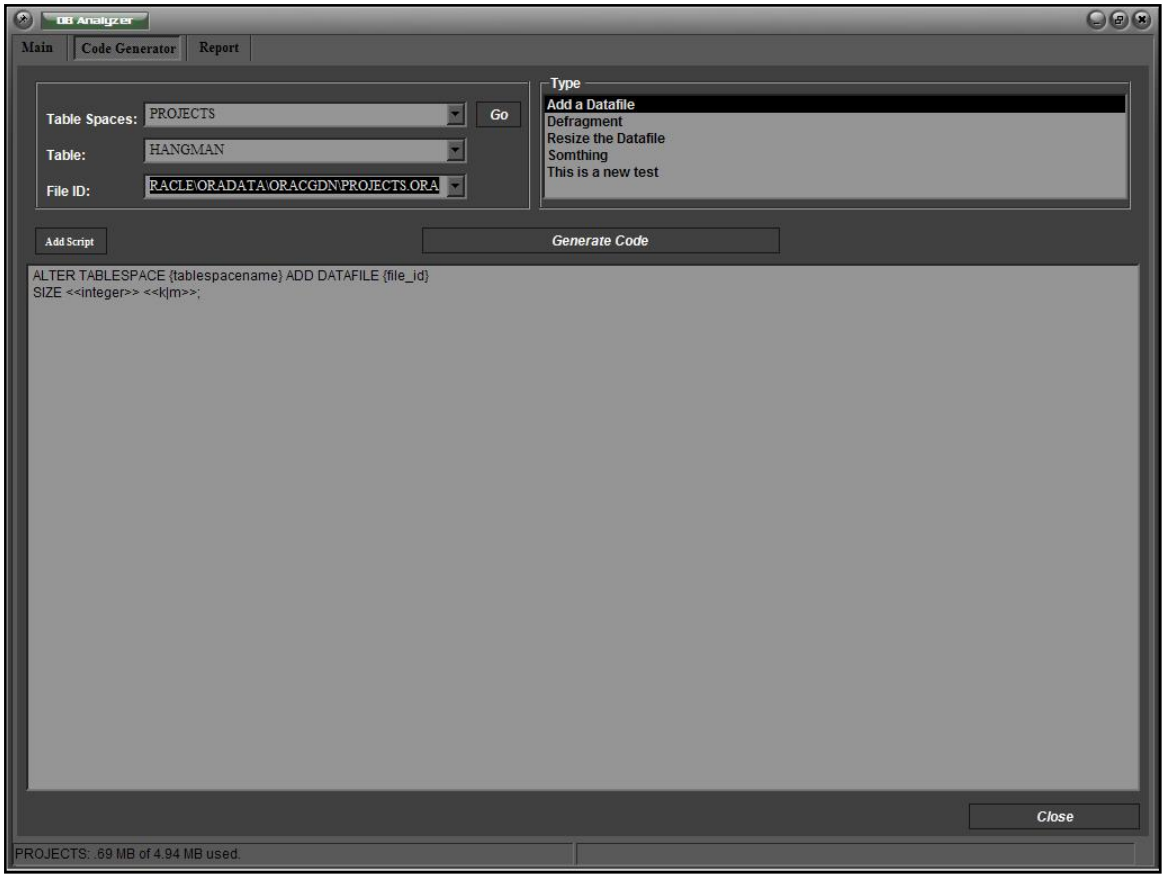


Figure 6. Script Tab

The editor reads and writes to the XML file used by DB Analyzer. Once a user has added their favorite scripts using the Configuration Editor, they can refresh the Script tab screen in DB Analyzer by hitting the F5 key.

The Script tab recognizes the following key words:

- {FILE_ID}
- {TABLESPACENAME}
- {TABLE}

If any of these keys are used in the selected scripts, DB Analyzer will replace it with the appropriate name.

