

## **ABSTRACT**

In this thesis we develop a new Enterprise Manager Tool with transformation data to Excel sheet capability. This is a user friendly tool. This tool gives user to access databases and database objects such as tables, views, stored procedures, they can manage databases where ever they are. With transfer to excel capability the datas can be seen more efficiently. To create this tool .NET technology and MSSQL Server is used.

## ÖZET

Enterprise Manager Tool ,kullanıcıların databaslerine ve databaselerin objelerine (table,stored procedure,view) erişebilmeleri ve bunları yönetebilmeleri için yapılmış içerisinde excel'e data transfer özelliği bulunan,windows ortamında yapılmış bir tooldur.Excel'e data transferi özelliği sayesinde kullanıcılar verilerini rahatlıkla Excel ortamında kullanabilirler.Tool'un geliştirilmesine Teknoloji olarak .NET Server olarakta MSSQL Server kullanılmıştır

## **ACKNOWLEDGEMENT**

I want to thank to my adviser Prof. Dr.Selahattin Kuru .He helps me during the time that I have prepared this thesis

## TABLE OF CONTENTS

<b>Abstract.....</b>	<b>i</b>
<b>Özet.....</b>	<b>ii</b>
<b>Acknowledgement.....</b>	<b>iii</b>
<b>Table of Contents.....</b>	<b>iv</b>
<b>List of Figures.....</b>	<b>vi</b>
<b>List of Tables.....</b>	<b>vii</b>
<b>1. INTRODUCTION.....</b>	<b>1</b>
<b>2. OVERVIEW OF DATABASE MANAGEMENT SYSTEM AND ENTERPRISE MANAGER TOOLS.....</b>	<b>2</b>
2.1. Enterprise Manager Tool.....	2
2.2. Database Management Sytem.....	2
2.2.1 Kinds Of Database Management System.....	4
2.3. Functionalities Of A Typical Enterprise Manager Tool.....	4
2.4. Examples Of Enterprise Manager Tools.....	5
2.4.1. MyLittle Admin Enterprise Manager.....	6
2.4.2. MySQL Enterprise Manager Tool.....	6
2.4.3. Oracle Enterprise Manager Tool.....	7
2.4.4. Database Web Explorer.....	8
<b>3. TOOLS AND TECHNOLOGIES USED.....</b>	<b>10</b>
3.1. .NET FRAMEWORK.....	10
3.2. ADO.NET.....	11
3.3. OVERVIEW OF C#.....	12
3.4. .NET INTERPROPERABILITY.....	14
3.5. MICROSOFT SQL SERVER.....	16

3.5.1 Relational Data Model.....	16
3.5.2 Database Administration.....	17
3.5.3 DBMS Architecture.....	18
<b>4. THE PROPOSED ENTERPRISE MANAGER TOOL.....</b>	<b>21</b>
4.1. Functionalities Of The Proposed Tool.....	21
4.2. Module Design Of The Proposed Tool.....	22
<b>5. EXAMPLE USE OF PROPOSED TOOL.....</b>	<b>29</b>
5.1 Description And Realisation Of the Example.....	29
5.2 Use Of The Enterprise Manager Tool on the Example.....	30
<b>6. CONCLUSSION.....</b>	<b>40</b>
<b>7. REFERENCESES.....</b>	<b>41</b>
<b>8.APPENDIX.....</b>	<b>42</b>

## LIST OF FIGURES

Figure 2.1	Client Server DBMS.....	3
Figure 2.2	My Little Admin.....	6
Figure 2.3	My SQL Enterprise Manager .....	7
Figure 2.4	Web Explorer.....	9
Figure 3.1	.NET Framework.....	11
Figure 3.2	ADO.NET Architecture.....	12
Figure 3.3	Event Handler.....	16.
Figure 3.4	SQL SERVER Architecture.....	20
Figure 4.1	Create Database.....	23
Figure 4.2	Drop Object.....	23
Figure 4.3	Classes.....	24
Figure 4.4	Treeview Control.....	26
Figure 4.5	Enterprise Manager TreeNode Design.....	26
Figure 4.6	Menu Items.....	26
Figure 4.7	Popup Menu.....	28
Figure 5.1	Registration Page.....	31
Figure 5.2	Enterprise Manager Tool.....	32
Figure 5.3	Table Properties.....	32
Figure 5.4	New Database.....	33
Figure 5.5	Database Backup.....	33
Figure 5.6	Restore Database.....	34
Figure 5.7	Open-Save .SQL Files.....	34
Figure 5.8	Database Popup Menus.....	35
Figure 5.9	Transfer Data ToExcel.....	36
Figure 5.10	Data in Excel Sheet.....	36
Figure 5.11	View Popup Menu.....	37
Figure 5.12	Current Activity.....	38
Figure 5.13	Quary Analayzer.....	39

## LIST OF TABLES

Table 5.1	Authors.....	29
Table 5.2	Titles.....	30
Table 5.3	Titleauthors.....	30

## **1. INTRODUCTION**

Enterprise Manager is an administrative tool that manages databases its objects and also it manages Database Management System. Database Management System is a software, that allows a user to store, update and retrieve data in abstract terms and this make it easy to maintain and retrieve information from a database. A DBMS relieves the user from having to know about exact physical representations of data and having to specify detailed algorithms for storing, updating and retrieving data.

In this thesis we develop and present a new Enterprise Manager Tool. This tool has additional features compared to a typical Enterprise Manager Tool. This feature involved a spreadsheet functionalities such as creating views, manipulating tables.

Enterprise Manager allows users:

- Register servers.
- Create databases.
- Create tables, views, stored procedures.
- Transfer data to Excel sheets.
- Design and test SQL statements.

This thesis is composed of 6 chapters Chapter 1 is Introduction, Chapter2 is Overview of Database Management System and Enterprise Manager, Chapter3 Tools And Technologies Used, Chapter 4 The Proposed Enterprise Manager Tool, Chapter 5, Example Use of Proposed Tool, Chapter 6 Conclusion and future Recommendation



## **2 OVERVIEW OF DATABASE MANAGEMENT SYSTEM AND ENTERPRISE MANAGER TOOLS**

In this chapter we overview enterprise manager tool and database management system(DBMS).

### **2.1 ENTERPRISE MANAGER TOOL**

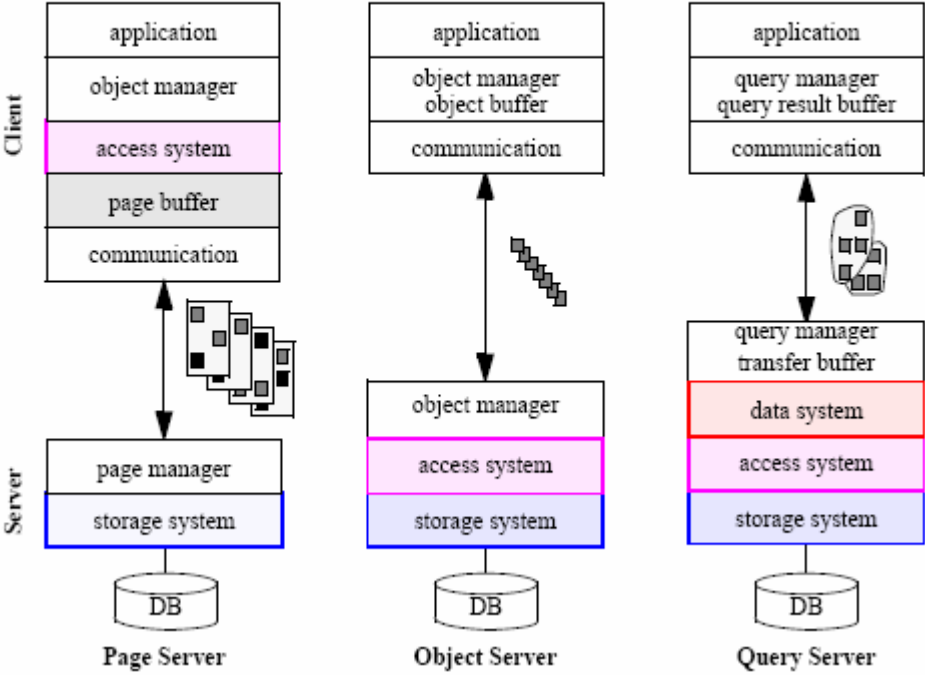
Enterprise Manager is an administrative tool that manages databases and its objects, and also manages Database Management System. Enterprise Manager Tools use programming APIs, which allow users to build robust database applications. Enterprise Manager Tools gives user lots of functionalities. Each server in the enterprise manager has two types of databases: system databases and user databases. System databases store information about SQL Server as a whole, this database operate and manage the system. User databases are databases that users create. The database is a collection of data items organized as a set of formally-described tables from which data can be accessed or reassembled in many different ways without having to reorganize the database tables. Database objects help user structure data and define data integrity mechanism. Database objects are tables, views, storedprocedures, functions. Table is one of the database object, it defines a collection of rows that have associated columns. There are two types of tables tables and user tables. User tables are the tables that the user create. The information about data in system tables includes configuration information and definitions of all of the databases and database objects. Users should not directly modify system tables.

Views is just a relation, but we store a definition, rather than a set of tuples. Views can be dropped using the drop view command. Views can be used to present necessary information, it provides a way to look at data from one or more tables or views in a database. Functions can return either a scalar value or a table. Functions are used to encapsulate frequently performed logic incorporated in a function can call the function rather than having to repeat all of the function logic. StoredProcedure is a named collection of precompiled Transact SQL statements that execute together.

**2.2 DATABASE MANAGEMENT SYSTEM (DBMS)**

To be able to carry out operations like insertion, deletion and retrieval, the database needs to be managed by a substantial package of software. This software is usually called a Database Management System (DBMS). The primary purpose of a DBMS is to allow a user to store, update and retrieve data in abstract terms and thus make it easy to maintain and retrieve information from a database. A DBMS relieves the user from having to know about exact physical representations of data and having to specify detailed algorithms for storing, updating and retrieving data.

The DBMS can maintain the integrity of the database by not allowing more than one user to update the same record at the same time. The DBMS can keep duplicate records out of the database; for example, no two customers with the same customer numbers (key fields) can be entered into the database.



**Figure 2.1 Client Server DBMS**

If the DBMS provides a way to interactively enter and update the database, as well as interrogate it, this capability allows for managing personal databases. However, it may not leave an audit trail of actions or provide the kinds of controls necessary in a multi-user

organisation. These controls are only available when a set of application programs are customised for each data entry and updating function.

### **2.2.1 KINDS OF DATABASE MANAGEMENT SYSTEM(DBMS)**

**File systems** - File systems contain the bulk of information. There are billions of application-specific file-formats and thousands of standardized file formats. File systems are relatively light weight and provide varying services (storing and naming persistent opaque large objects (blobs), sometimes security, versioning, and replication). First and later Web architectures depend on access to HTML usually stored in files.

**Information Retrieval Systems** - After years of playing a back seat role to DBMSs, information retrieval in the form of web crawlers has become a basic ingredient in browsing the Web. These systems generate indexes of the information stored on the internet. We cover these separately in the internet tools survey on searching and indexing.

**Relational DBMS** - Relational DBMS systems are mature in the market (but only since the late 1980s). Dominant vendors are Oracle, Sybase, Informix, IBM, and Microsoft but there are many more (not listed).

**Object DBMS** -Systems are the perpetual runners up. Object Database Management Group(ODMG) is a consortium of vendors of many of the object-oriented DBMS systems. ODMG-93 is their standard specification. Voting members of ODMG are GemStoneSystems (formerly Servio), IBEX Computing, O2 Technology, Object Design, Objectivity, POET Software, UniSQL, and Versant Object Technology.

**Object-Relational** - Object-Relational is a term for systems that support both relational (table-based) SQL queries and object data types. At this point, most ODBMSs offer a relational query capability and some RDBMS' provide some ADT capabilities so differences are in the details.

