DESIGN AND IMPLEMENTATION OF A WORLD WIDE WEB INTERFACE TO QUERY AND UPDATE A RELATIONAL DATABASE

By

RUI ZHANG

Bachelor of Mathematics Science

Capital Normal University

Beijing, China

1986

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 1997

OKLAHOMA STATE UNIVERSITY

DESIGN AND IMPLEMENTATION OF A WORLD WIDE WEB INTERFACE TO QUERY AND UPDATE A RELATIONAL DATABASE

Thesis Approved:

H. Lu

Thesis Adviser

R. E. Haskell

J Chandler

Thomas C. Collins

Dean of the Graduate College

ACKNOWLEDGMENTS

I wish to express my sincere appreciation to my thesis advisor, Dr. HuiZhu Lu for her intelligent supervision, constructive guidance, and constant inspiration. I also wish to express my gratitude to Dr. John P. Chandler for his guidance, encouragement, and friendship. My appreciation extends to Dr. Kathleen Kaplan for her guidance. My sincere appreciation extends to Dr. G. E. Hedrick, whose assistance, and encouragement are also invaluable.

My appreciation extends to our research team member, Mr. Yuwen Lin, for his suggestions and help. My thanks go to all of my fellow students who broadened the quality of my education at OSU.

I would like to give my special appreciation to my husband, Shaokai Wen, for his precious suggestions for my research, his strong encouragement at times of difficulty, love and understanding throughout this whole process. My respectful thanks goes to my parents Mr. Chuqiao Zhang and Mrs. Yuying Cai for all the love and support they have given me throughout my life. And, I would like to thank all other members of my family for their love, encouragement and confidence they have contributed to me.

I would also like to express my gratitude to all of those who have helped by giving many valuable suggestions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table Page

Abstract

World Wide Web (WWW) is widely used today. It can show static information, and provide interactive information. WWW users can access the system anywhere, anytime. In this thesis, we design and implement a WWW interface to query and update a relational database. We use Microsoft Windows NT Server 4.0 Network Operating System, Microsoft SQL Server 6.0, Microsoft Internet Information Server 3.0(MS IIS) for our server. MS Windows NT Server supports network and security. MS SQL Server is used to integrate a Relational Database Management System (RDBMS) with WWW to handle the database queries and other operations. MS IIS and MS Windows NT Server provide an excellent platform for developing web sites.

CHAPTER I

Introduction

This thesis describes the design and implementation of a World Wide Web (WWW) interface to query and update of a relational database. The WWW allows users to access the information managed by the WWW server anytime and anywhere. The information can be a set of relational database records or others. A relational database is defined as a database in which the individual files, termed relations, hold data in the form of tables. Each record has a fixed number of fields, all of which are explicitly named. The fields within a table are distinct, and repeating groups are not allowed. There are no duplicate records and no predetermined sequence of records. In the construction of a relational database, it is necessary to give consideration to the inherent relationships between attributes (fields) of a record. Software distribution and cross-platform compatibility problems can be solved by implementing the system through the WWW [BK, 1996].

A distributed database system consists of a collection of sites, each of which maintains a local database system; each site is able to process local transactions, those transactions that participate access data only in that single site. A site may participate in the execution of global transactions, those transactions that access data in several sites.

The execution of global transactions requires communication among the sites [KHSA, 1991]. The main advantage of a distributed database is the ability to share and access data in a reliable and efficient manner. The other advantages are hardware and operating system independence, modularity, and efficient division of labor between client and server machines. Clients and servers communicate asymmetrically. Clients make requests to servers; servers attempt to grant these requests if they are valid. The server's responses may be to return data or generate commands to control a hardware device.

In the client-server database application, clients may communicate with a server by means of programming interfaces. These interfaces provide connectivity between applications and databases. So the client can manage the user interface, accept data from the user, process application logic, generate database requests, transmit database requests to server, receive results from the server and format the results. The server may accept the database requests from clients, process database requests, format results and transmit to client, perform integrity checking (maintain database overhead data and provide concurrent access control), perform recovery, optimize query and update processing.

The objectives of this thesis are to design and implement a World Wide Web interface for qualified users to query a relational database on the internet, and to send updating information to the server for updating the database. The project uses ODBC as a multiple database management system. The system supports multiple applications (browser). An authorized user can access the browser and view the results, the system administrator performs system maintenance tasks through system administrator interface.

The thesis contains five Chapters. The second Chapter briefly reviews basic concepts and definitions including MS Windows NT, IIS, ASP, MS SQL, and ODBC. The design implementation of a world wide web interface to query and update a relational database is discussed in Chapter III. The procure of define a DSN, create tales, and permissions for the database is also presented. Then, results is discussed in Chapter IV. Finally, conclusions and future work are included in Chapter V.

CHAPTER II

Review of the Literature

The client-server paradigm is a good approach to implement the distributed database management. In this chapter, the author will briefly overview previous research results and currently available products for internet access and security handling. MS Windows NT Server 4.0 Resource Kit [MSCA, 1996] [MSCS, 1996] [MSCI, 1996] on-line Documentation includes Windows NT Server 4.0 Resource Guide, Windows NT Server 4.0 Network Operating System Guide, Windows NT Server 4.0 Internet Guide, Windows Workstation Resource Guide. Those are special important resources. Most of contents in this chapter are adopted from that resource kit.

Microsoft Windows NT Server

MS Windows NT Server is a multitasking environment. This means that multiple threads of execution may be run simultaneously. The threads , do not actually run simultaneously, but take turns at the CPU.

In Windows 3.x, each EXE program runs as a task, but the tasks are scheduled nonpreemptively; each task runs until it explicitly yields the processor by calling some Windows Application Programming Interface (API) function. This means that if an

application (EXE or COM file) uses the CPU, no other application can be initialized until the current application explicitly releases the CPU.

MS Windows NT Server uses preemptive scheduling. This means that the operating system forces tasks to yield control periodically. In this type of system, an uncooperative process (the process can not terminal normally) does not cause the entire system to stop. Other processes always continue to run. This is very important. Under MS Windows NT Server, each EXE program runs as a task, each subsystem (like MS SQL Server, IIS) runs as a "server" task (system task), and it responds to the requested messages from an application.

The other very important difference between the tasking models of Windows 3.x and MS Windows NT Server is that under MS Windows NT, each task runs in its own address space and cannot directly address another task's memory. This is an essential part of the design of MS Windows NT Server. This ensures security and robustness because an errant application can not damage any server data, windows in NT Server system structures, or memory, but applications that rely on sharing memory between tasks do need to be modified.

In MS Windows NT Server, the API calls from the clients are converted into messages that are sent to its subsystem of the Server. Then makes a Local Procedure Calls (LCPs) to initialize the subsystem of the server to execute the Dynamic Linked Library (.DLL) file to respond the requests of the Client. For performance reasons, some functions are actually handled in the DLLs, and sometimes API functions are deferred until a few of them can be sent to the server at once. Sending messages between tasks

require a task switch and, without a few tricks, the performance impact is substantially improved.

In any case, when the system receives a message, it makes direct calls at the MS Windows NT Server Executive, it provides system services that run in KERNEL mode on the caller's thread, the mode similar to the way Windows 3.x programs run the code in the USER, and KERNEL modules. Running the NT Sever Executive calls on the subsystem (or server) thread instead of the client (application) thread provides a high degree of security because of the isolation of the address spaces.

As a multitasking system, MS Windows NT Server Executive services include the actual message-passing mechanisms, process management, security services, and virtual memory management. I/O management is also provided to support the subsystem file functions on MS Windows NT Server through supported file system. The MS-Windows NT I/O manager manages the native executive I/O calls (made by subsystems) and packets them into I/O packages sent to the file systems that break up the requests into actual media I/O requests that are sent to MS-Windows NT device drivers.

In the resource sharing multitasking MS Windows NT Server, security is very important to guarantee the subsystem will execute correctly. In MS Windows NT Server all the objects (including files, processes, and kernel objects) are restricted in their access to other system resources. This is done by using a system of Security Identifiers (SIDs) and access control lists (ACLs).

When a user logs on to an MS-Windows NT system, the user name maps to a process access to kernel that determines the SIDs for that process and any child processes it creates. The SID is determined firstly for the user name itself, and secondly

based on groups that the user belongs to. The system administrator can create groups, assign users to groups, assign privileges to groups and control the resource access.

Due to the multithreaded, multitasking support on MS Windows NT Server, it can be used as the Server on the network. Generally the network can be divided into two kinds of networks: tightly coupled (sharing memory), or loosely coupled (over a network) according the resource sharing and frequency of the communication between the process. MS-Windows NT can be used for both networks. Some of the interprocess communication objects are designed to work in either environment in a manner that is virtually transparent to application programs. Client-server computing, in which client tasks request services from server tasks, has become easier and more powerful than ever because the tasks may reside in separate machines or the same machine without modification of their process. Due to the advantage of the Windows NT Server, we use the system for our server system to manage the web enabled database.

Microsoft Internet Information Server and Active Server Page

Microsoft Internet Information Server (IIS) is the only World Wide Web server that is tightly integrated with the Microsoft Windows NT® server operating system and is designed to deliver a wide range of Internet and intranet server capabilities. By optimizing around the Windows NT Server platform, Internet Information Server delivers high performance, excellent security, ease of management and is up and running in minutes. It serves as the best platform for both integrating with existing solutions and delivering a new generation of Web applications.

[http://microsoft.com/iis/default.asp]

Microsoft's Active Server Pages (ASP) with IIS 3.0 offer the web developer a flexible, easy to use, scaleable methods to interact with ODBC compliant databases for an Internet site.

ASP encompasses the capabilities of both JavaScript and VBScript with the added bonus that components can be easily added to extend the Internet/intranet application. Using ASP as part of the development we can initially scale up to a Microsoft SQL Server 6.5 database; we also can access other vendor databases that are ODBC compliant.

Files created with Active Server Pages have the extension .ASP. With ASP files, user can activate a Web site using any combination of HTML, scripting--such as JavaScript or Visual Basic® Scripting Edition (VBScript)--and components written in any language. This means a ASP file is a file that can contain any combination of HTML, scripting, and calls on components. When user make a change on the ASP file on the server, user need only save the changes to the file, the script will automatically be compiled when next time the Web page is loaded.

ASP Model is an object model. How it works? When a browser requests an ASP file from a Web server, The Web server calls Active Server Pages to read through the ASP file, execute any of the commands contained send the resulting HTML page to the browser. An ASP file can contain any combination of HTML, script, or commands. The script can assign values to variables, request information from the server, or combine any set of commands into procedures.