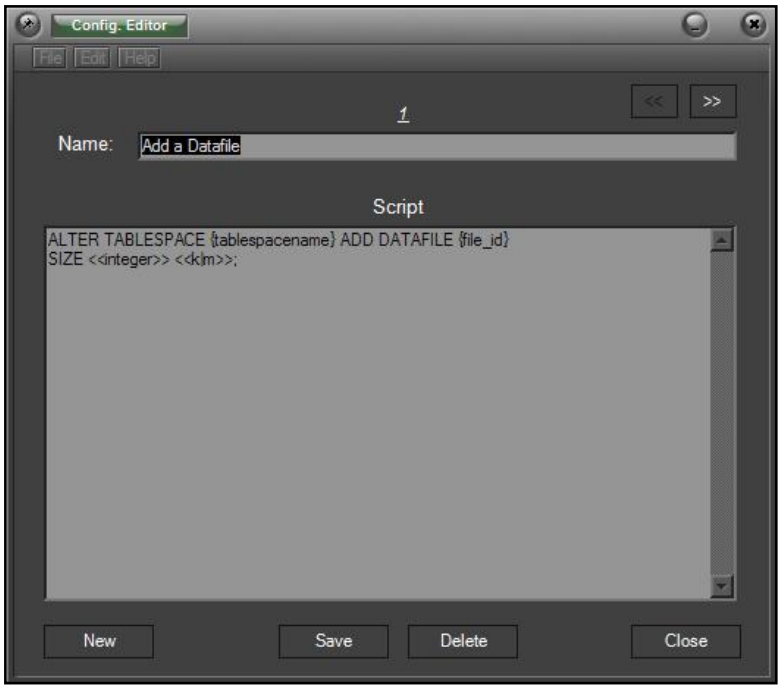


Figure 7. Configuration Editor

5.4 Report Tab

The Report tab is the last of three that users have the option of choosing (See Figure 8). Users are able to create two different types of reports, table space and full. A table space report informs the user of the following:

- Number of file ids in the table space
- File ID locations
- Every table in the table space
- Number of blocks in a table
- File ID in which a table belongs