**Componentware DBMS** - OMG CORBAServices provide services for data modeling, persistence, transactions, queries, and several other DBMS functions and provide an alternative middleware approach to exporting DBMS function. So far, there are few vendors of CORBAServices and so it is early to say whether DBMS functionality will appear in componentware form though some promising experiments demonstrate this approach (e.g.the

TI Open OODB system) and some interested products are on the horizon (Microsoft OLE/DB).

### **2.3 FUNCTIONALITIES OF TYPICAL ENTERPRISE MANAGER TOOLS**

Enterprise Manager tools manages databases so it has lots of functionalities .

The major capability of enterprise managers is to *create databases* and its object. The database is a collection of data items organized as a set of formally-described tables from which data can be accessed or reassembled in many different ways without having to reorganize the database tables. There are several database objects that are common to many Database Management Systems (DBMS).

These objects are tables, views, storedprocedures, systemfunctions.

#### ***Tables:***

The concept of tables is quite simple. Series of rows and columns, like a spread sheet. The columns represent the grouping of the data, and the rows represent the unique sets of data. The "cell," or the single intersection of a row and column, contains a single datum.

#### ***Views:***

Views are all about limiting or joining data. Views are simply the result of a select statement. The statements that make up a view are stored in text right inside the database on a server.

#### ***Indexes:***

indexes are used to speed up access to data. If your database is used in an application that performs many inserts, edits, and deletes, then speedy access isn't always the driving factor. An index is used to access data quickly.

#### ***Stored Procedures:***

- Stored procedures are similar to views. They are just text objects, and don't store data. They do provide access to data. They can return datasets, just like a view, but that's where the similarity ends.
- Stored procedure goes farther than a simple select statement. Stored procedures can have full-blown code, meaning that they can have loops, conditions, and other program logic.
- Stored procedure also have a serious trump-card over a view – they can accept inputs and return outputs. The output they produce can be a recordset, and they can also return a computed value, the result of a calculation on values returned from another storedprocedure, and more.

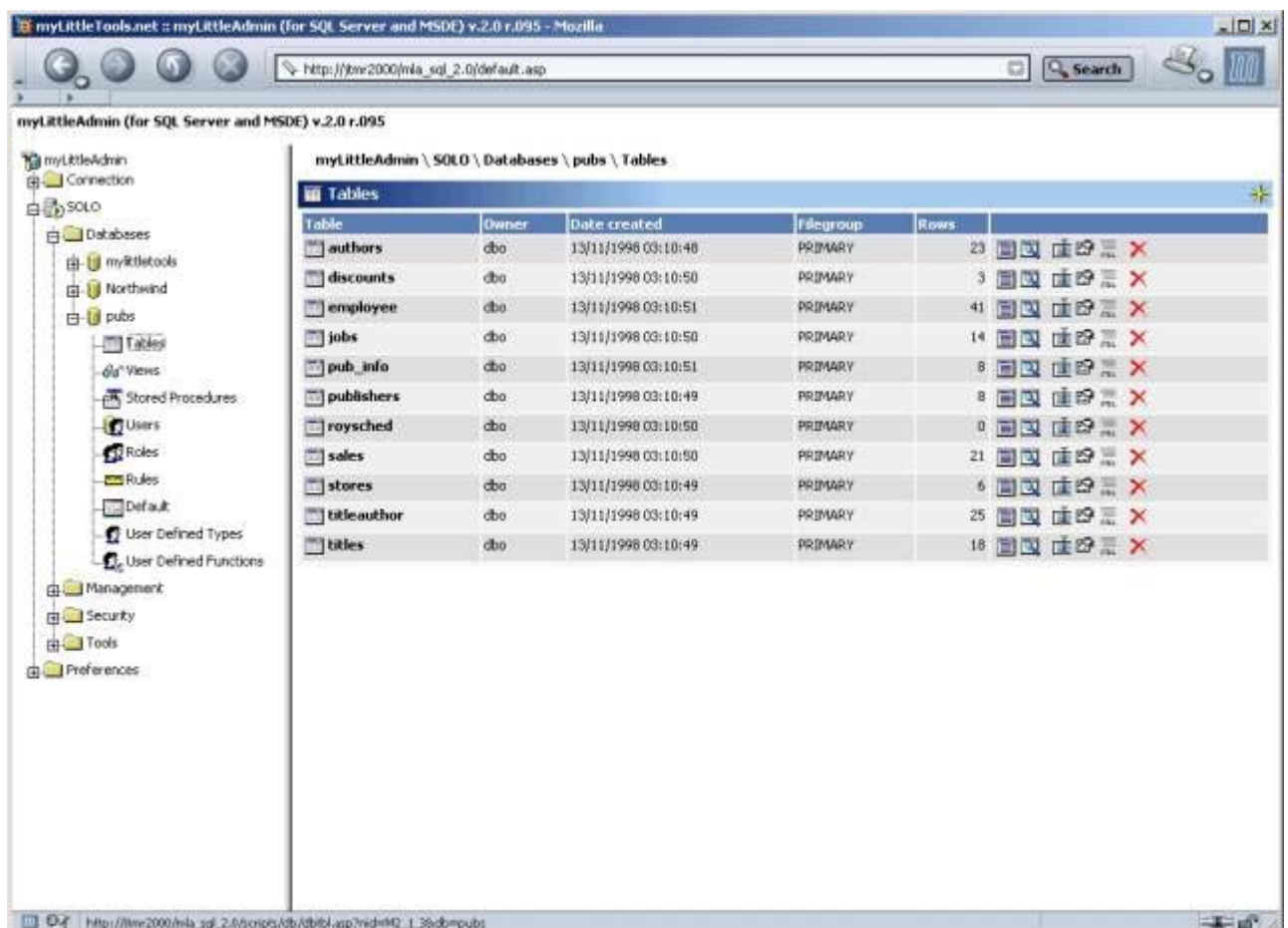
## 2.4 EXAMPLES OF ENTERPRISE MANGER TOOLS

There are lots of Enterprise Manager Tools that the people use, some of the tools are web based tools some of them are windows based tools. Each tool created with different Technologies. Some of these tools are MyLittle Admin[Figure 2.1], My SQL Enterprise Manager .[Figure 2.2], Oracle Enterprise Manager[Figure 2.3], Database Web Explorer [Figure 2.4].Most of these tools used Microsoft SQL Server. In this chapter we wil review these tools.

### 2.4.1 MY LITTLE ADMIN

MyLittleAdmin is a web-based SQL Server and MSDE Database administration tool. It is intended to handle the administration of SQL Server and MSDE databases over the www. It allows user to fully manage their databases even when they don't want or cannot use Microsoft Enterprise Manager.

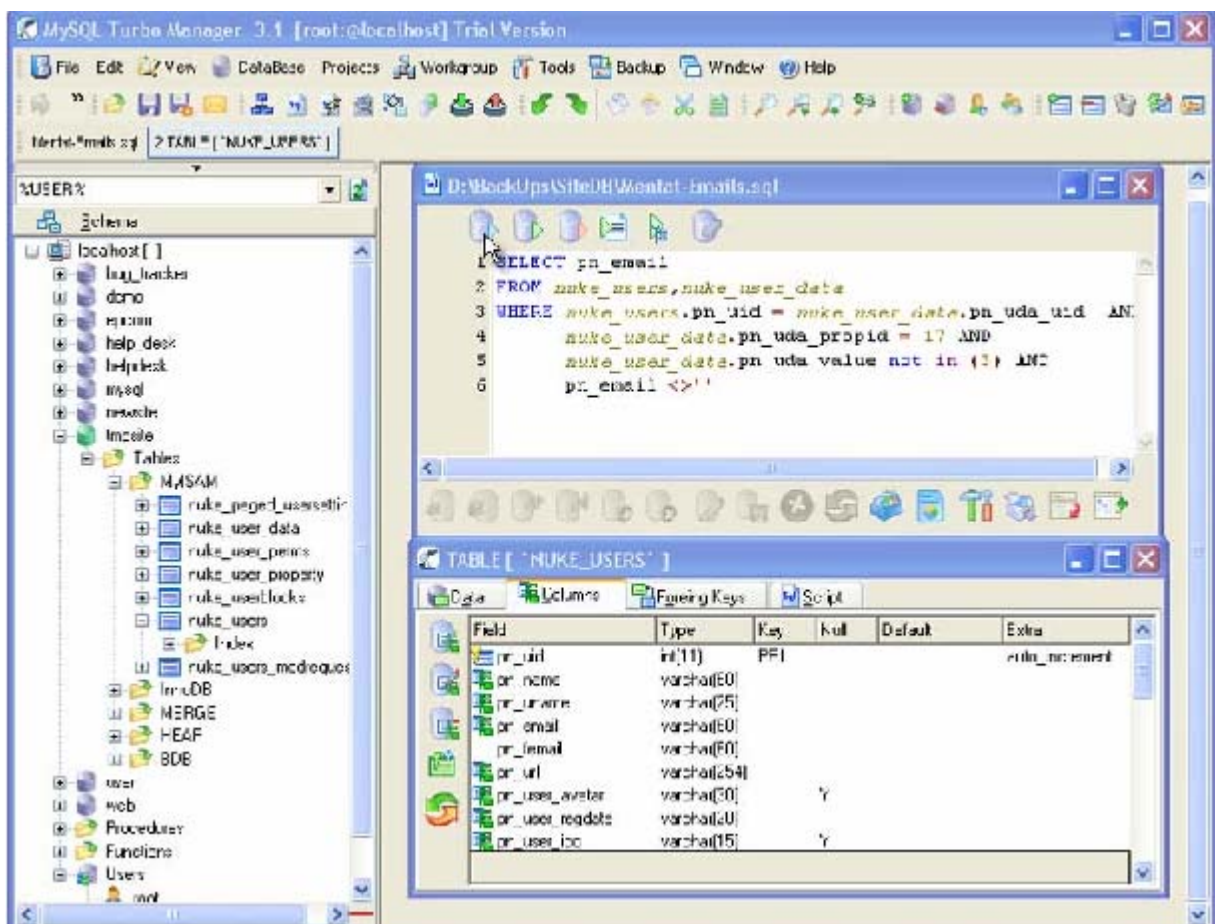
With myLittleAdmin, users will do through a browser almost everything they did before with Enterprise Manager. myLittleAdmin is the best solution to remotely administer SQL Server and MSDE[11]. myLitteAdmin also includes several tools.[Figure 2.1]



**Figure 2.2 MyLittleAdmin**

### 2.4.2 MY SQL ENTERPRISE MANAGER

EMS MySQL Manager is a powerful tool for MySQL® Database Server administration and development. MySQL Manager works with any MySQL versions from 3.23 to 5.02 and supports all of the latest MySQL features including views, stored procedures and functions, InnoDB foreign keys and so on [12]. It offers plenty of powerful tools for experienced users to satisfy all their needs. MySQL Manager has a new state-of-the-art graphical user interface with well-described wizard system, so clear in use that even a newbie will not be confused with it.



**Figure 2.3 My SQL Enterprise Manager**

### 2.4.3 ORACLE ENTERPRISE MANAGER

Oracle Enterprise Manager enables database administrators (DBAs) to manage databases distributed across a network from a single centralized Console. Database management is performed through an easy-to-use graphical user interface (GUI). From the Console, you can do following operations.