Microsoft SQL Server

As mentioned before the MS Windows NT Server is the framework of the distributed database system, but if there is no corresponding subsystem to responds to the request from the client, no task can be executed. Thus, MS SQL Server for Windows NT Server is designed to manage large databases for mission-critical applications and to meet the demands of networked client/server applications. It is running as a subsystem in the server to respond the requests from the client to manipulate the relational database in heterogeneous environment by using Open Database Connectivity (ODBC). MS SQL server is the best choice that provides an essential link in the integrated client/server architecture. MS SQL server is using Structured Query Language (SQL) that was developed by IBM in the mid-1970s under the name SEQUEL as the manipulating language. SQL has been endorsed as the recognized industry standard for manipulating databases by the American National Standards Institute (ANSI). It is the data access language used by many database management system (DBMS) products.

SQL language allows users to run queries on a relational database, e.g., create new tables, accesses existing tables, and manipulate data, etc. All these activities can be implemented only by the SQL engine that is the SQL server. The Microsoft SQL database servers provides a multiple user relational database management systems (RDBMSs).

SQL commands can be used interactively as a query language or embedded in applications programs. In performing a query, SQL accepts one or more relations as input and produces a single relation table as output. In querying multiple tables relation,

the tables are joined by SQL. Provisions are included in SQL for inserting new data into a table, deleting data from a table, or modifying data.

The Open Database Connectivity (ODBC) is used to manipulate the relational database in heterogeneous environment. We briefly introduce the ODBC in the next section.

Open Database Connectivity (ODBC)

ODBC technology provides a common interface for accessing heterogeneous SQL databases. ODBC is based on Structured Query Language (SQL) as a standard for accessing data. This interface provides maximum interpretability: a single application can access different SQL Database Management Systems (DBMS) through a common set of code. This enables a developer to build and distribute a client/server application without targeting a specific DBMS. A database drivers may be used to link an application to the user's choice of DBMS.

The main components of MS ODBC are ODBC driver manager and ODBC drivers. Each ODBC driver implements ODBC function calls and interacts with a particular DBMS. To accomplish this, the driver needs to manage the communication protocol between an application and a data source when the application make a call to connect to the data source. After a connection is established, the driver sends the requests to the DBMS made by the application. To solve semantic conflict, the driver also converts data types defined by the DBMS to corresponding ODBC SQL data types. Finally, it returns results containing the required information to the application. The ODBC drivers are provided by individual DBMS manufacturers.

The application/driver architecture for 32-bit environments is depicted in Figure 1 which is adopted from [MSCA, 1996] with a modification.

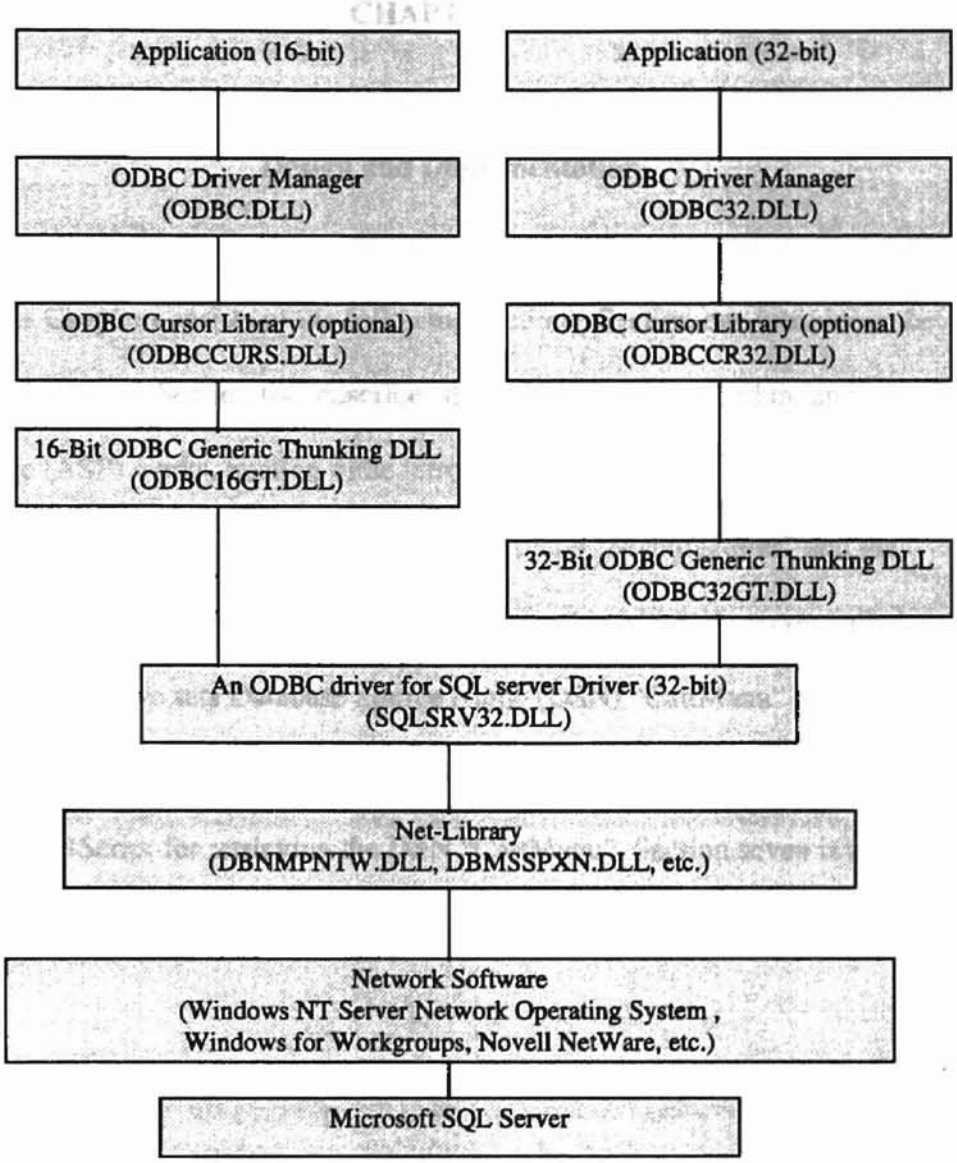

Figure 1: Application / driver architecture for 32-bit environments

CHAPTER III

Design and Implementation

This Chapter consists of the following sections. Section one introduces the system environment. Section two describes the server system procedure and how Active Server Page (ASP) works. Section three introduces the structure of the database. Section four illustrates the procedure to create a database name "Manufacturer" and the permissions for access to the database "Manufacturer" in MS SQL server called "LUNT". Section five sets Database Source Name (DSN) "CattManu" which uses the database "Manufacturer". Section six illustrates how to write .ASP file which combines HTML and VBScript for retrieving the DSN "CattManu". Section seven is a part of the flow chart for the project.

System Environment:

In this project Microsoft SQL Server 6.5, Microsoft Windows NT Server 4.0, and Microsoft Internet Information Server 3.0 are used at the server side. The required system environments on the server side and the client side are shown in Table 1.

Server	Hardware	32-bit x86-based microprocessor (such as Intel 80486 or higher, Intel Pentium.) one or more hard disks with 200MB minimum free hard disk space , one or more network adapter cards.
	Memory	16 MB minimum, 64 MB memory recommended
	Software	MS Windows NT Server 4.0 System MS Internet Information Server 3.0 MS SQL Server 6.5
Client		any platform Internet connection HTML compatible browser

Table 1: System environments

The next section introduces the ASP model and how the system works when a browser has a request for .ASP files which reside at the server side.

Server Side Description:

The codes (.ASP files) are developed on Windows NT 4.0 with MS SQL Server 6.5 as a DBMS. We use the 32-bit ODBC Drivers for MS SQL Server 6.5.

An .ASP file begins to run when a browser requests an .ASP file at Web server. Web server then calls .ASP file. The server reads through the requested file from top to bottom, executes any VBScript or JavaScript commands, and sends an HTML page to the browser. The web system with Active Server Pages (ASP) model architecture is depicted in Figure 2.

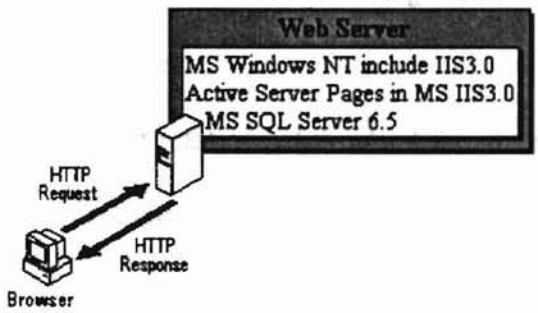

Figure 2. The Active Server Page model

In our Web server system, the Server host uses MS Windows NT Server 4.0 and MS Internet Information Server 3.0 (IIS 3.0). The IIS 3.0 supports Internet Server Application Programming Interface (ISAPI). IIS 3.0 is a product of Microsoft as an alternative to Common Gateway Interface (CGI). The ISAPI calls a Dynamic Linked Library (.DLL) file that is usually written in C++. This mechanism is much faster and uses fewer system resources than a CGI call. It receives client requests from web server and retrieves or updates the database accordingly.

The block diagram is given in Figure 3 to illustrate how the server works in the client/server system.

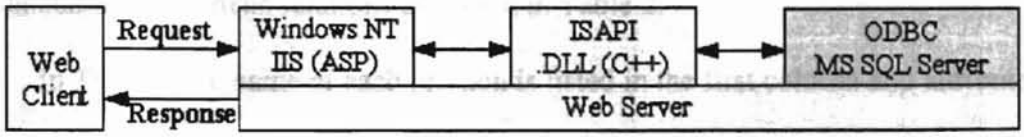

Figure 3. Web server system

The project uses MS Windows NT Server integrated security to implement security requirements for the project.

The next section describes relations of the database, their attributes, and an Entity-Relationship diagram for the project.

The Database

We used MS SQL Server 6.5 to develop a database and call it "Manufacturer". The database has fourteen relations as shown in Table 2.

In Table 2, the name of each relation is listed in the first column and attributes of each relation are listed in the second column.