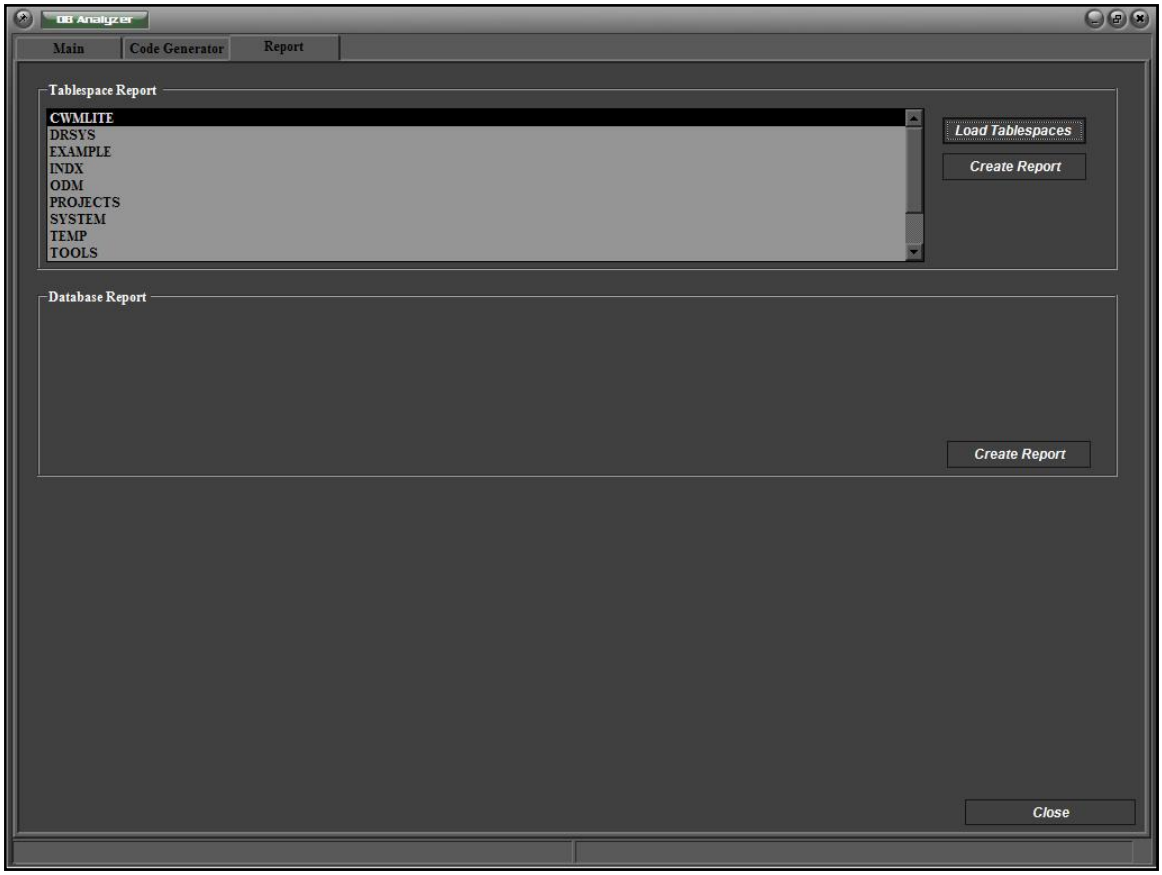


Figure 8. Report Tab

A full database report informs the user of the following:

- All table spaces within the database
- Amount of free space per table space
- All data file id's and locations per table space

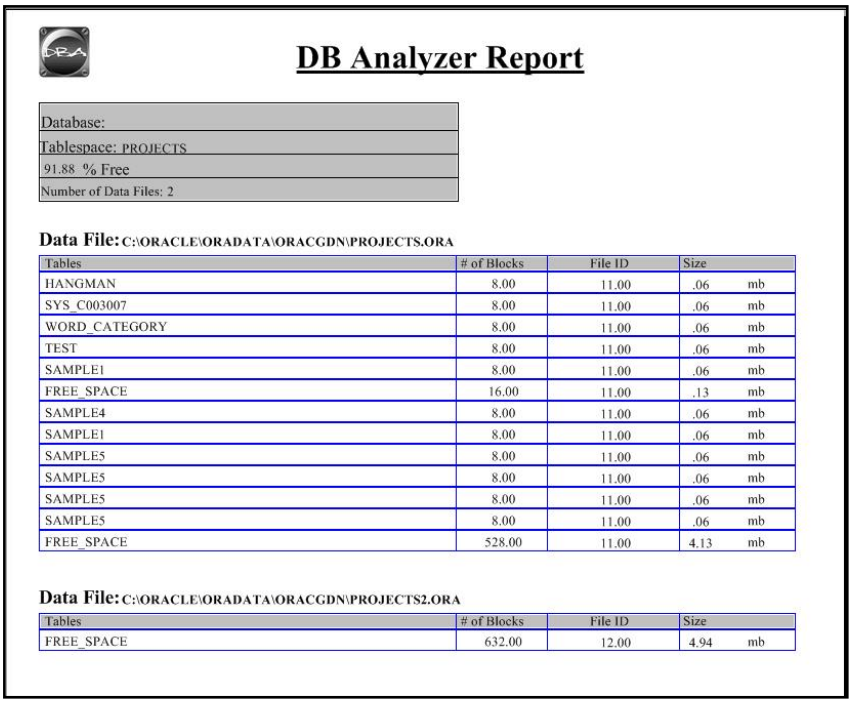
6. Conclusion

This application was created to aid 3M Precision Optics database administrator with his daily management and planning tasks. DB Analyzer was written in Visual Basic.Net and utilizes the Microsoft's .Net Provider for Oracle to connect to an Oracle 8 and 9i database. Quick and easy to use, DB Analyzer is a great tool for any DBA. This project as fulfilled all items scoped in Senior Design II,

Design Freeze Deliverables. Testing was performed and DBA is currently being used by the intended user.