- Administer, diagnose, and tune multiple databases distributed across a network.
- Distribute software to multiple servers and clients.
- Accommodate growing distributed environments by easily scaling upwards to maintain performance and automate routine tasks.
- Automate remote task execution by scheduling jobs on multiple databases simultaneously or at varying time intervals.
- Schedule and run tasks such as remote database backups or report execution at off hours, providing the kind of "lights out" management vital in a large, distributed environment.
- Remotely monitor critical database and system events that you define, have detected events represented graphically on the Console (or be notified through electronic mail or paging), and specify for corrective actions to be performed in response to an event.
- Run integrated third-party applications and tools.
- Accommodate both large database configurations and environments with numerous databases.
- Administer multiple databases simultaneously in separate windows.
- Localize task execution, so that tasks are completed even during network downtime. Messages are queued locally and saved until they can be delivered, even if a network connection is down.
- Allow DBAs single logon access to particular databases by storing their privileges in a credentials file used to manage connections. When connecting to a database from the Console, credentials are transparently passed, so you do not have to log in repeatedly.

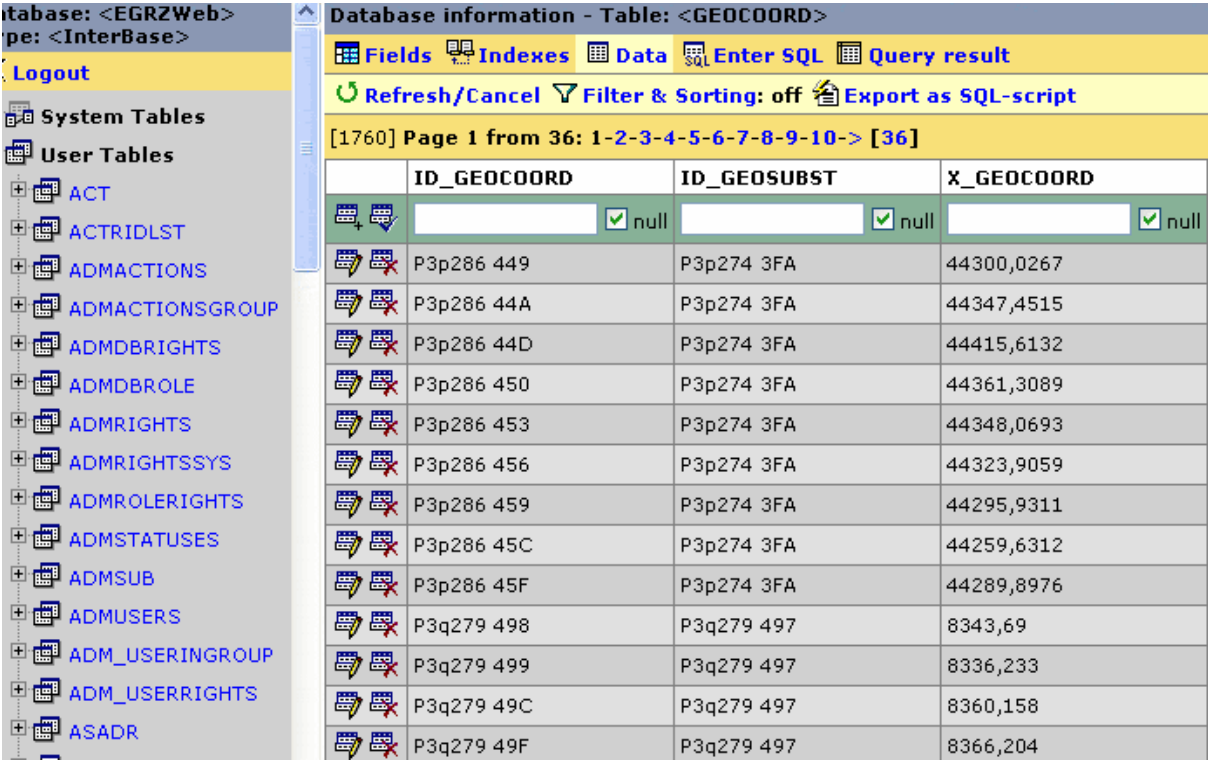
#### **2.4.4 DATABASE WEB EXPLORER**

Database Web Explorer is an universal Web-based database manager intended to work with DBMS in Internet/Intranet networks. Database Web Explorer allows you to manipulate the data stored in databases via Internet. As program of the remote access the usual Web-browser is used. Database Web Explorer was developed with ASP technology in pure script language and does not required installation of any additional COM-components, DLL or ActiveX neither on the Web-server nor on the computer of the user. Database Web Explorer

is distributed in source codes - HTML and JavaScript. Supports connections to databases through ODBC drivers and OLE DB providers. Configuration and installation of Database Web Explorer is very easy. The product works under the Web-server Microsoft Internet Information Server[Figure2.4].

*Common features:*

- Very easy installation, configuration and usage
- Supports the multiuser work with DBMS
- Does not required installation of any additional COM-components, DLL or ActiveX neither on the Web-server nor on the computer of the user
- As programm of the remote access the usual Web-browser is used
- Supports connections to databases through ODBC drivers and OLE DB providers
- Compact code in pure script language



**Figure 2.4 Web Explorer**



### **3 TOOLS AND TECHNOLOGIES USED**

To develop this tool .NET Technologies and MICROSOFT SQL SERVER is used in this chapter we review both of them

#### **3.1. NET FRAMEWORK**

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET Framework is designed to fulfill the following objectives:

To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.

To provide a code-execution environment that minimizes software deployment and versioning conflicts[10].

To provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party.

To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.

To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.

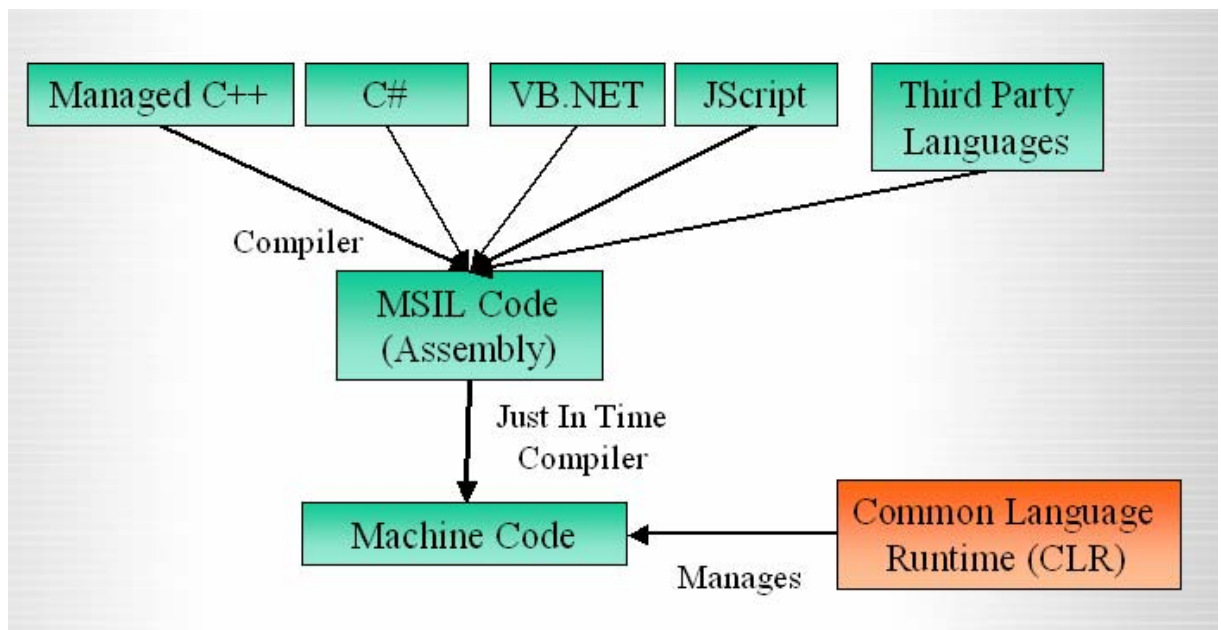
To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library[10].

*The common language runtime* is the foundation of the .NET Framework [Figure 3.1]. The runtime is as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that ensure security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the

runtime is known as managed code, while code that does not target the runtime is known as unmanaged code.

The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.



**Figure 3.1 .NET Framework**

The .NET Framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and unmanaged features. The .NET Framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts.

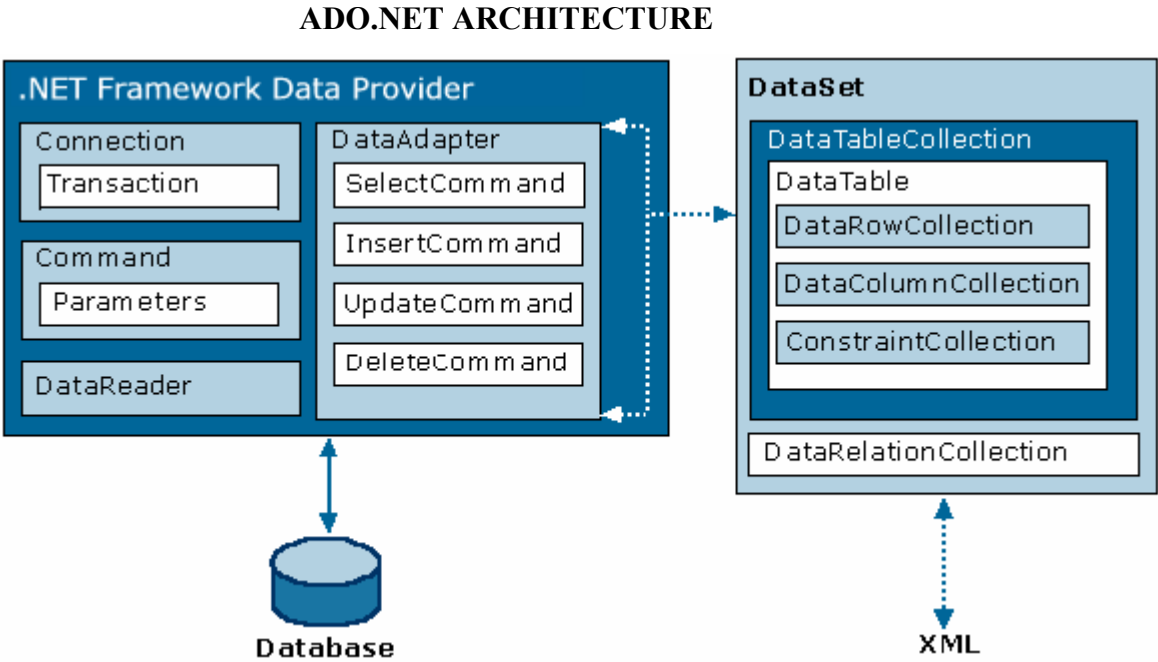
### **3.2 ADO.NET**

ADO.NET provides consistent access to data sources such as Microsoft SQL Server, as well as data sources exposed through OLE DB and XML. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, manipulate, and update data. ADO.NET cleanly factors data access from data manipulation into discrete components that can be used separately or in tandem.

ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, or placed in an ADO.NET **DataSet** object in order to be exposed to the user in an ad-hoc manner, combined with data from multiple sources, or remoted between tiers[20].

The ADO.NET **DataSet** object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML [Figure3.2].

The ADO.NET classes are found in System.Data.dll, and are integrated with the XML classes found in System.Xml.dll. When compiling code that uses the **System.Data** namespace, reference both System.Data.dll and System.Xml.dll.[9]



**Figure 3.2 ADO.NET Architecture**

**3.3 OVERVIEW OF C#**

*Classes*

Classes are types, but are far more powerful than the simple types like int and float. Not only can you customize your data storage using classes, but you can also add methods to classes. That kind of compartmentalization—where data and methods are rolled up into a single class—is the entire reason that OOP was introduced in the first place. It enables the programmers to deal with larger programs. The process of wrapping related data and methods

into a class (and so preventing them from cluttering up the rest of the program) to create a single entity is called *encapsulation* [15].

Classes enable you to develop applications using object-oriented programming (OOP) techniques. Classes are templates that define objects. When you create a new form in a C# project, you are actually creating a class that defines a form; forms instantiated at runtime are derived from the class. Using objects derived from predefined classes, such as a C# Form class, is just the start of enjoying the benefits of object-oriented programming—to truly realize the benefits of OOP, you must create your own classes.