Assembly_Capabilities (It holds information regarding the manufactures assembly capability)	codekey; arc_welding; mig_welding; tig_welding; laser_welding; spot_welding; welding_other; swage_press_tonnage; swage_part_width; swage_part_height; thermal_shrink_part_length; thermal_shrink_part_width; thermal_shrink_part_height; brazing_temperature; tin_lead_soldering; silver_soldering; indium_soldering; soldering_other ; roll_riveting; blind_riveting; flat_riveting; riveting_other; thermal_adhesion_temperature; thermal_adhesive_part_length; thermal_adhesive_part_width; thermal_adhesive_part_height; pressure_adhesion_pressure; pressure_adhesive_part_length; pressure_adhesive_part_width; pressure_adhesive_part_height; adhesives_other.
Basic_Company_Information (It holds basic information about the companies)	codekey; cage_code; company_name; street_address; city; state; zip_code; phone_number; fax_number; date_of_data_capture; contact1_name; contact1_phone; contact1_email; contact2_name; contact2_phone; contact2_email; size_of_typical_equipment; calendar_days_aro; government_contracts.
CAD_CAM_Capabilities (It holds information regarding the manufacturers' CAD/CAM capabilities)	codekey; iges; pdes; dxf; other_cad; master_cam; smart_cam; ez_cam; other_cam.
EC_EDI_Capabilities (It holds all manufacturers with EC/EDI capabilities)	codekey; computer_type; ibm_speed; monitor_size; hard_drive_size; dos windows; word; certral_email; www_address; modem_speed; modem_number; ram; cd_rom_speed.
Memo (It Holds other information not included in the other tables)	codekey; memo
Plating_Capabilities	codekey; material; length; width; height;
Sculptured_Shapes (It holds all manufacturers who can produce sculptured parts)	codekey; material; equipment_name; min_length; max_length; min_width; max-width; min_height; max_height; min_profile_tol; min_groove_profile_tol; min_thread_class; min_hole_location_tol; min_hole_profile_tol.
Flat_Shapes (It holds all the manufacturers who can produce flat parts)	codekey; material; equipment_name; min_length; max_length; min_width; max_width; min_thickness; max_thickness; min_profile_tol; min_groove_profile_tol; min_thread_class; min_hole_location_tol; min_hole_profile_tol; min_gear_profile_tol; min_angle_tol
Prismatic_Shapes (It holds all manufacturers who can produce prismatic parts)	codekey; material; equipment_name; min_length; max_length; min_width; max_width; min_height; max_height; min_profile_tol; min_groove_profile_tol; min_thread_class; min_hole_location_tol; min_hole_profile_tol; min_gear_profile_tol.

Revolved_Shapes (It holds all manufacturers who can produce revolved parts)	codekey; material; equipment_name; min_length; max_length; min_diameter; max_diameter; min_profile_tol; min_concentricity_tol; min_groove_profile_tol; min_thread_class; min_hole_location_tol; min_hole_profile_tol; min_gear_tol; min_head_profile_tol.
Procurement_History (It holds manufacturers with a procurement history with the government)	codekey; service_branch; last_delivery_date; quantity_parts; quality_soecs.
Treatement_Capabilities (It holds all manufacturers with treatment capabilities)	codekey; treatment; length; width; height; tonnage.
Inspection_Capabilities (It contains manufacturers' inspection capabilities and certifications)	codekey; first_part_destructive_testing; mechanical_gaging; x_ray; magnetic; liquid_penetrant; ultrasonic; coordinate_measurement_machine; other_inspections1; other_inspections2; other_inspections3.
Quality_specifications (It holds manufacturers with quality certifications)	codekey; iso_certification_no; quality_certification1; quality_certification2; quality_certification3.

Table 2: Relations and their attributes

Figure 4 shows an entity-relationship (E-R) diagram for the database. The database system may reject any operations that violate the referential integrity constraints. This can reduce the chance of possible database corruption.:

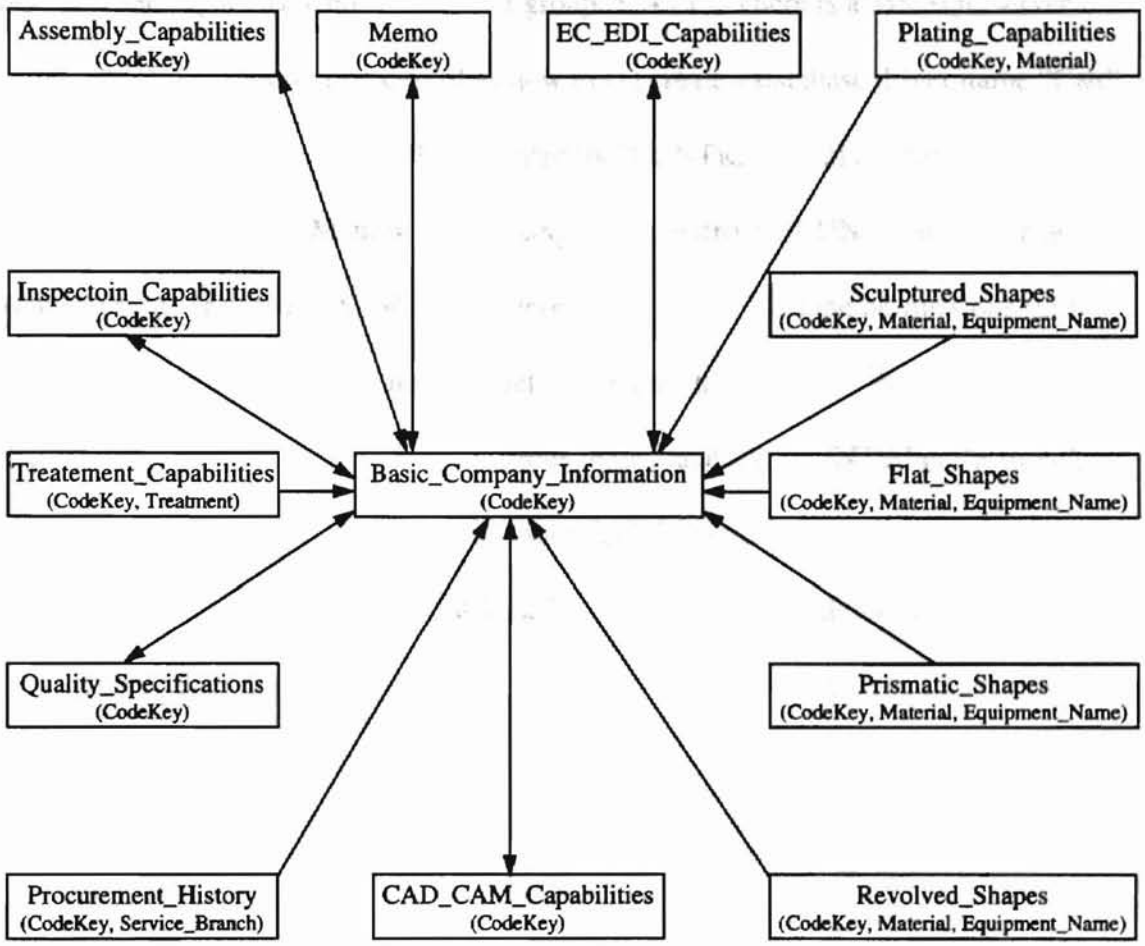

Figure 4: An E-R diagram for the project

Creating a database "Manufacturer" and permissions for users to access and updating the database

For this project, the first step is to create a database in the MS SQL server named "LUNT". We have Microsoft SQL Servers which has two server groups (CATT and SQL 6.5, see Figure 5). Under the server group "CATT", There is a MS SQL server named "LUNT". This section describes how to (1) create a database driver name "Catt" under the MS SQL server "LUNT" indicated by "LUNT\Catt", (2) a database name "Manufacturer" (LUNT\Manufacturer) using database driver "LUNT\Catt", (3) a new login for the user to login the MS SQL server "LUNT", and (4) the permissions for a user or a user group to access, insert, delete, or update the database "Manufacturer" (LUNT\Manufacturer) or one of the relations in the database "LUNT\Manufacturer".

The procedure for creating a database "LUNT\Manufacturer" and permissions for a user or a group to access the database by using MS SQL is described below.

- Click on "Start" button of Windows 95, and go further to "MS SQL Enterprise" under MS SQL 6.5 sub-menu to obtain the screen (Microsoft SQL Enterprise Manager) as shown in Figure 5.