Appendix A.
Sample Reports

A 1. Table Space Report Sample



DB Analyzer Report

Database:
Tablespace: PROJECTS
91.88 % Free
Number of Data Files: 2

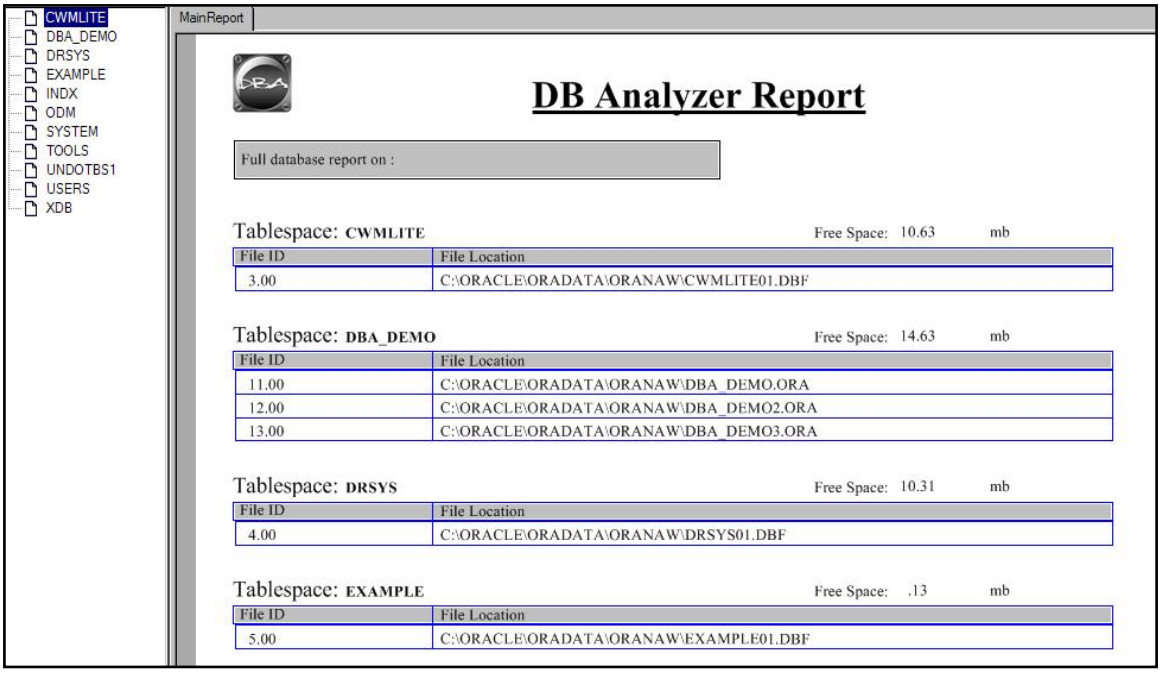
Data File: C:\ORACLE\ORADATA\ORACGDN\PROJECTS.ORA

Tables	# of Blocks	File ID	Size
HANGMAN	8.00	11.00	.06 mb
SYS_C003007	8.00	11.00	.06 mb
WORD_CATEGORY	8.00	11.00	.06 mb
TEST	8.00	11.00	.06 mb
SAMPLE1	8.00	11.00	.06 mb
FREE_SPACE	16.00	11.00	.13 mb
SAMPLE4	8.00	11.00	.06 mb
SAMPLE1	8.00	11.00	.06 mb
SAMPLE5	8.00	11.00	.06 mb
SAMPLE5	8.00	11.00	.06 mb
SAMPLE5	8.00	11.00	.06 mb
SAMPLE5	8.00	11.00	.06 mb
FREE_SPACE	528.00	11.00	4.13 mb

Data File: C:\ORACLE\ORADATA\ORACGDN\PROJECTS2.ORA

Tables	# of Blocks	File ID	Size
FREE_SPACE	632.00	12.00	4.94 mb

A 2. Full Report Sample



CWMLITE
DBA_DEMO
DRSYS
EXAMPLE
INDX
ODM
SYSTEM
TOOLS
UNDOTBS1
USERS
XDB

MainReport

DB Analyzer Report

Full database report on :