The philosophy of programming with classes is considerably different from that of "traditional" programming. Proper class-programming techniques can make your programs better, both in structure and in reliability. Class programming forces you to consider the logistics of your code and data more thoroughly, causing you to create more reusable and extensible object-based code[15].

Controls used on your forms have events associated with them you can respond to with event handlers in your application. C# makes it easy to have the same behavior in other parts of your application by creating your own delegates and events, and raising the events based on your program logic.

There are three pieces that are related and must be included to make our custom events work. They are a delegate, an event, and one or more event handlers. In this article we will examine delegates and events, how they relate, and how to use them to create custom event handling for your applications[9]. The code to implement the delegates and events is also shown. Details about the code are included in comments.

### ***Delegates***

A delegate is a class that can contain a reference to an event handler function that matches the delegate signature. It provides the object oriented and type-safe functionality of a function pointer. The .NET runtime environment implements the delegate; all you need to do is declare one with the desired signature. It is customary for delegates to have two arguments, an object type named sender and an EventArgs type named e[8].

When you double click on a control in the forms designer of Visual Studio.NET, you are provided with a stub for an event handler that matches this signature. This signature is not required and you may want a different signature for your custom events. When a delegate is created and added to an event invocation list, the event handler is called when the event is raised. Multiple delegates can be added to a single event invocation list and will be called in the order they were added.

## *Events*

Events are notifications, or messages, from one part of an application to another that something interesting has happened. When an event is raised, all the delegates in the invocation list are invoked in the order they were added. Each delegate contains a reference to an event handler, and each event handler executed. The sender of the event does not know which part of the application will handle the event, or even if it will be handled [8].

It just sends the notification and is finished with its responsibilities. It is often necessary to provide some information about the event when the notification is sent. This information is normally included in the EventArgs argument to the event handler. You will probably want to develop a class derived from EventArgs to send information for custom events.

Suppose you are creating an application for a bank to manage accounts and want to raise an event when a transaction would cause the balance falls below some minimum balance. There is no button to click when this happens, and you would not want to depend on a person noticing the balance is low and performing some action to indicate the balance is low to the rest of the application.

A low balance happens because of a withdrawal of some amount that reduces the account balance below the minimum required balance. You want the notification to be automatic, so appropriate action can be taken by some other part of the application without user intervention. This is where custom delegates and events are useful.

## *Constructors and Destructors*

As you open your new class file, you'll notice that C# added the public class declaration and a method called `clsMyClass()`. This is known as the *class constructor*. A constructor has the same name as the class, includes no return type, and has no return value. A class constructor is called whenever a class object is instantiated. Therefore, it's normally used for initialization if some code needs to be executed automatically when a class is instantiated. If a constructor isn't specified in your class definition, the Common Language Runtime (CLR) will provide a default constructor.

## **3.4 NET INTERPPROPERABILITY:COM INTEROP**

NET is a new programming platform representing the future of Windows programming.

Developers are moving across to it and learning the new .NET oriented languages and frameworks, but new systems do not appear overnight. It is a lengthy process moving entire applications across to a new platform and Microsoft is very much aware of this.

To this end, .NET supports a number of interoperability mechanisms that allow applications to be moved across from the Win32 platform to .NET piece by piece, allowing developers to still build complete applications, but which comprise of Win32 portions and some .NET portions, of varying amounts.

When building new .NET applications, there are provisions for using existing Win32 DLL exports (both custom DLL routines and standard Win32 API routines) as well as COM objects (which then act like any other .NET object).

When building Win32 applications there is a process that allows you to access individual routines in .NET assemblies. When building Win32 COM client applications, there is a mechanism that lets you use .NET objects as if they were normal COM objects.

Because of the different data types available on the two platforms (such as PChar in Win32 and the new Unicode String type on .NET), inter-platform calls will inevitably require some form of *marshaling* process to transform parameters and return values between the data types at either end of the call. Fortunately, as we shall see, the marshaling is done for us after an initial process to set up the inter-platform calls.

In a COM Interop system, there must be some form of reconciliation between the COM reference counting model and the .NET garbage collection model. Again, after the initial setup step, this is all taken care of for the developer by wrapper objects manufactured by the .NET support tools.

#### .NET Clients Using Win32 COM Server Objects (RCW)

COM object, wrapper objects called Runtime-Callable Wrappers (RCW objects) need to be generated. These wrapper objects cater for the difference in lifetime management between .NET and COM. RCW objects are .NET objects that manage the reference count of a COM object as well as dealing with the marshaling of parameters and return types for the COM object methods.

RCW objects are manufactured at runtime by the CLR using information found in an *Interop Assembly* (an assembly containing definitions of COM types that can be used from managed code). You use a type library importer to scan the COM server type library and generate appropriate .NET-compatible information in an Interop Assembly for your COM server.

The type library importer can be invoked from a utility, Tlbimp.exe, that is supplied with the .NET Framework SDK. This can be also done under program control using the TypeLibConverter class in the System.Runtime.InteropServices namespace, although use of the utility program is much more common.

#### *Hooking COM Server Events*

Setting up event handlers for COM object events is much the same as for any other object. The only things to know are the types involved. Button Click event handler first. The delegate type of this event is EventHandler, defined as [Figure 3.3]

```
type  
EventHandler = procedure (Sender: TObject; Args: EventArgs) of object;
```

**Figure 3.3 Event Handler**

## **3.5 MICROSOFT SQL SERVER**

### **3.5.1 RELATIONAL DATA MODEL**

Microsoft SQL Server lets you create and maintain user-defined data types that are based on the built-in data types. By creating a user-defined data type and binding a number of integrity rules and a default value to it, you can simulate relational domains. You can use these domains wherever a standard data type can be used.

The database owner can define views on base tables or other views. The view definition can also be encrypted to protect the definition text. Microsoft SQL Server supports the *with check option* feature, which ensures that updated values still comply with the view's original *where* clause [14].

You have no control over the physical storage structure of the database tables and indexes, but you can specify tuning information such as fill factors and default behavior if duplicate data is encountered. One index per table can be a clustered index; this index physically sorts the records.

Microsoft SQL Server also supports stored procedures. In SQL Server, a stored procedure can return multiple row data sets to an application. Stored procedures on one server can invoke remote stored procedures on another server, if the required permissions have been granted on both servers.

There are few restrictions on the type of SQL statements a stored procedure can execute. Stored procedures can start a transaction, create a table, manipulate the table's records, and then commit the transaction. They are very powerful in environments in which the application is a "thin" software layer that simply displays data and interacts with the user. Stored

procedures can then be used to process each transaction on the server. In this way, all manipulation statements can be encapsulated within the database itself.

Stored procedures are usually defined using Transact-SQL statements but can also be specified as a function in an external dynamic link library (DLL)[14]. You can therefore add a lot of functionality to the server by adding customized functions defined in a DLL. These procedures are called extended stored procedures.

SQL Server supports three triggers for each table: one each for “Insert”, “Update”, and “Delete” operations. Triggers execute after each statement that changes data. The changed data is available through two virtual tables called *Inserted* and *Deleted*. Before SQL Server adopted the check constraints defined in the SQL-92 standard, these triggers were the only method available to let the server implement integrity checking.

One of Microsoft SQL Server's powerful new features is its ability to manipulate server-side cursors within stored procedures and triggers. A trigger or stored procedure can define, open, and traverse a whole cursor without sending the resulting rows back to the client. This capability enables a trigger to process each of the changed rows in turn, thus simulating row-at-a-time validation.

### **3.5.2 DATABASE ADMINISTRATION**

DBAs can write their own management applications, and sophisticated applications can incorporate management information in the application itself. For example, before running a very intensive process, an application may check the amount of connected sessions to warn the user of possible effects on other users. Alternatively, an application may interrogate the locking information to see which user is blocking its access to a specific resource.

SQL Server has Trace utility that can tell a DBA exactly which SQL statements are being executed by any particular user. You can define filters to restrict the amount of information displayed. The Enterprise Manager lets system administrators use the Database Maintenance Plan Wizard to enable all of the required routine maintenance tasks on the database. Also new in SQL Server is the ability to define fallback servers that take over from primary servers in case of hardware failure.

When configuring SQL Server, you can choose to use standard security, NT integrated security, or mixed security. Standard security means that each user must be defined within SQL Server, each with a separate SQL Server password; thus a user may have two different sets of usernames and passwords, one for NT and one for SQL Server[7]. With integrated security, the system manager can map the NT usernames directly to a SQL Server login. A



single sign-on provides access to both Windows NT and SQL Server. Integrated security can be easier for end users and system managers because fewer usernames and passwords must be created, maintained, and remembered. Mixed security lets the user choose whether to use a separate SQL Server login [7].

The SQL Server executive service provides a powerful scheduling mechanism for almost any job the system manager might want to run on a regular basis. It includes built-in jobs such as replication tasks and Transact-SQL commands; user can also specify operating system commands, which opens up just about anything user can think of.

### **3.5.3 DBMS ARCHITECTURE**

A commonly used views of data approach is the three-level architecture suggested by ANSI/SPARC (American National Standards Institute/Standards Planning and Requirements Committee). The reports proposed an architectural framework for databases. Under this approach, a database is considered as containing data about an enterprise. The three levels of the architecture are three different views of the data:

- External - individual user view
- Conceptual - community user view
- Internal - physical or storage view

The three level database architecture[Figure 3.4] allows a clear separation of the information meaning (conceptual view) from the external data representation and from the physical data structure layout. A database system that is able to separate the three different views of data is likely to be flexible and adaptable. [7]

The external level is the view that the individual user of the database has. This view is often a restricted view of the database and the same database may provide a number of different views for different classes of users. In general, the end users and even the applications programmers are only interested in a subset of the database.

The conceptual view is the information model of the enterprise and contains the view of the whole enterprise without any concern for the physical implementation. This view is normally more stable than the other two views. In a database, it may be desirable to change the internal view to improve performance while there has been no change in the conceptual view of the database. The conceptual view is the overall community view of the database and it includes all the information that is going to be represented in the database. The conceptual view is defined by the conceptual schema which includes definitions of each of the various types of

data. The internal view is the view about the actual physical storage of data. It tells what data is stored in the database and how.

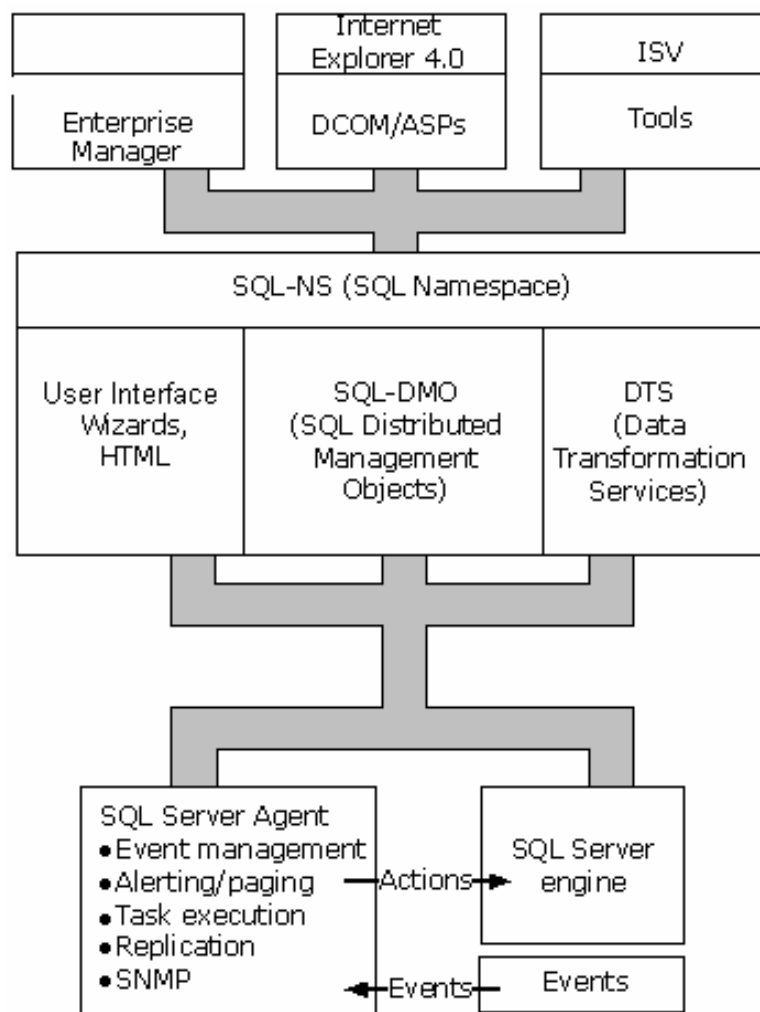
Efficiency considerations are the most important at this level and the data structures are chosen to provide an efficient database. The internal view does not deal with the physical devices directly. Instead it views a physical device as a collection of physical pages and allocates space in terms of logical pages.