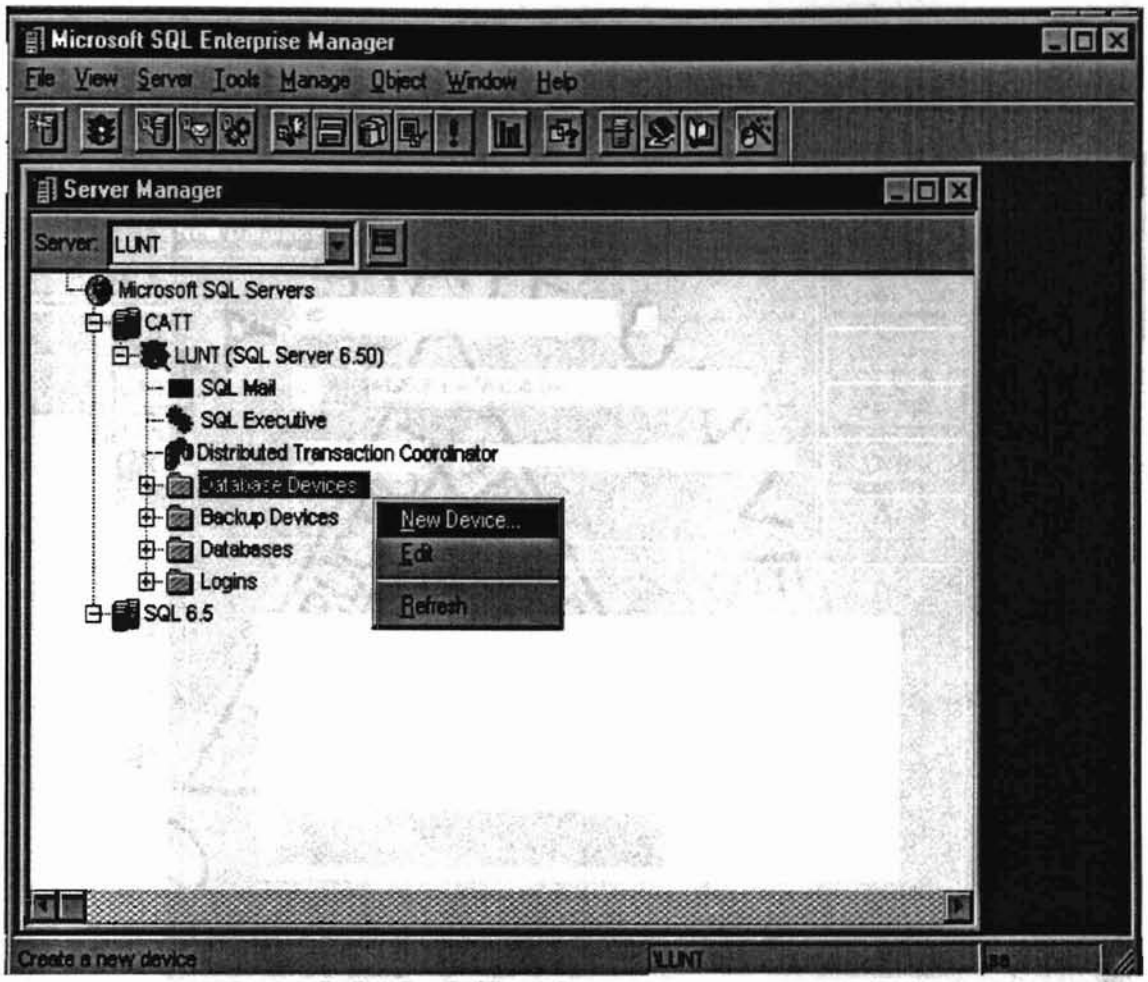

Figure 5: "Microsoft SQL Enterprise Manager" window and

Open "New Database Device - LUNT" window

Under "Microsoft SQL Enterprise Manager" window (Figure 5), there are two MS SQL server groups, CATT and SQL 6.5. In the sub-window "Server Manager", under "CATT", there is a MS SQL server named "LUNT" which has "SQL Mail", "SQL Executive", "Distributed Transaction Coordinator", "Database Devices", "Backup Devices", "Databases", and "Logins" functions. These functions are also available for any other MS SQL server.

- Create a database device named "Catt" (Figure 6).

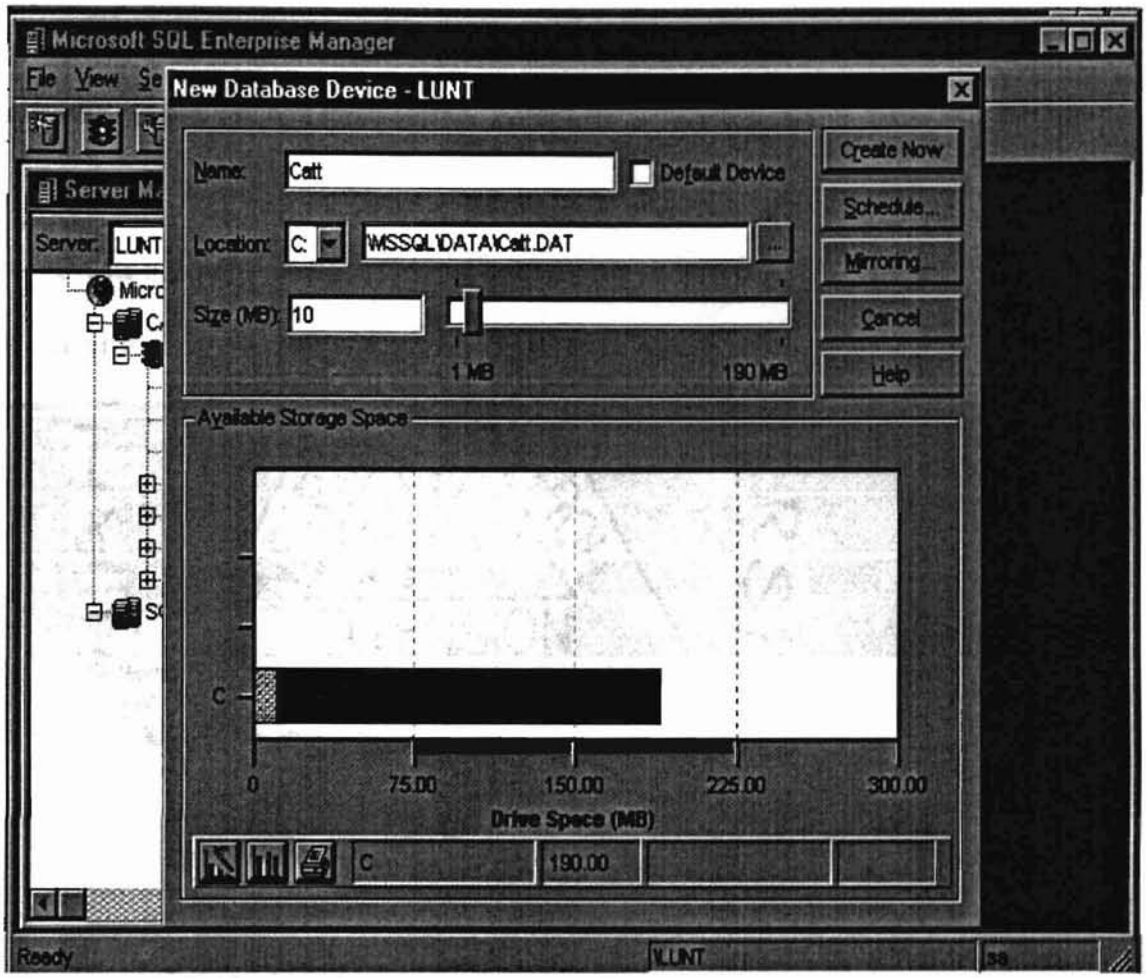

Figure 6: Create a new database device "LUNT\Catt"

In the window shown in Figure 5, move the mouse to point at the "Database Devices", click the right button on the mouse and highlight the "New Device...". The window of creating a database device will be opened as shown in Figure 6. In Figure 6, we define the new database device name (Catt) and its location (or path), and decide the maximum storage space for the database device, which is estimated as 10 MB for the

Catt. Then click the "Crate Now" button, the system will create a database device named "Catt" and a file "Catt.DAT" corresponding to it under the defined location (or path) automatically. The database administrator (DBA) who has such central control over the system can add more storage spaces for a database device later on.

- Create a database named "Manufacturer" which uses the "LUNT\Catt" database device under the MS SQL server "LUNT" (Figure 8).

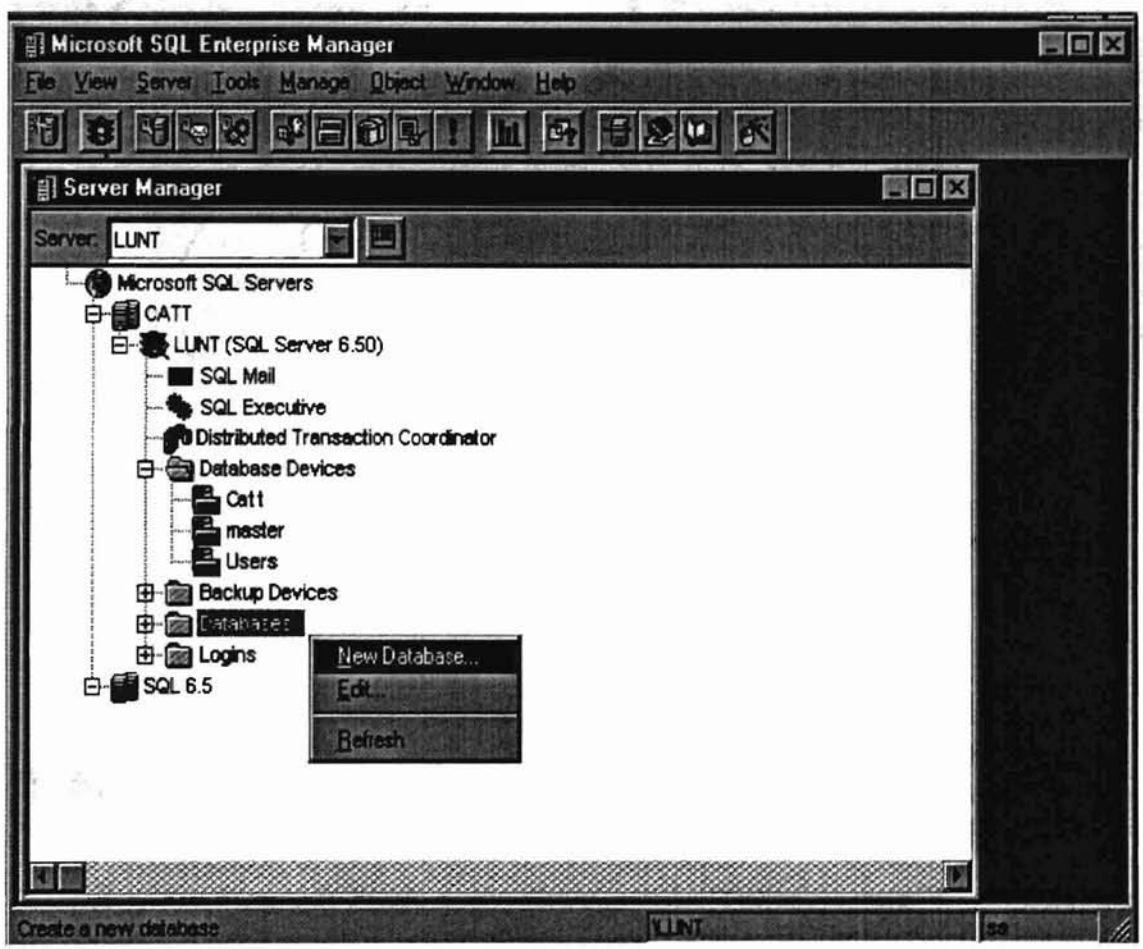

Figure 7 : Highlight the "New Database"

In Figure 7, move the mouse to point at "Databases" under the MS SQL Server "LUNT", click the right button on the mouse and highlight "New Database...". Then, the window for creating a new database is opened as shown in Figure 8.