Tablespace: **CWMLITE** Free Space: 10.63 mb

File ID	File Location
3.00	C:\ORACLE\ORADATA\ORANAW\CWMLITE01.DBF

Tablespace: **DBA_DEMO** Free Space: 14.63 mb

File ID	File Location
11.00	C:\ORACLE\ORADATA\ORANAW\DBA_DEMO.ORA
12.00	C:\ORACLE\ORADATA\ORANAW\DBA_DEMO2.ORA
13.00	C:\ORACLE\ORADATA\ORANAW\DBA_DEMO3.ORA

Tablespace: **DRSYS** Free Space: 10.31 mb

File ID	File Location
4.00	C:\ORACLE\ORADATA\ORANAW\DRSYS01.DBF

Tablespace: **EXAMPLE** Free Space: .13 mb

File ID	File Location
5.00	C:\ORACLE\ORADATA\ORANAW\EXAMPLE01.DBF

Appendix B.
Interesting Code Samples

B 1. Login Animation Code Sample

In-order to animate the image on the login screen, I used a timer control to cycle through the images I needed to display. In the page load event, I started the timer that changes the images.

```
Dim i As Integer

Private Sub Login_Load()
    /…/
    Timer1.Start()
    i = 1
End Sub
```

Twenty images were used in the animation. Every tick of the timer increments the counter by one, changing the name of the picture box image source.

```
Private Sub Timer1_Tick() Handles Timer1.Tick
    If i = 20 Then
        i = 1
        pb1.Image = Image.FromFile("../images/DBAnimation0" + i.ToString + ".bmp")
    ElseIf i > 9 Then
        pb1.Image = Image.FromFile("../images/DBAnimation" + i.ToString + ".bmp")
    Else
        pb1.Image = Image.FromFile("../images/DBAnimation0" + i.ToString + ".bmp")
    End If
        i = i + 1
End Sub
```

B 2. Clickable Graph Code Sample

Making it possible for DB Analyzer to recognize what space a user has clicked on was a very interesting and challenging piece of code to write. Since one space on the graph can span several rows, I had to make sure that multiple locations could be stored for one space. I accomplish this by capturing all values in a DictionaryBase collection.

```
Private Sub pbGraph_Click() Handles pbGraph.Click
    Dim dEntry As DictionaryEntry
    Dim pPos As New Point(MousePosition.X, MousePosition.Y)
    Dim i, j As Integer
    Dim arrFileID() As String
    Try
        'colTableSpaces
        For Each dEntry In colBlocks
            i = dEntry.Value.Blocks.Length
            'FileID
            arrFileID = Split(lbFileId.Items(lbFileId.SelectedIndex), " - ", 0)
            Select Case i
                Case 1  ' If space only spans one row
                    If dEntry.Value.FileID = arrFileID(0) Then
                        If pbGraph.PointToClient(pPos).X > dEntry.Value.Blocks(0).Left And
                            pbGraph.PointToClient(pPos).X < dEntry.Value.Blocks(0).right Then
                            If pbGraph.PointToClient(pPos).Y > dEntry.Value.Blocks(0).Top And
                                pbGraph.PointToClient(pPos).Y < dEntry.Value.Blocks(0).Bottom
                                Then
                                StatBar.Panels.Item(1).Text = dEntry.Value.Segment & " : " &
                                colTableSpaces.CalculateTableSize(dEntry.Value.Segment,
                                dEntry.Key)
                                Exit Sub
                            End If
                        End If
                    End If
                Case Else  ' If space spans more then one row
                    For j = 0 To i - 1
                        If dEntry.Value.FileID = arrFileID(0) Then
                            If pbGraph.PointToClient(pPos).Y > dEntry.Value.Blocks(j).Top And
                            pbGraph.PointToClient(pPos).Y < dEntry.Value.Blocks(j).bottom Then
                                If pbGraph.PointToClient(pPos).X > dEntry.Value.Blocks(j).Left And
                                pbGraph.PointToClient(pPos).X < dEntry.Value.Blocks(j).right Then
                                    StatBar.Panels.Item(1).Text = dEntry.Value.Segment & " : " &
                                    colTableSpaces.CalculateTableSize(dEntry.Value.Segment,
                                    dEntry.Key)
                                    Exit Sub
                                End If
                            End If
                        End If
                    Next j
            End Select
        Next dEntry
    /.../
End Sub
```