The separation of the conceptual view from the internal view enables us to provide a logical description of the database without the need to specify physical structures. This is often called physical data independence. Separating the external views from the conceptual view enables us to change the conceptual view without affecting the external views. This separation is sometimes called logical data independence.

Assuming the three level view of the database, a number of mappings are needed to enable the users working with one of the external views. For example, the payroll office may have an external view of the database that consists of the following information only:

- Staff number, name and address.
- Staff tax information e.g. number of dependents.
- Staff bank information where salary is deposited.
- Staff employment status, salary level, leave information etc.

The conceptual view of the database may contain academic staff, general staff, casual staff etc. A mapping will need to be created where all the staff in the different categories are combined into one category for the payroll office. The conceptual view would include information about each staff's position, the date employment started, full-time or part-time.. This will need to be mapped to the salary level for the salary office. Also, if there is some change in the conceptual view, the external view can stay the same if the mapping is changed.



**Figure 3.4 Sql Server Architecture**

## **4 THE PROPOSED ENTERPRISE MANAGER TOOL**

Enterprise Manager is a web based tool implemented in ASP.NET that enables you to easily manage your SQL Server data wherever you are. This tool have lots of functionalities such the other existing Enterprise Manager Tools and most important functionality transfer data to Excel spread sheets.

### **4.1 FUNCTIONALITIES OF THE PROPOSED TOOL**

Enterprise Manager tool helps to manage SQL Server data, and allows the user to administrate his/her database. Users can browse through database servers, databases and tables and can retrieve the data they want. With the Query Analyzer, users can select which databases, tables and columns they want in the query, and they can see the results in the tables.

This tool has lots of functionalities.

The first functionality is creating a *database* and its objects:

- Create Database.
- Drop Database.
- Backup Database.
- Restore Database.
- View names of tables, fields in database.
- View properties of tables such as primary keys data types of fields.
- View records in the tables.
- Add, delete records in tables The records in the tables can be modified.
- Create views see the datas inside the views.
- Create stored procedure.
- Transfer data to *Excel spread sheets* functionality.
- Transfer to data to Excel is the important capability user can transfer data from the table to the Excel sheets, and they can analyze the data in spread sheets.

#### *Quary Analyzer functionality:*

User can see the data in the table with exploring database object, also they can write their own queries on the quary analyzer pad and they can achieve the data that they want to see. User can create tables, stored procedures, views, they can add into the table or delete data from table.

#### *Security functionality:*

Microsoft SQL Server uses two accounts to get to data. The first is a SQL Serve logon, the is the database login, these MSSQL accounts are usually the same name, and can also be windows accounts or MSSQL accounts. It's quite simple to use this object to set up both the server and database accounts.

## **4.2 MODULE DESIGN OF PROPOSED TOOL**

In my projecret I used COM Objects:

- Component Object Model (COM) objects are basically black boxes that can be used by applications to perform one or more tasks. They are most commonly implemented as a dynamic-link library (DLL). Like a conventional DLL, COM objects expose methods that your application can call to perform any of the supported tasks.
- A COM object's public methods are grouped into one or more interfaces. To use a method, you must create the object and obtain the appropriate interface from the object. An interface typically contains a related set of methods that provide access to a particular feature of the object.
- You must use COM-specific techniques to control the lifetime of the object.
- COM objects do not need to be explicitly loaded. COM objects are typically contained in a DLL. However, you do not need to explicitly load the DLL or link to a static library in order to use a COM object. Each COM object has a unique registered identifier that is used to create the object. COM automatically loads the correct DLL.
- COM is a binary specification. COM objects can be written in and accessed from a variety of languages.

#### ***Create Database:***

SQL DMO helps to create database and its objects Tables, Stored Procedures, Views, Functions [Figure 4.1]. While created databases all objects are copied to the new database. Each SQLServer object has its own Database collection. Within each database are database objects, such as tables, views, and stored procedures. The Tables collection in a SQL Server database have columns, and keys. A Column object enables to manipulate the properties of a single column in a SQL Server table. With the Column object add and remove individual columns in a table.

```
CreateDatabase newDatabase = new CreateDatabase();
newDatabase.DbName = DbName;
newDatabase.DbFileName =FileName;
newDatabase.DbPhysicalName = DataFile;
newDatabase.DbOnPrimary = DataOnPrimary;
newDatabase.DbSize = DataSizeInitial;
```

**Figure 4.1 Create Database**

#### ***Drop Objects:***

The first thing to drop objects is looking at the type of the objects, if it's a table, a database, stored procedure or it's a view. To drop objects sql "drop" command is used. To drop tables, views and stored procedures the first thing is to determine the database of the objects and then used the "drop" command [Figure 4.2]. If the object type is a database it must be drop from Master database

```
Use Northwind drop table "deneme"
```

**Figure 4.2 Drop Object**

#### ***Scripts Of The Objects:***

SQL.DMO helps to get the scripts of the objects. The create script of the objects is `SqlDmo.storedprocedure.Script(SQLDMO_SCRIPT_TYPE.SQLDMOScript_Default`

#### ***Get Data From Microsoft SQL Server:***

While working with SQL Server some of the classes are used [Figure 4.3]. These classes `SqlConnection` class, `SqlCommand` class, `SqlDataAdapter` class, `SqlDataReader` class.

- *SqlConnection Class:* The `SqlConnection` class represents a connection to SQL Server data source.

- *SqlCommand Class*: The SqlCommand class represents a SQL statement or stored procedure for use in a database with SQL Server.
- *SqlDataAdapter Class*: The SqlDataAdapter class represents a bridge between the dataset and the SqlServer database. It includes the Select, Insert, Delete And update commands for loading and updating data.
- *SqlDataReader Class*: The SqlDataReader class creates a data reader to be used with SQL Server using.

System.Data.SqlClient SqlConnection Connection As SqlConnection SqlCommand Command As SqlCommand
--

**Figure 4.3 Classes**

***DataReaders :***

A DataReader is a lightweight object that provides read-only, forward-only data in a very fast and efficient way. DataReader is read-only, meaning, we cannot make any changes (update) to data and forward-only, which means we cannot go back to the previous record which was accessed.

A DataReader requires the exclusive use of an active connection for the entire time it is in existence. We instantiate a DataReader by making a call to a Command object's ExecuteReader command. When the DataReader is first returned it is positioned before the first record of the result set. To make the first record available we need to call the Read method. If a record is available, the Read method moves the DataReader to next record and returns True.

***Data Adapter :***

Data Adapter object utilizes the Fill method to populate a DataSet or a DataTable object with data retrieved by a SELECT command. The Fill method makes use of a data reader to get to the data and the metadata that describe the structure and content of the source tables. The data read is then copied into ad hoc memory containers. The *table mapping mechanism* is the set of rules and parameters that control how the SQL result sets are mapped onto in-memory objects.

DataTable object is added to the DataSet for each result set that the execution of the SELECT statement may have generated. When it comes to copying rows of data from the result set to a given DataTable, the contents of matching columns are merged. By contrast, if no match is

found on the column name, then a new DataColumn object is created and added to the in-memory DataTable.

The Fill method maps the first result set to a DataTable object in the given DataSet. Next, it loops through the result set and adds rows of data to the DataTable. When the end of the result set is reached, the method looks for a new result set and repeats the operation.

During the table mapping step, the data adapter has to find a name for the DataTable that will contain the rows in the result set being processed.

The adapter looks up its TableMappings collection for an entry that matches the default name of the result set. If a match is found, the adapter attempts to locate a DataTable object with the name specified in the mapping in the DataSet. If no such DataTable object exists, it is created and then filled. If such a DataTable exists in the DataSet, its contents are merged with the contents of the result set.

### ***Backup Database:***

Backup strategy minimize the data lost if the data lost of hardware or software failure and recover lost data. A database backup creates a copy of the full database. Not all pages are copied to the backup set, only those actually containing data. Both data pages and transaction log pages are copied to the backup set.

A database backup set is used to re-create the database as it was at the time the BACKUP statement completed. If only database backups exist for a database, it can be recovered only to the time of the last database backup taken before the failure of the database. To backup database you have to give the location that the database is bacup.

### ***Restore Database:***

To Restore Database:

The name of the database to which the transaction log will be applied.

The backup device from where the transaction log backup will be restored.

Individual file and filegroup restores. Files and filegroups can be restored either from a file or filegroup backup operation, or from a full database backup operation.

When restoring files or filegroups, you must apply a transaction log. In addition, file differential backups can be restored after a full file restore.

### ***Adding Database Objects into the treenode:***

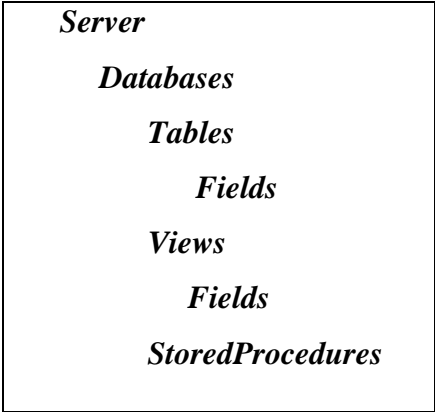


The Windows Forms TreeView control displays a hierarchy of nodes, like the way files and folders are displayed in the left pane of the form. Each node might contain other nodes, called child nodes. Parent nodes, or nodes that contain child nodes, can be displayed as expanded or collapsed. The key properties of the TreeView control are Nodes and SelectedNode. The Nodes property contains the list of top-level nodes in the tree view. The SelectedNode property sets the currently selected node. To determine which treenode was selected the treeview event args object is used[Figure 4.4].

```
Private void treeView1_DoubleClick(object sender, System.EventArgs e)
    TreeView treeView = (TreeView)sender;
```

**Figure 4.4 Treeview Control**

On the left side of the Enterprise Manager Tool there is a treenode console all the objects can be seen on this treenode[Figure 4.5].



**Figure 4.5 Enterprise Manager TreeNode Design**

***Adding Menu Items:***

A menu on a form is created with a MainMenu object, which is a collection of MenuItem objects. You can add menus to Windows Forms at design time by adding the MainMenu control [Figure 4.6] and then adding menu items to it using the Menu Designer. Menus can also be added programmatically by adding one or more MainMenu controls to a form and adding MenuItem objects to the collection.

First add a MainMenu control to the form. Then to add menu items to it add MenuItem objects to the collection. By default, a MainMenu object contains no menu items, so that the

first menu item added becomes the menu heading. Menu items can also be dynamically added when they are created, such that properties are set at the time of their creation and addition.

```
Public Menu()  
    mainMenu = new MainMenu();  
    MenuItem File = mainMenu.MenuItems.Add("&File");  
    File.MenuItems.Add( new MenuItem("&Open") );
```

**Figure 4.6 Menu Items**

Enterprise Manager Tool menu items are File,View,Query,Transfer To Excel and Current Activity menus.