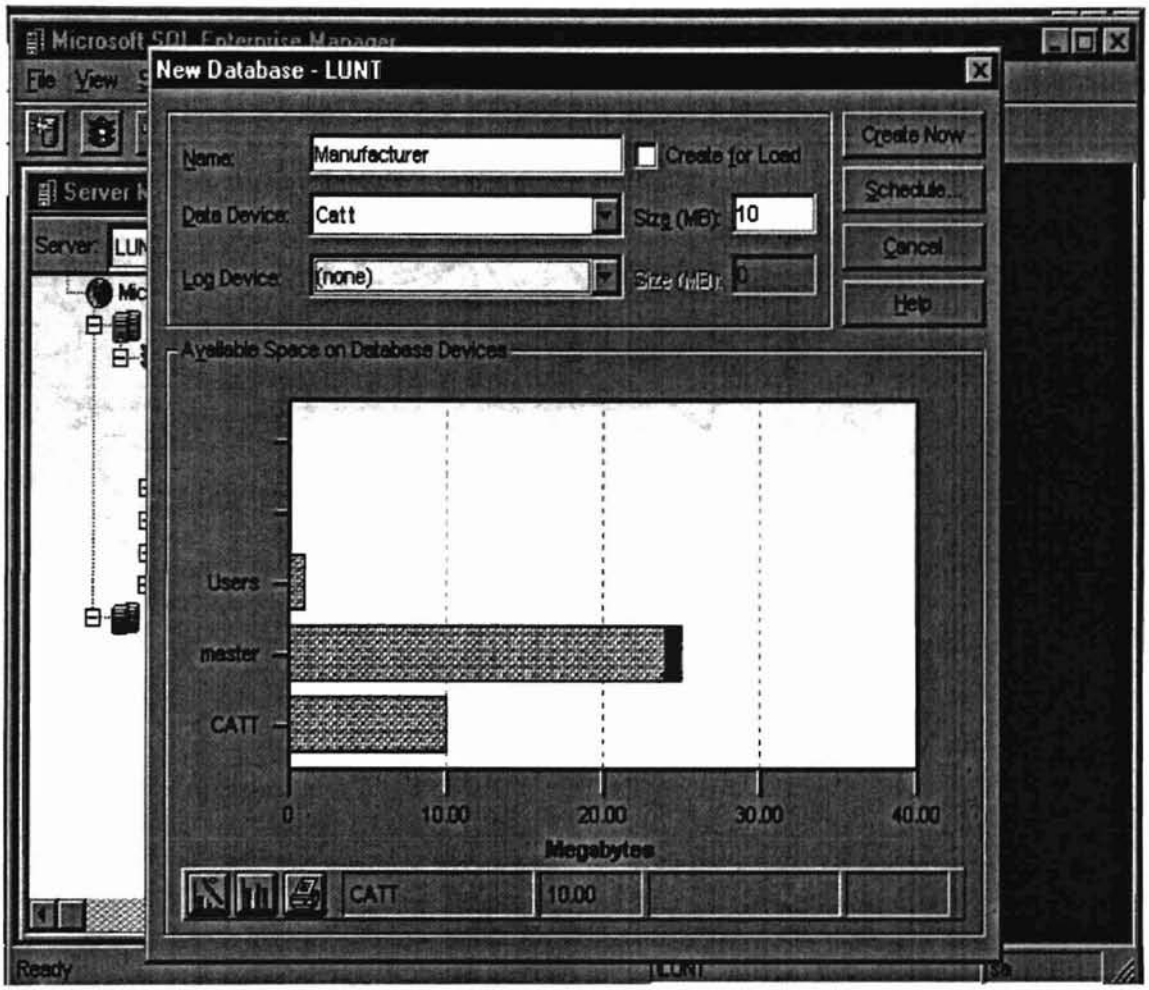

Figure 8: Create a new database "LUNT\Manufacturer"

In Figure 8, we define the name of the new database as "Manufacturer" and use the database device "Catt" which was created in Figure 6 earlier. Click the "Create Now" button, the system will create a storage space in database device "Catt" driver under the MS SQL server "LUNT" (quoted as "LUNT\Manufacturer" in the rest of the thesis) automatically.

- Open "Manage Logins - LUNT" window to create a new login for a user to access the MS SQL server "LUNT" as shown in Figure 9.

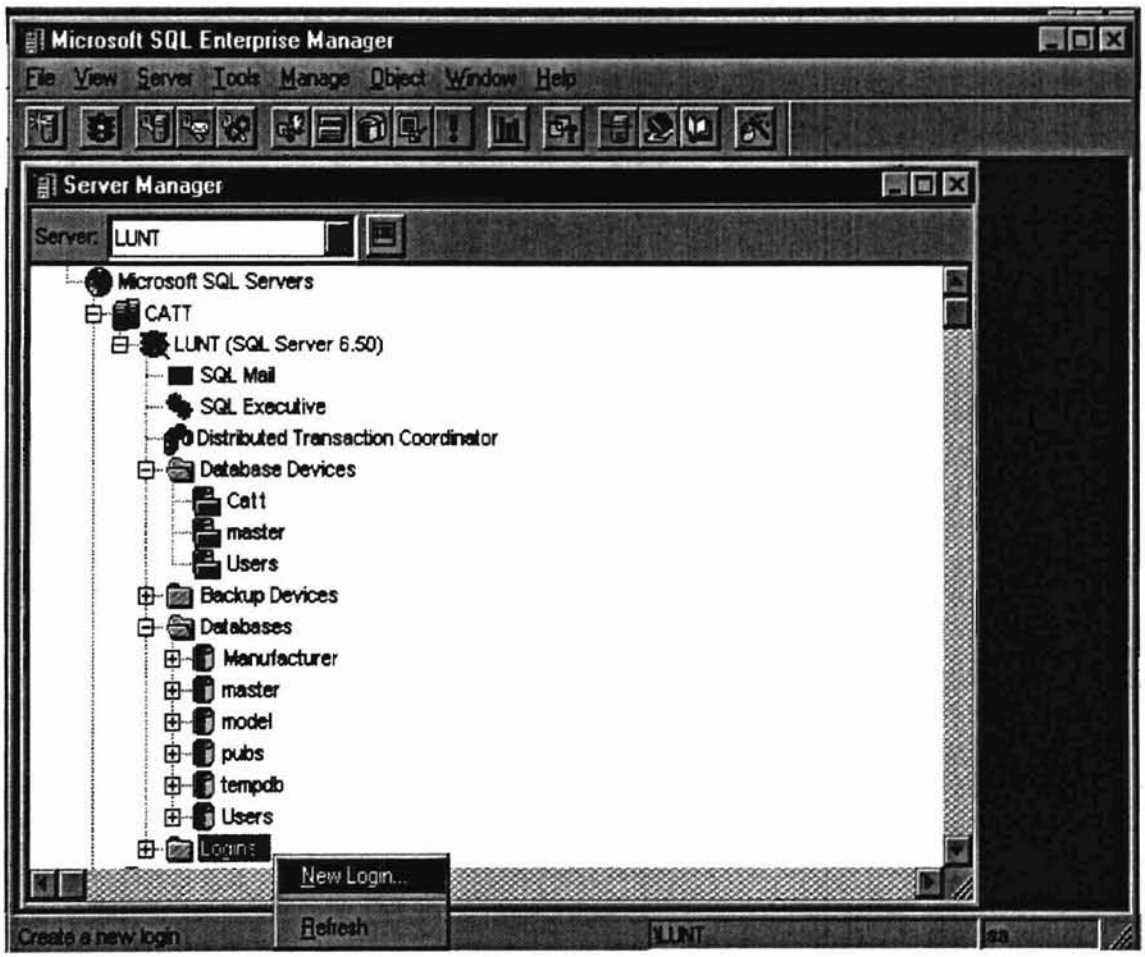

Figure 9: Create a new login

In Figure 9, move the mouse to "Logins" under the MS SQL server "LUNT" in the "Server Manager" sub-window, click the right button of a mouse, and highlight "New Login...". Then click the mouse once. This will bring up the "Manage Logins - LUNT" window as shown in Figure 10.

- Create a new login for a new user to access the MS SQL Server "LUNT" (The user should have the permission to log in the Window NT server before he/she can access the MS SQL server "LUNT". See Figure 10).

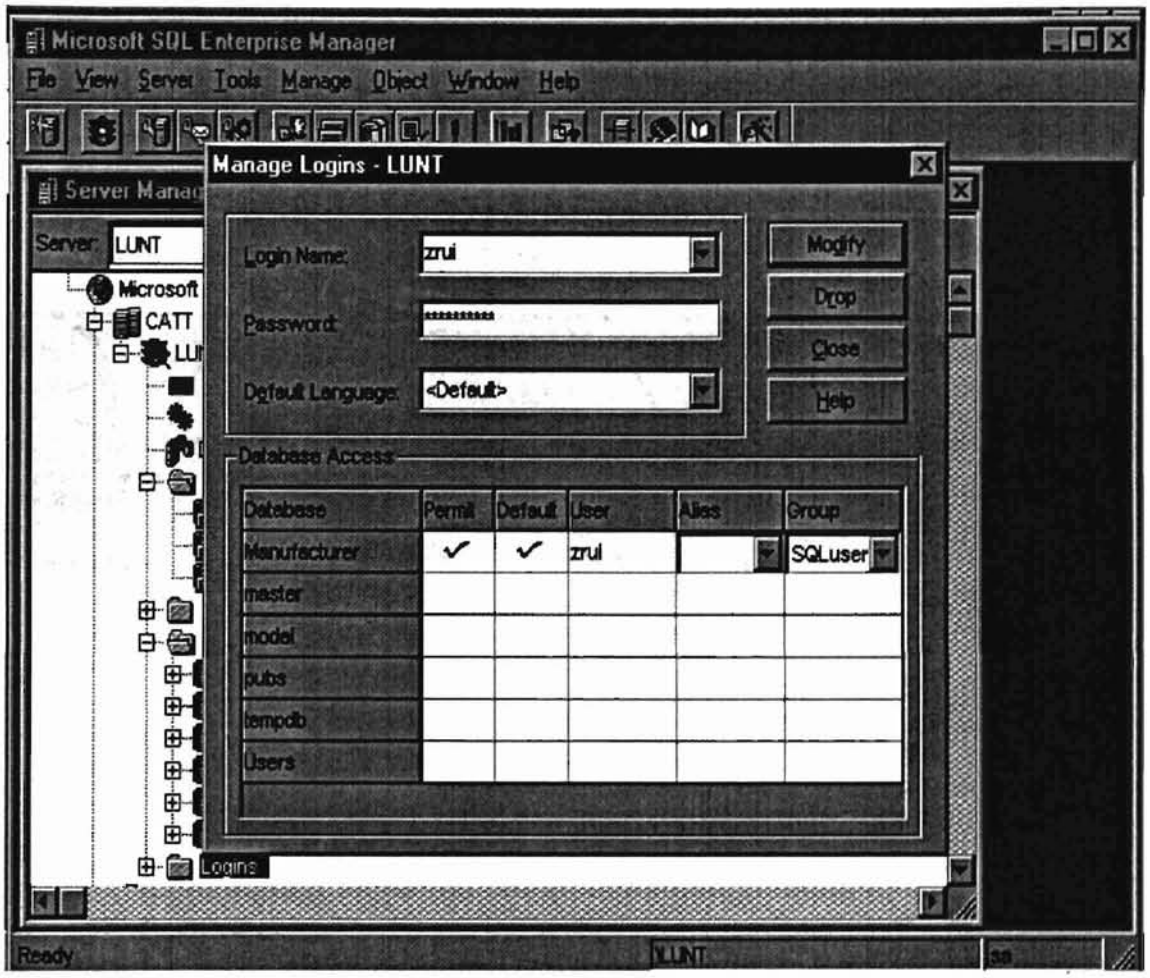

Figure 10: Add a new login for a user to access the MS SQL Server "LUNT"

Figure 10 shows the window to create a new login for a user to access the MS SQL server "LUNT". In Figure 10, click scroll bar to select a name from the "Login Name" field. Then set a check mark (✓) in the "Permit" column corresponding to

the database "Manufacturer" in the "Database Access" box, and click the "Modify" button. The system will create a new login for the user to access the MS SQL server "LUNT". The server manager can terminate a user's access to the server or some databases in the server by clicking "Drop" button at any time.

- Create a new user group or alias logins for a user to access the database "LUNT\Manufacturer" (Figures 11 and 12).

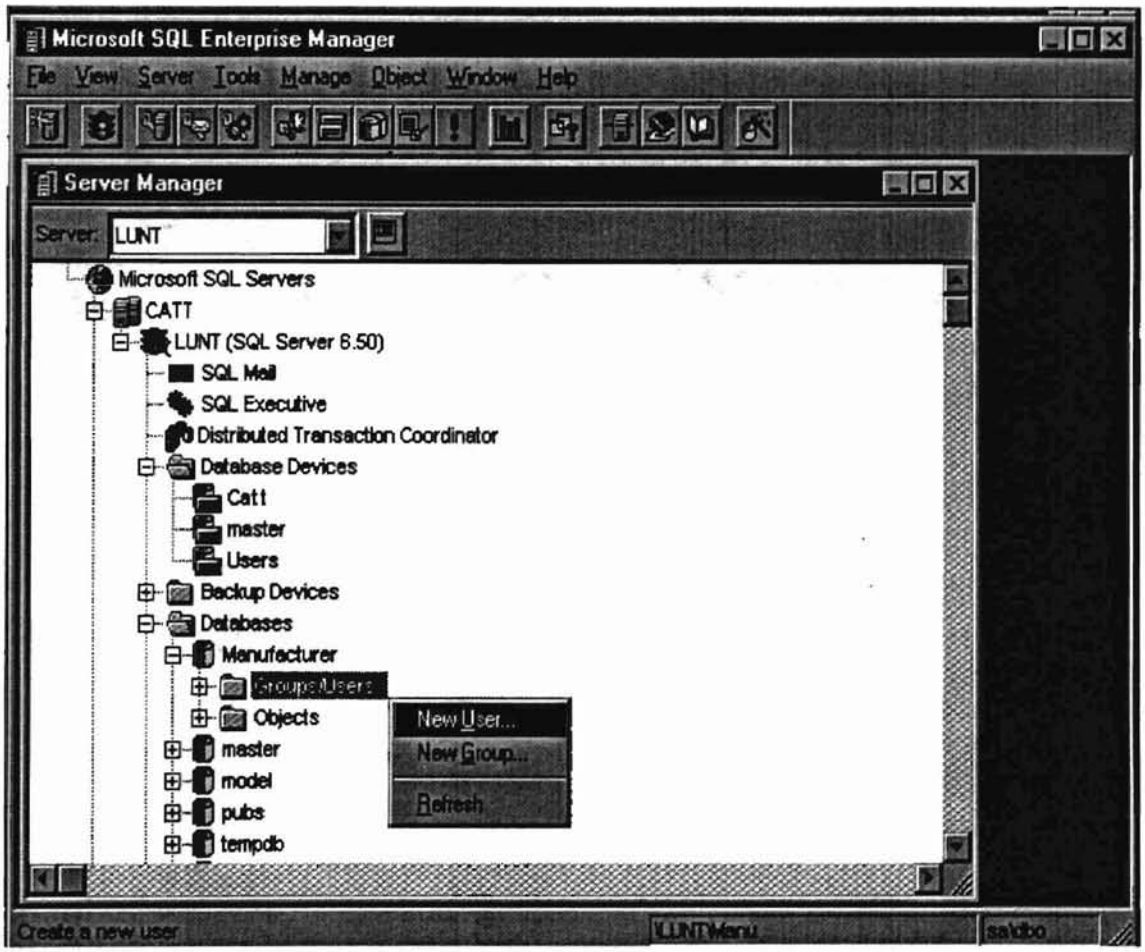

Figure 11: Open the "Groups\Users" manage

In Figure 11, move the mouse to "Groups/Users" under the database named "LUNT\Manufacturer", click the right button of a mouse, and highlight "New User" or "New Group". The system will open the "Manage Users - LUNT\Manufacturer" as shown in Figure 12 or "Manage Groups - LUNT\Manufacturer" window as shown in Figure 13.

(Note: When the server manager creates a "New User" or a "New Group", it may not appear in the list under the title of "Groups/Users" until he/she clicks "Refresh" button).

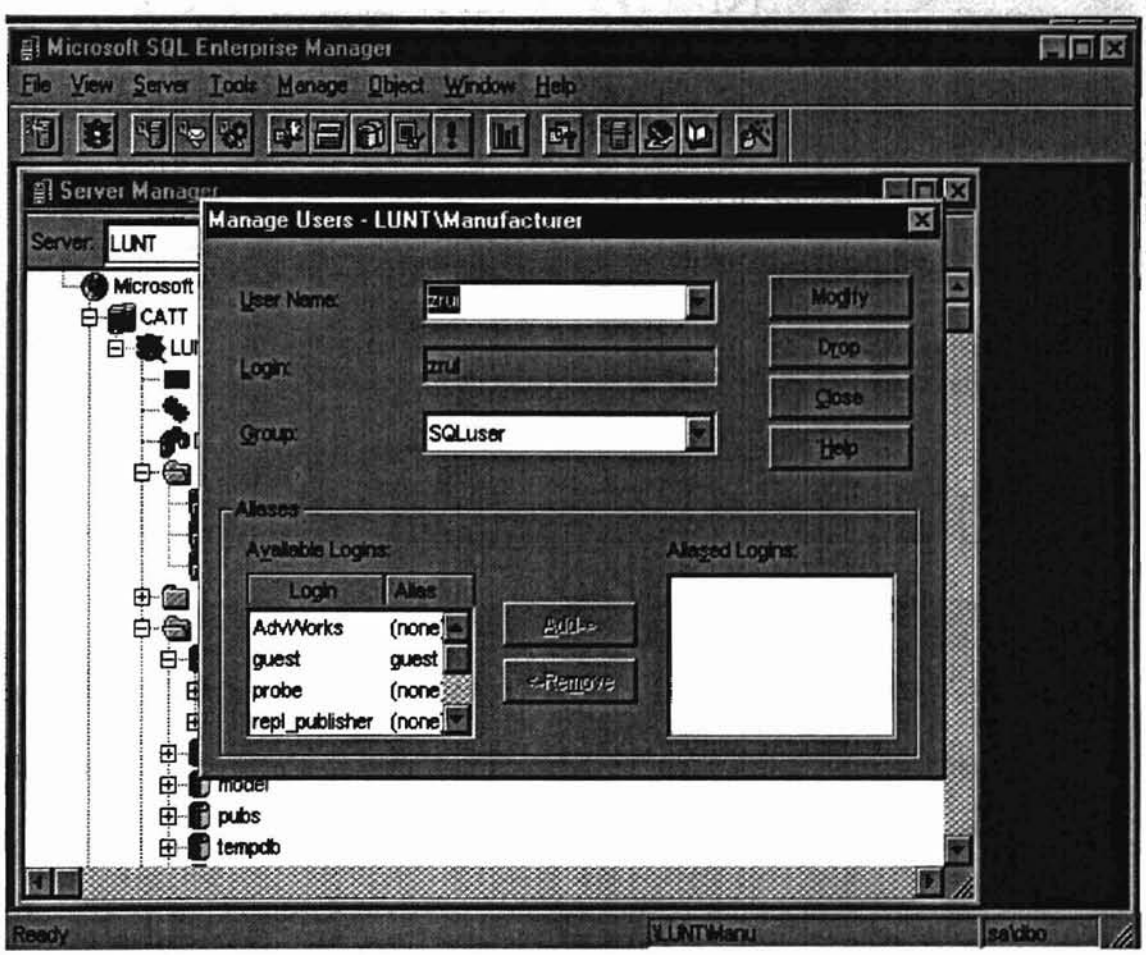

Figure 12: Manage users to access the database "LUNT\Manufacturer"