**File menu item** has Open,Save,Save As, Close and Exit sub menu items.

- With using Open menu item user can open the “.sql” files that saved before. When the file opened user can see the content of the file on the query analyzer pad and then execute the query.
- Save menu item helps user to save the sql queries or stored procedures on a file,then he can use it again.
- Close menu item is used to close the open documents before the documents are closed a message is shown this message asks the user if he wants to save the document if the answer is yes the document is saved into a folder that the user wants to save.
- Exit menu item is used to exit from the Enterprise Manager Tool.

**Query menu item** has two sub menu items

- Execute query is used to execute the sql sentences, then the result is shown in a table.
- Parse query is used to check if it is a true sql query or not.

**Transfer to Excel menu item** is used to data transformation from sql tables to Excel sheets.

User can determine the database name and the table name that he wants to see, then records are transferred to the Excel sheets.

**Current Activity menu item:**

Current Activity gives process and lock information at a designated time. This information is a snapshot taken every time when Current Activity button is refreshed. Current Activity provides information about the processes (connections) running, the locks a certain

connection is holding or trying to acquire, and the current and waiting locks on databases and tables.

***Context menus (Popup menus):***

Context menus are used inside applications to provide users access to often used commands by means of a right-click of the mouse. Often, context menus are assigned to controls, and provide particular commands that relate to that precise control[Figure 4.7].

My tool has lots of popup menus when the user right click of a database in the popup many create database, backup database, restore database and drop database items can be seen. these items gives user to do the operations easily. On the tables node user can see poppup menus these are create table drop table, open table. Open table popup item give some alternative such as open all rows, top 50 rows or top 100 rows. If the table has losts of recods with choosing one of these options part of the records can be seen.

```
mnuContextMenu.MenuItems.Add(mnuItemNew);  
mnuContextMenu.MenuItems.Add(mnuItemOpen);
```

**Figure 4.7 Popup Menu**

## **5 EXAMPLE USE OF PROPOSED TOOL**

In this part use of tool is described with an example. This example will show what we can do with this tool and the functionalities of this tool.

### **5.1 DESCRIPTION AND REALISATION OF AN EXAMPLE**

This tool helps users to create databases, tables stored procedures, views. By using this tool everybody can create their personal databases and save information in it.

For example user want to store all the information about authors. First thing to realise this example begin with creating a database then creating tables. We can create our own databases and tables with this tool. We can create a database called personel and inside of this database create a table called Authors[Table5.1] and create another table called Titles [Table5.2]. The relationship between two tables is au\_id field then we can create a view with using this relationship then we can see the all information about the authors and titles in one table.[Table5.3]. Views help user to see all the data that they want to see in a single table.

If user want to see all records about authors in Excel sheets, then this tool helps to transfer data to Excel, so with Excel sheet we can analyze all information about authors more efficiently. This tool helps to write simple sql queries with writing queries we can achieve spesific informations. For example we can find the authors live in “MenloPark” ,by writing an simple query. With this query we can see the authors live in “MenloPark”.

```
Select * from Authors where city=' MenloPark'
```

With query analyzer plane, user can write queries and get results of this queries. They can make lots of operations.

**Table5.1Authors**

au_id	au_lname	au_fname	phone	address	city	state	zip	contract
172-32-1176	White	Johnson	408 496- 7223	10932 Bigge Rd.	Menlo Park	CA	94025	1

**Table 5.2 Titles**

au_id	Title_id	au_ord	royaltyper
172-32-1176	PS3333	1	100

**Table 5.3 Title Authors**

title_id	Title	type	pub_id	price	advance	royalty	ytd_sales	notes	pubdate
PS3333	Prolonged Data Deprivation: Four Case Studies	psychology	0736	19.99	2000	10	4072	What happens when the data runs dry? Searching evaluations of information- shortage effects.	6/12/1991

## **5.2 USE OF THE ENTERPRISE MANAGER TOOL**

When the user want to connect to the Enterprise Manager Tool she/he must register the server. [Figure5.1]

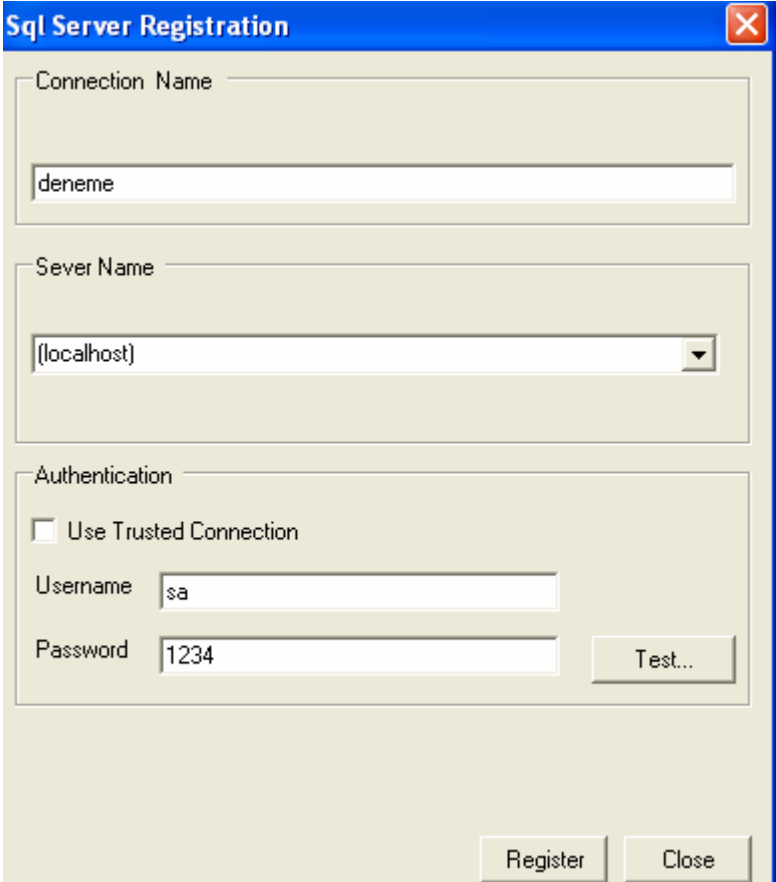
First she write the user name and password then the server name. After user passed this procedure user can achieved Enterprise Manager Tool.

When user connects to SQL Server, the client opens a trusted connection to Microsoft SQL Server, which passes the user's Windows security credentials to SQL Server.

The client opened a trusted connection, SQL Server knows that Windows has already validated the login account.

If the SQL Server finds the user's Windows user account or group account in the list of SQL login accounts in the sysxlogin system table, it accepts the connection.

If the SQL Server does not find the login account set up to accommodate the user, authentication fails and the connection fails.



**Figure 5.1 Registration Page**



### Figure 5.3 Table Properties

To create a new database, in the console tree expand your server then select create new database command and enter the name of the database Figure[5.4].

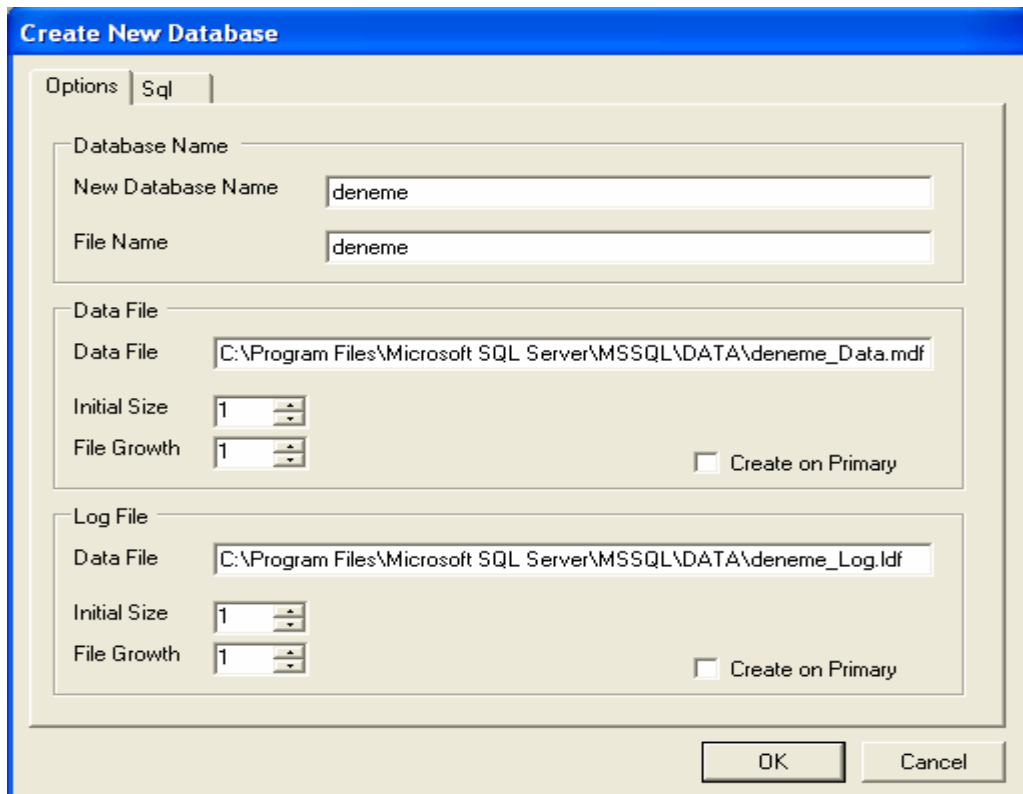
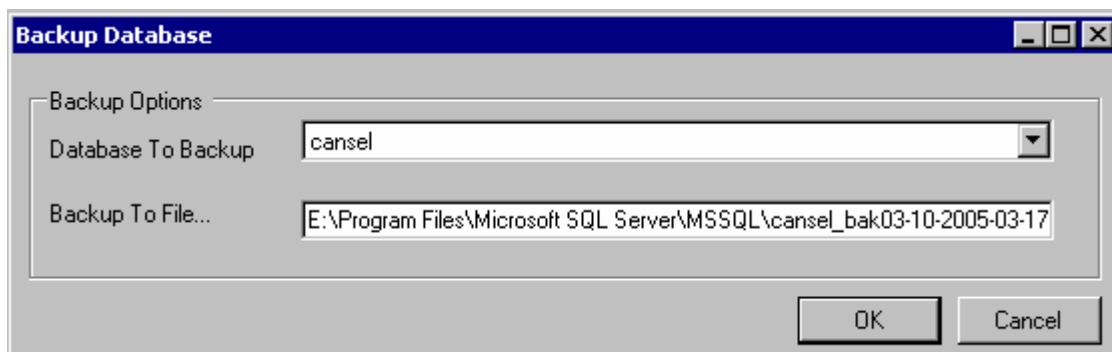


Figure 5.4 New Database

To Backup Database enter the name of the database that you want to backup [Figure 5.5]





### Figure5.5 Database Backup

To Restore Database you can right click on the database on the popup mwnu select RestoreDatabase then write the File where the Database Restore from and the Database Name To Restore To [Figure5.6]