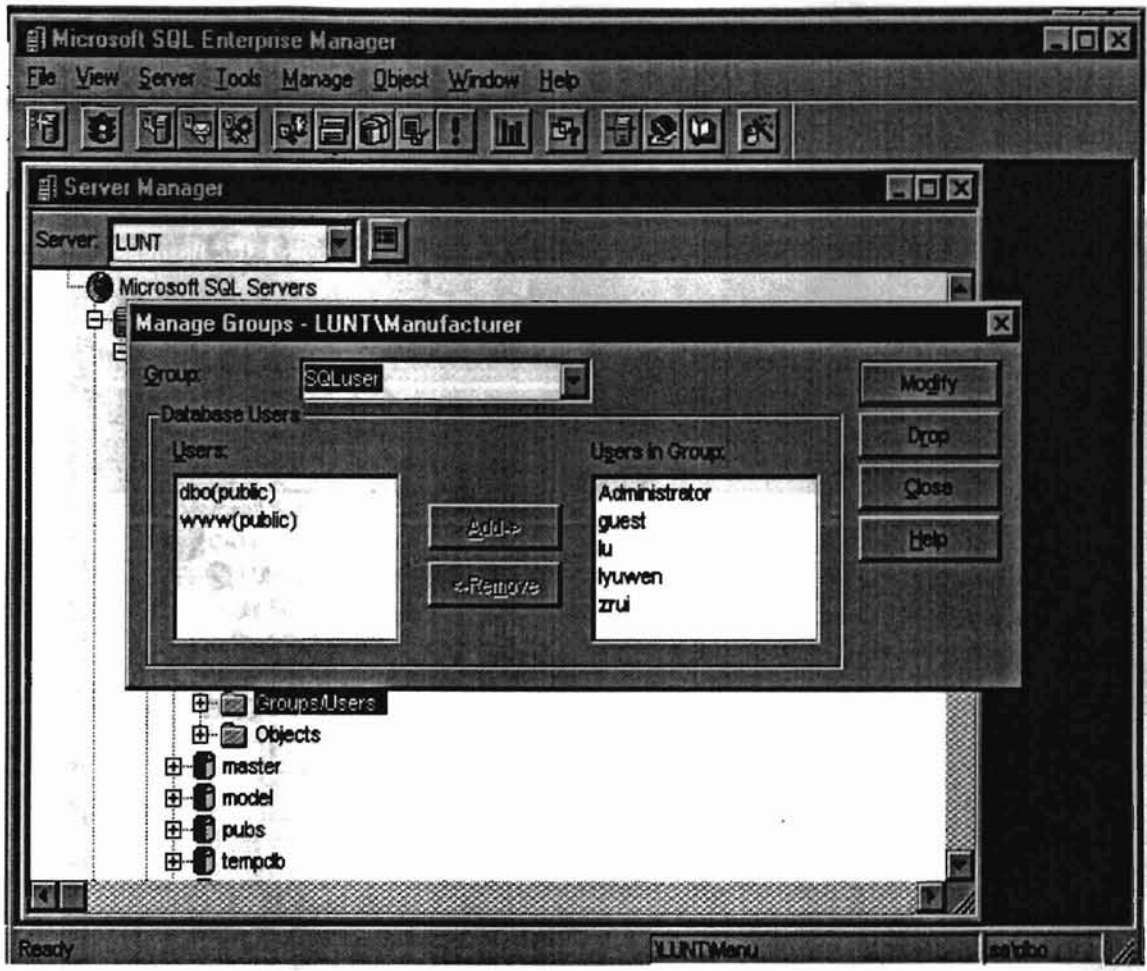

Figure 13: Manage user groups to access the database "LUNT\Manufacturer"

Figure 12 is the "Manage Users - LUNT\Manufacturer" window. Click the scroll bar in the "User Name" field to select a user name such as "zrui". Then click the scroll bar in the "Group" field to select a group name, such as "SQLuser", which the user name belongs to, and click the "Modify" button to activate the parameters in the current window. The logins created here will be used in the "Permissions" window (Figures 18 through 20) that issues permissions to a user or a user group to access, edit, delete, and update a particular table.

- Create database tables (Figures 14 through 17).

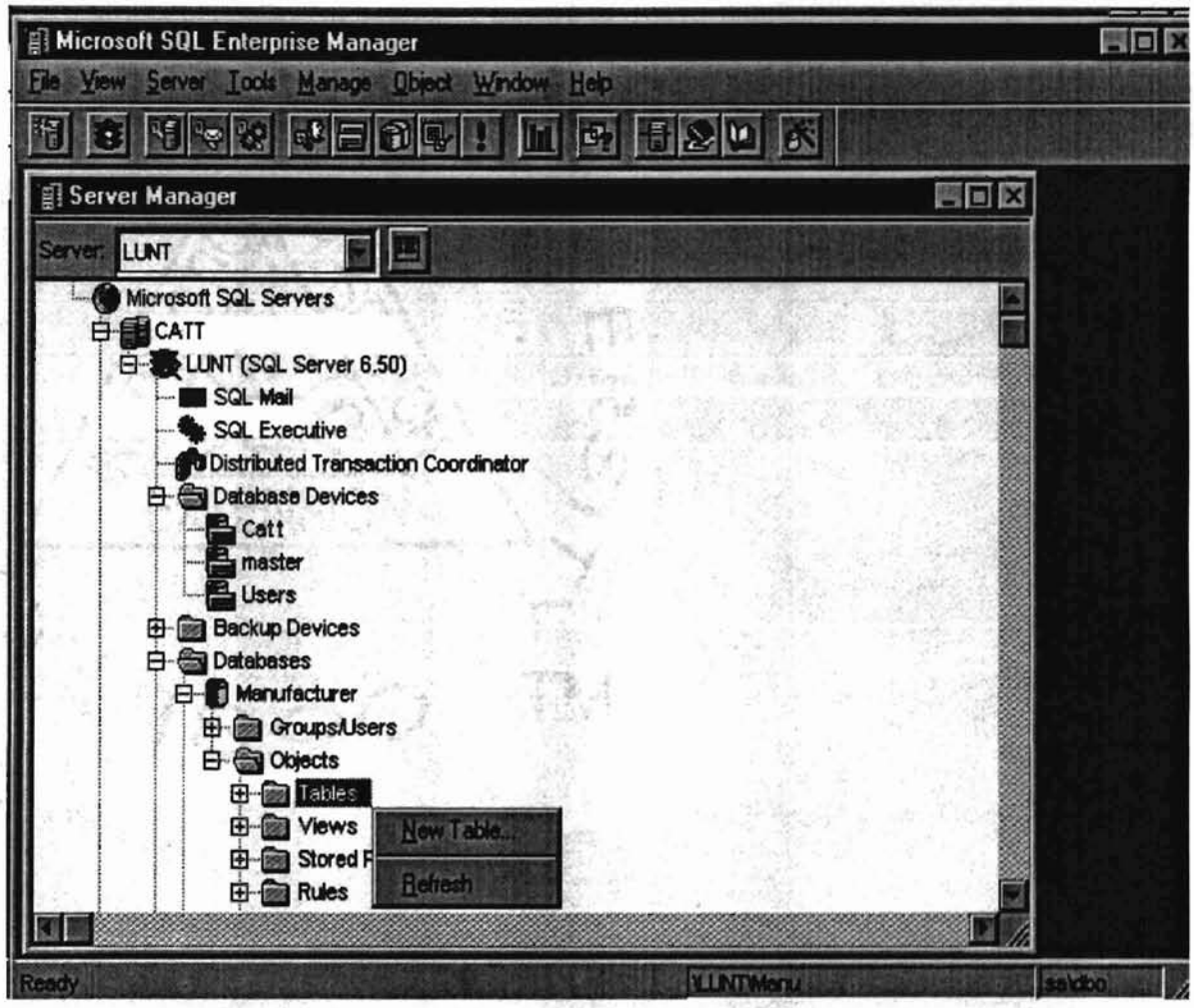

Figure 14: Open "Manage Table - LUNT\Manufacturer" window to create a new table for the database "LUNT\Manufacturer"

If the "Tables" menu does not appear in the "Server Manager" window, click the "Objects" menu under the database "LUNT\Manufacturer".

There are many methods to create the database relations (tables). One of the methods is shown in Figure 14. Move the mouse to point at "Tables" menu, click the right button of the mouse. Select "New Table..." sub-menu, the "Manage Tables - LUNT\Manufacturer" window as shown in Figure 15 will be opened.

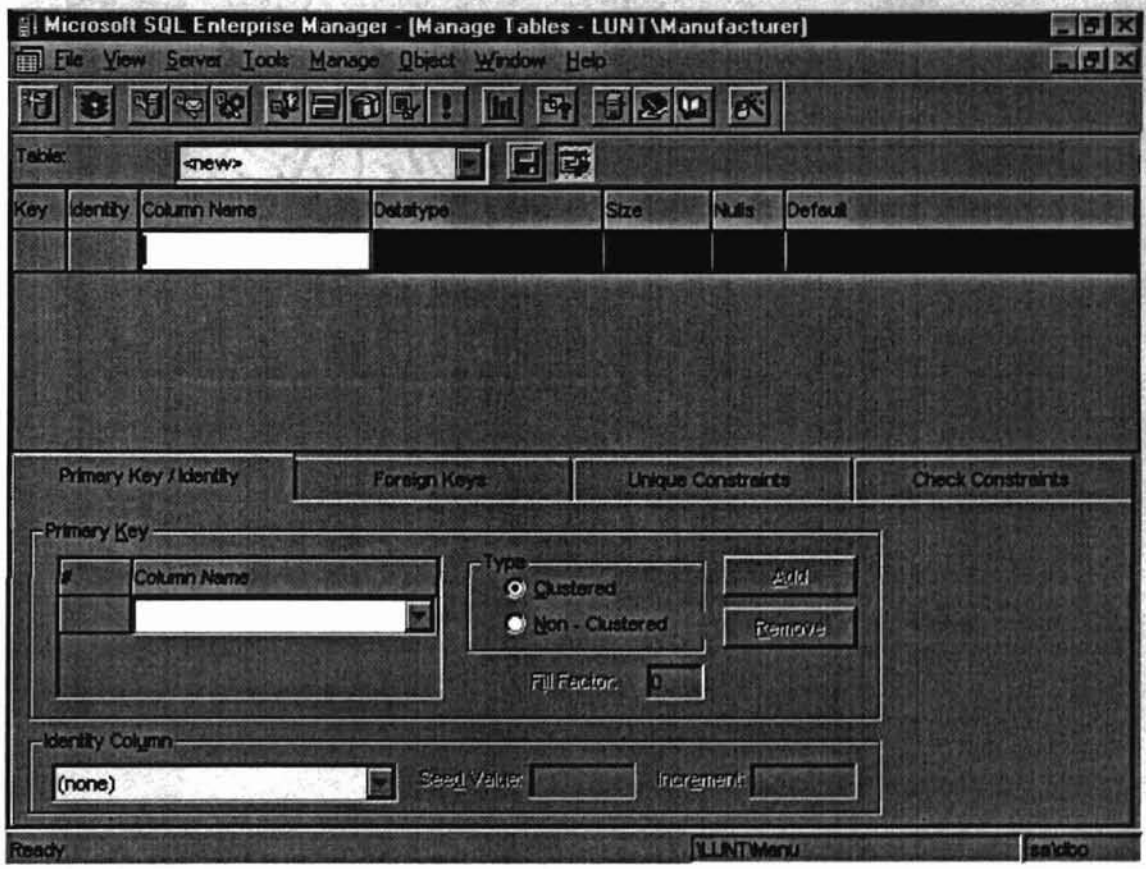

Figure 15 : "Manage Tables - LUNT\Manufacturer"

Another method is to use "SQL Query Tool" (Figure 16) which can be obtained by click the 5th button from the right in the toolbar (Figure 14) in "Microsoft SQL Enterprise Manager" window.