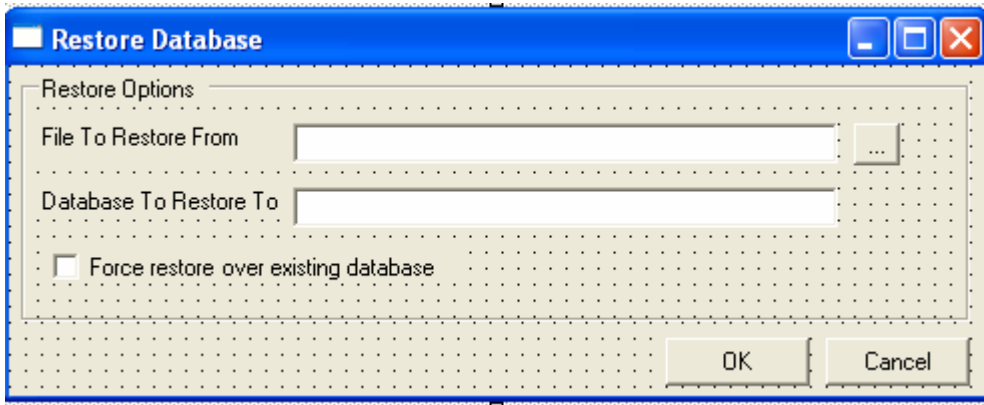
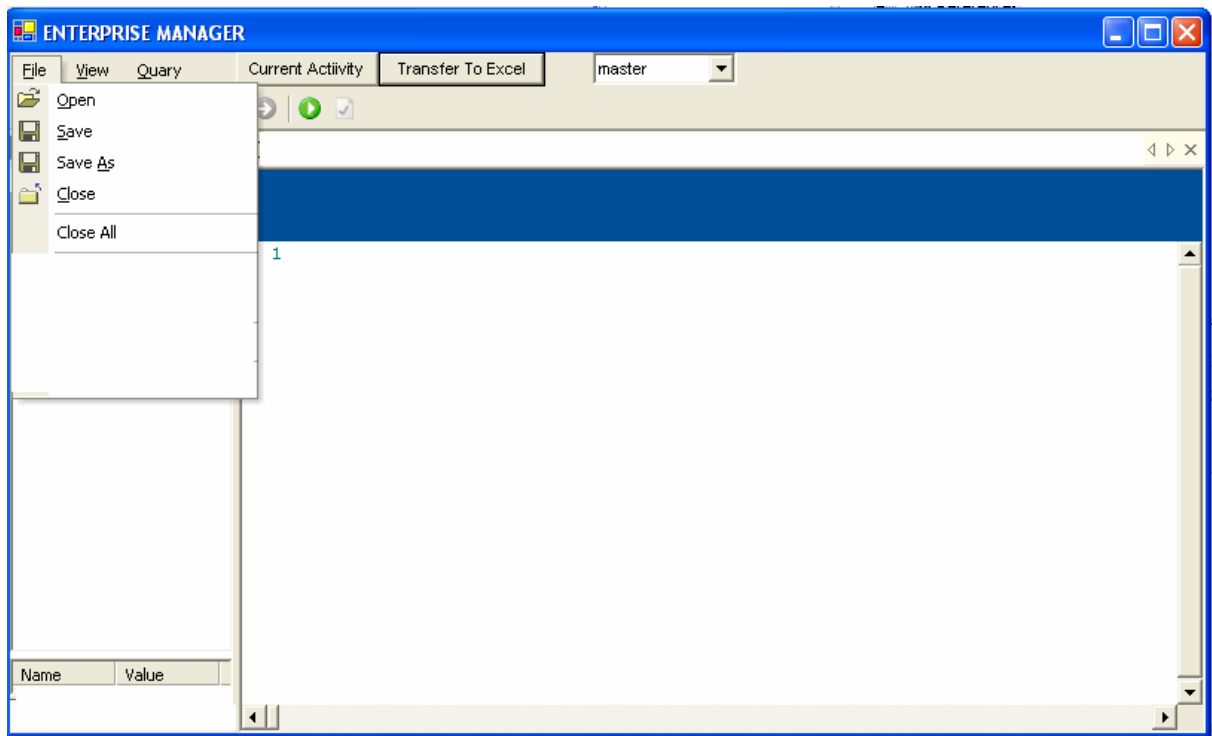


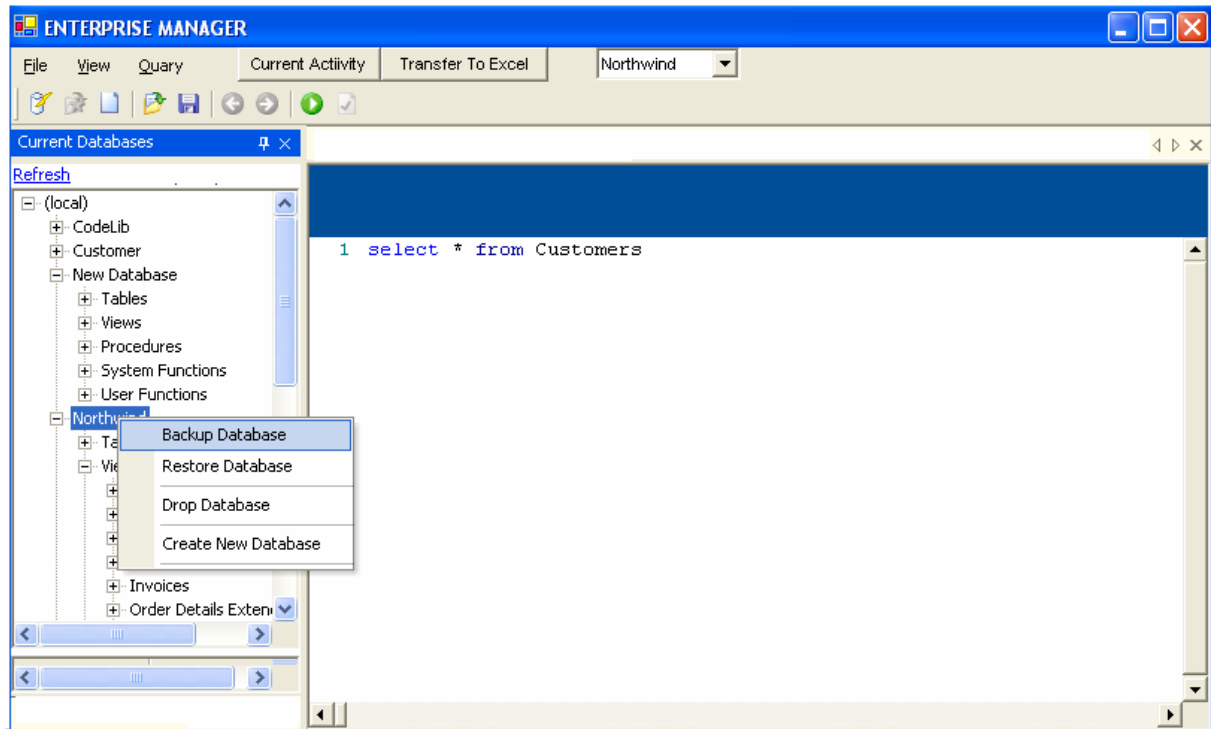
Figure 5.6 Restore Database

You can save your sql queries into a file or you can open sql files in file directory [Figure5.7]



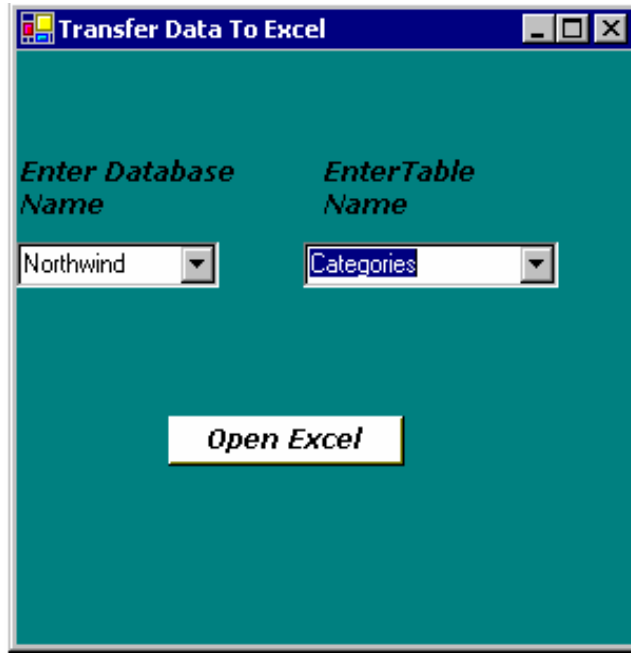
**Figure 5.7 Open - Save .Sql Files**

Figure [Figure 5.8] shows the Database popup menu, you can select one of the item by right click on to the database. This is very simple menu with this menu you can do lots of operations, such as Restore Database Backup Database, Create a New Database or Drop Database



**Figure 5.8 Database Pop Menus**

To see the data in Excel sheets you can select Transfer To Excel button and then enter the database name and table name that you want to see the data when you press Open Excel button an Excel sheet will be open Figure[5.9]

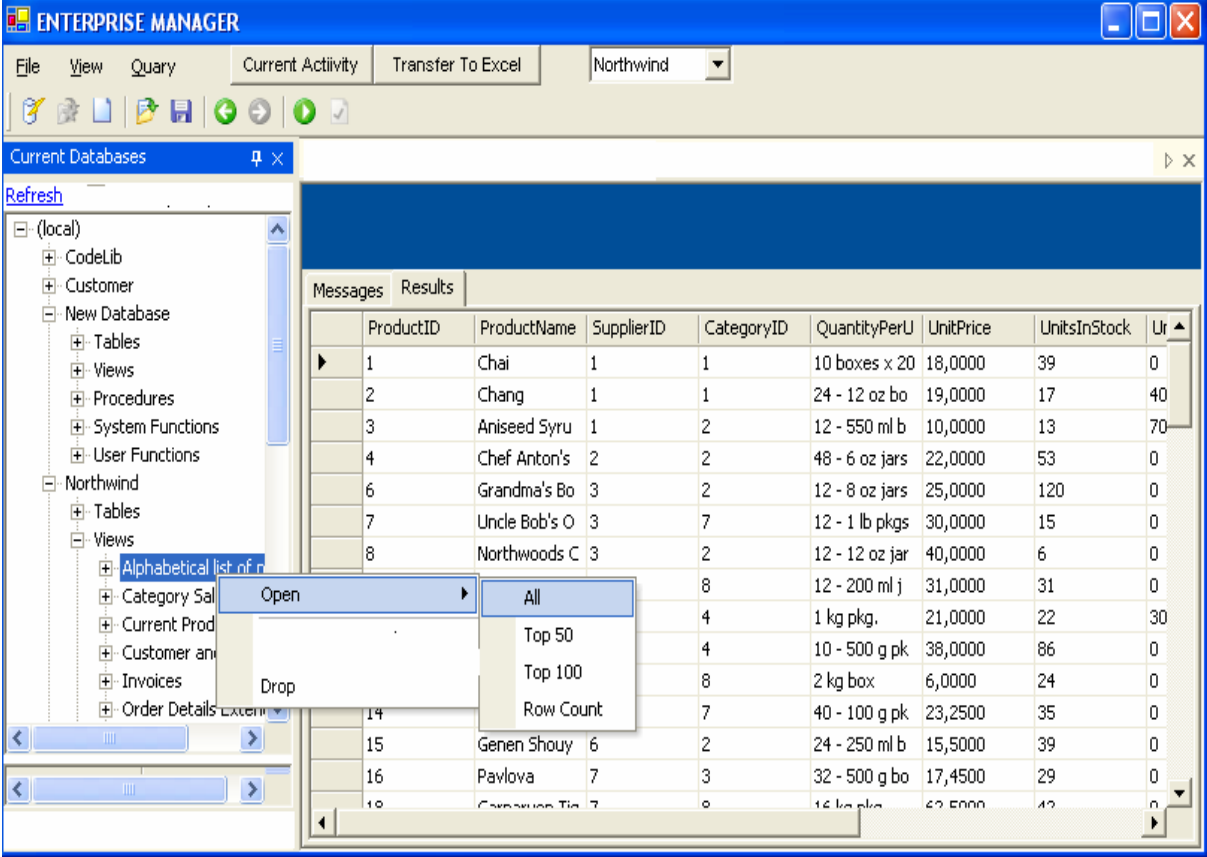


**Transfer Data To Excel Figure 5.9**

	A	B	C	D	E	F
1	CategoryID	CategoryName	Description	Picture		
2		1 Beverages	Soft drinks, coffees, teas, beers, and ales	System.Byte []		
3		2 Condiments	Sauces, relishes, spreads, and seasonings	System.Byte []		
4		3 Confections	Desserts, candies, and sweet breads	System.Byte []		
5		4 Dairy Products	Cheeses	System.Byte []		
6		5 Grains/Cereals	Breads, crackers, pasta, and cereal	System.Byte []		

**Data in Excel Sheet Figure 5.10**

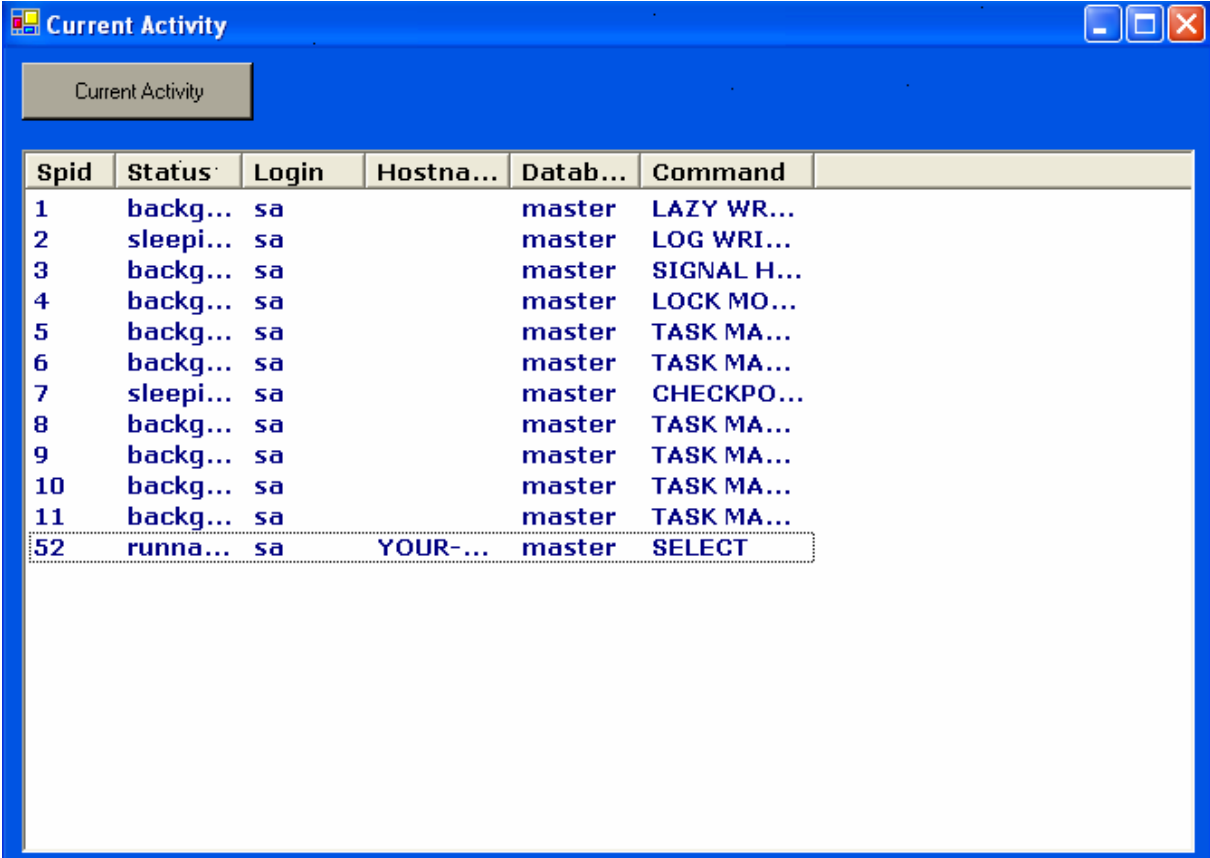
User can right click on a view then select open and drop items if open item is selected then anew popup menu will be seen with using secon popup menu user can open all rows or top50 rows of the view.[Figure5.8]



View Popup Menu Figure 5.11

Using current activity menu item user can see the activities on SQL Server

[Figure 5.9]

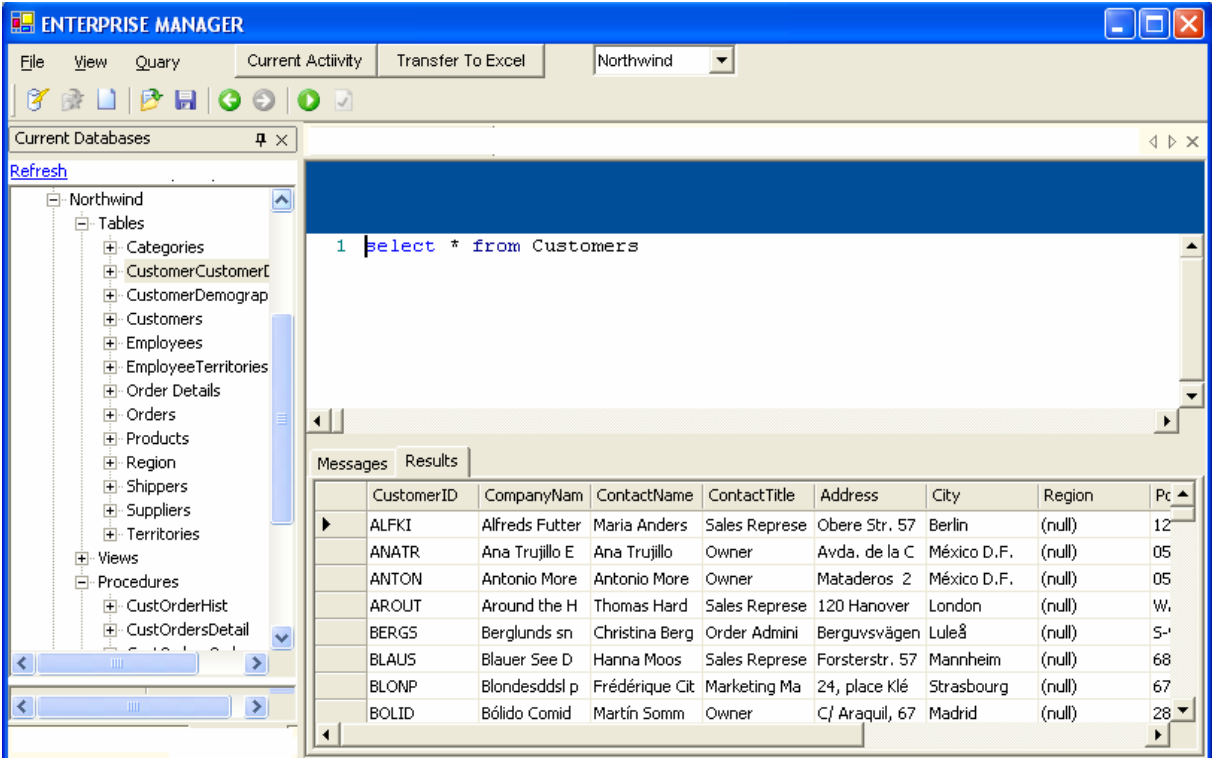


The screenshot shows a window titled 'Current Activity' with a table listing active SQL Server processes. The table has columns for Spid, Status, Login, Hostname, Databasename, and Command. The process with Spid 52 is highlighted with a dotted border.

Spid	Status	Login	Hostna...	Datab...	Command
1	backg...	sa		master	LAZY WR...
2	sleepi...	sa		master	LOG WRI...
3	backg...	sa		master	SIGNAL H...
4	backg...	sa		master	LOCK MO...
5	backg...	sa		master	TASK MA...
6	backg...	sa		master	TASK MA...
7	sleepi...	sa		master	CHECKPO...
8	backg...	sa		master	TASK MA...
9	backg...	sa		master	TASK MA...
10	backg...	sa		master	TASK MA...
11	backg...	sa		master	TASK MA...
52	runna...	sa	YOUR-...	master	SELECT

Current Activity Figure 5.12

On the right side of the Enterprise Manager Tool there is a Query Analyzer Plane [Figure 5.10]. With using this plane you can write queries and when you execute the queries the results can be seen.



Query Analyzner Figure 5.13

## **6. CONCLUSION AND FUTURE RECOMANDATION**

This thesis is about creating a new Enterprise Manager tool. To create this tool I used .NET Technologies and Microsoft SQL Server. MSSQL Server has strong capabilities, so this tool based on SQL Server. Using this tool people can create their own databases and its objects. These objects are tables, views, stored procedures. User can create databases and its objects. Backup Databases, Restore Databases, Drop Databases are main capabilities. Backup Database is very important functionality because backup strategy minimize the data lost sometimes the data can be lost of hardware or software failure this can be a big problem for users, but if the user backup the database the data lost problem can be solve by restoring the Database. This tool gives user to backup databases and restore databases with these functionalities records never lost. User can store the records into the tables, they can insert delete or modify the records. Using this tool is very simple, user dont have a great difficulty while using this tool. There is a simple main menu and popup menus in this tool so, using this tool is very simple. If user want to open a .sql file from a directory its he/she can click the file menu then select the open menu and then select the file that she want to open, its too much simple and user friendly tool so Everybody who want to make operation with Microsoft SQL Server can use this tool. This tool is based on .NET technology so ,this tool can be develop when technology has great functionalities.

.In the future this tool will develop and some strong functionalities such as to create table capability will be inside of this tool.



## REFERENCES

- [1] Vinod Kumar “Execute As Option In Sql Server”, 2005
- [2] Vinod Kumar “Sql Server Schema Definition”, 2005
- [3] <http://msdn.microsoft.com/vsharp/default.aspx>
- [4] Chris Pappas, William Murray “Why C# ”, 2002
- [5] Stephen Bullen “Understanding and Using Windows API Calls for Excel”, 2005
- [6] <http://www.informit.com/articles/article.asp>
- [7] Jeffrey R. Shapiro , *SQL SERVER 2000*, The McGraw-Hill Companies , 2001
- [8] *Programming with C#* ,Microsoft Corporation, 2002
- [9] *ADO.NET*, Microsoft Corporation, 2002
- [10] Adam Nathan “*NET and COM, The Complete Interoperability Guide*”
- [11] <http://www.mylittletools.net>
- [12] <http://www.softpedia.com/get/Internet/Servers/Database-Utils/MySQL-Turbo-Manager-Enterprise.shtml>
- [13] [http://www.oracle.com/enterprise\\_manager/index.html](http://www.oracle.com/enterprise_manager/index.html)
- [14] Mitchell Harper, ”Your First SQL Server 2000 Database”, 2002
- [15] [http://www.mantrotech.com/technology/csharp/article\\_csharp\\_exception\\_1.asp](http://www.mantrotech.com/technology/csharp/article_csharp_exception_1.asp)
- [16] <http://www.theserverside.net/news>
- [17] Gaurav Mantro “C# Exception Handling Basics”, 2005
- [18] Naveen Kohli “Using ADO.NET For Data Access”, 2004
- [19] Joe Mayo ,”Writing C# Expressions “, 2005
- [20] Feldman Arlen, *ADO.NET Programming*, Greenwich, CT
- [21] Troelsen, Andrew. *C# and the .NET Platform*, Berkeley, CA: Apress, 2001
- [22] [www.thedotnetmag.com](http://www.thedotnetmag.com)
- [23] Liberty, Jesse. *Programming C#*, Sebastopol, CA: O’Reilly & Associates, 2001.

## **8 APPENDIX**

CD containing Project source codes, Word document of the thesis and PowerPoint document of the presentation.