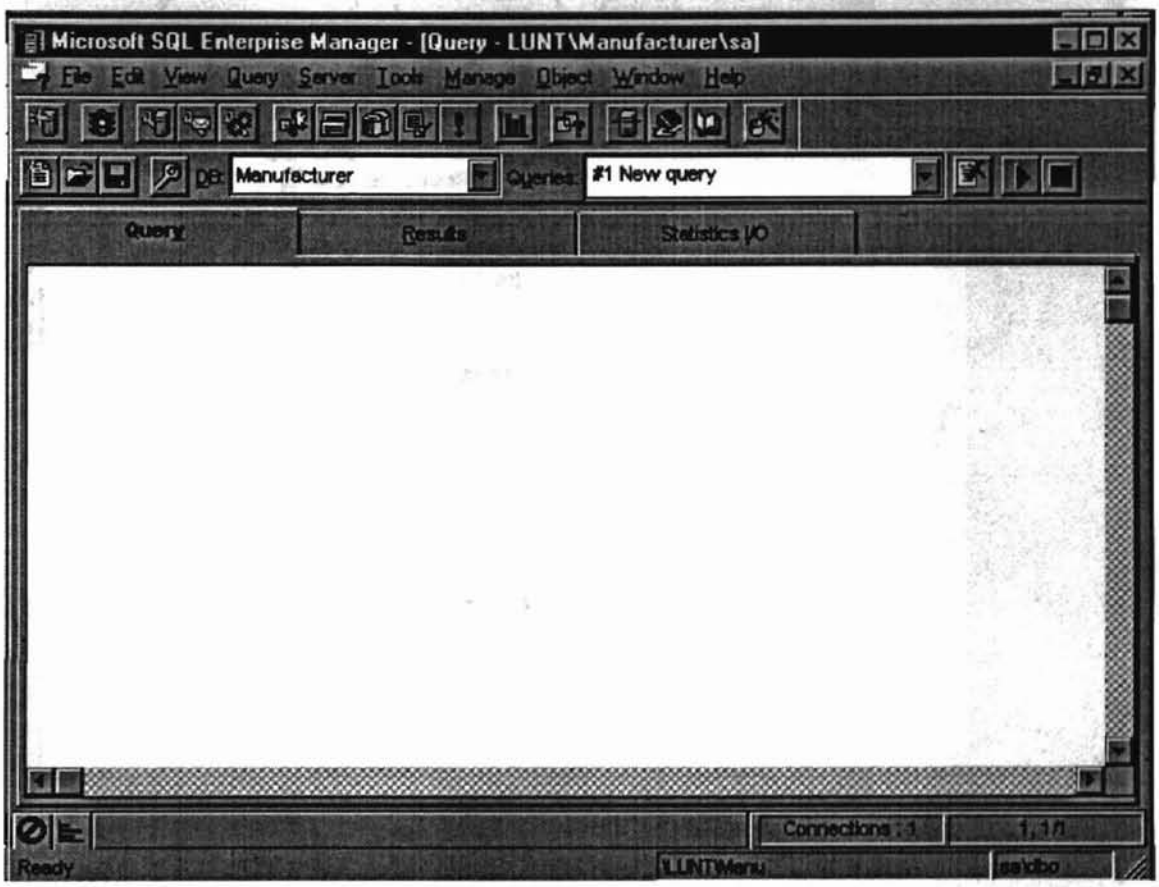

Figure 16 : SQL Query tool window

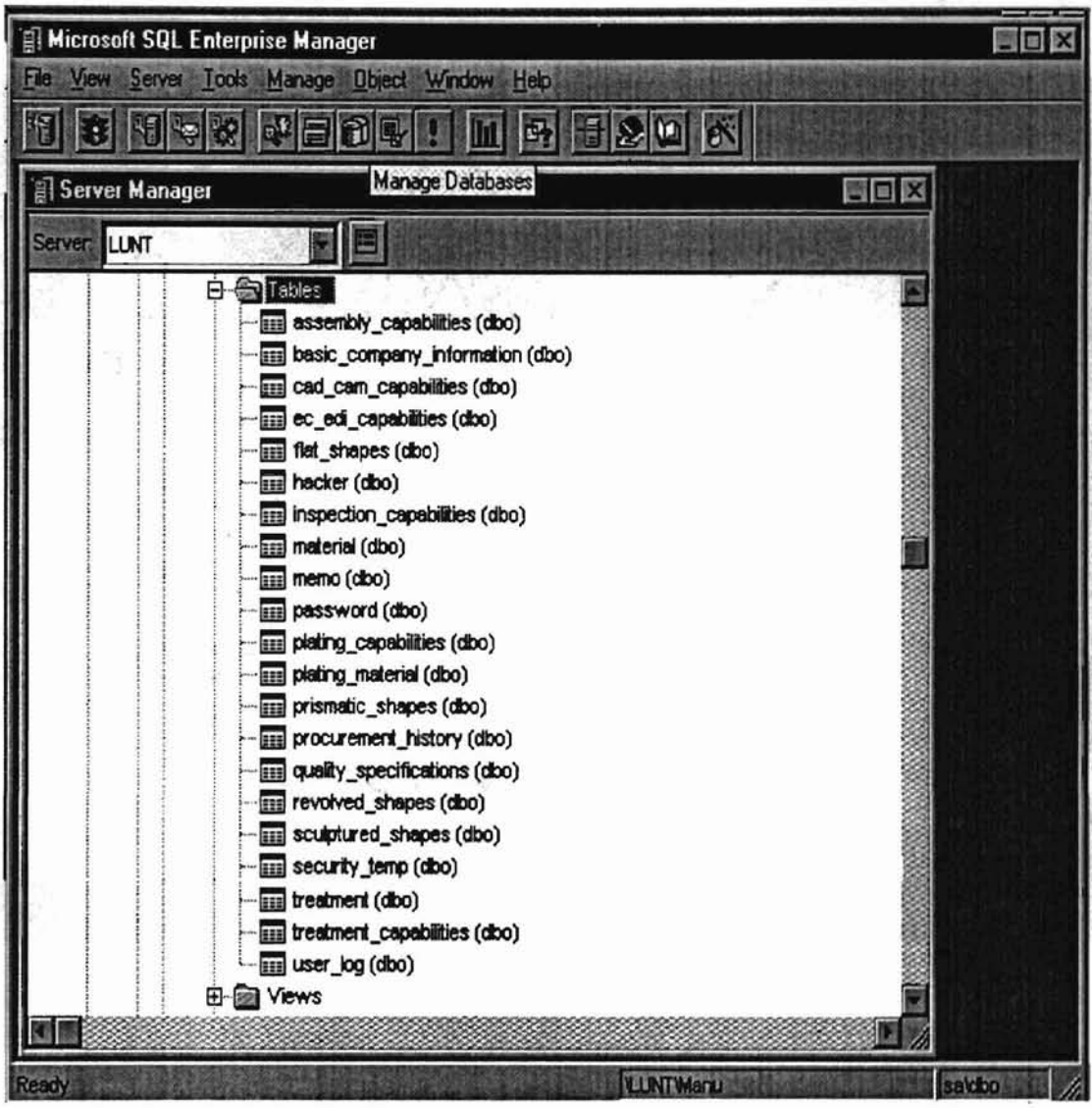

Figure 17: Tables in the database "LUNT\Manufacturer"

- Set "Object Permissions" in the database "LUNT\Manufacturer" (Figures 18 through 20).

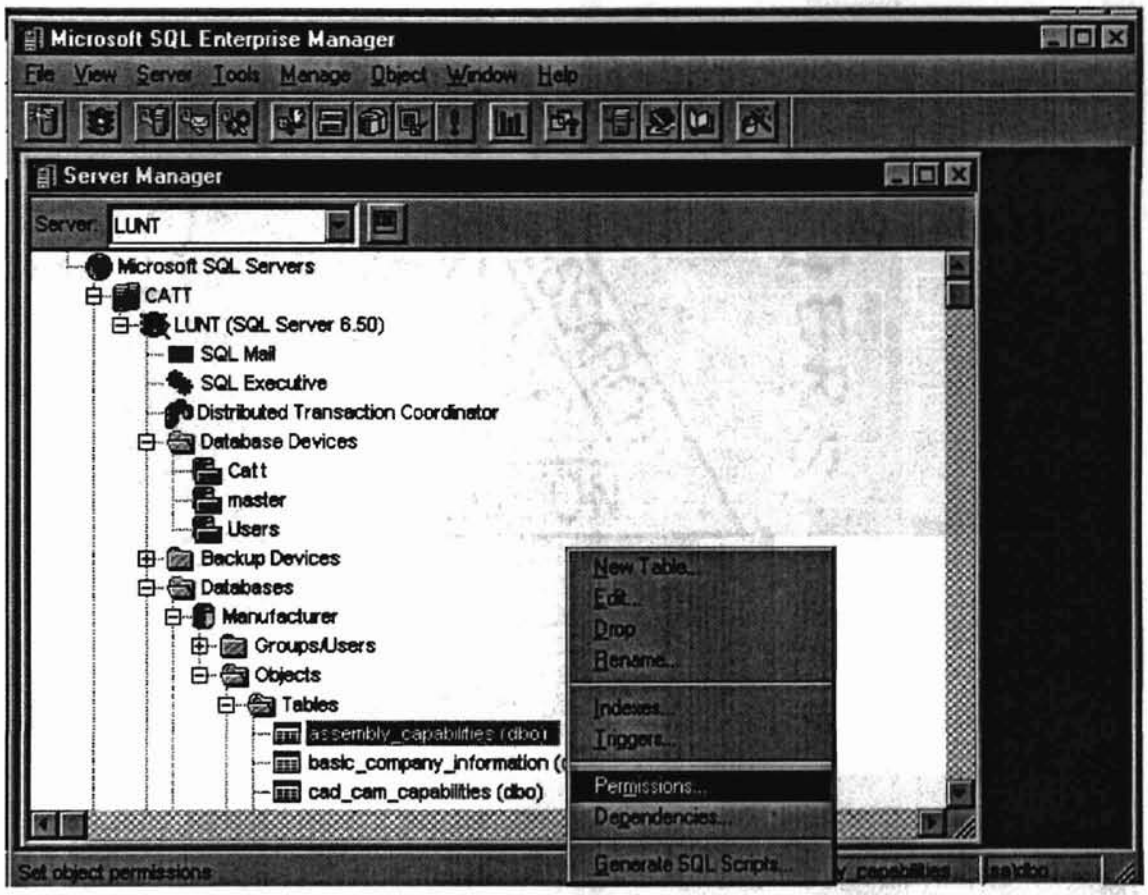

Figure 18: Database "LUNT\Manufacturer" permissions entry point

In Figure 18, move the mouse to any table name in the "Tables" field under the database "LUNT\Manufacturer", click the right button of a mouse, highlight and click the "Permissions". The "Object Permissions - LUNT\Manufacturer" window will be opened (Figures 19 and 20).

There are two tabs in the "Object Permissions - LUNT\Manufacturer" window. One is permissions "By User" (Figure 19), the other is permissions "By Object" (Figure 20).

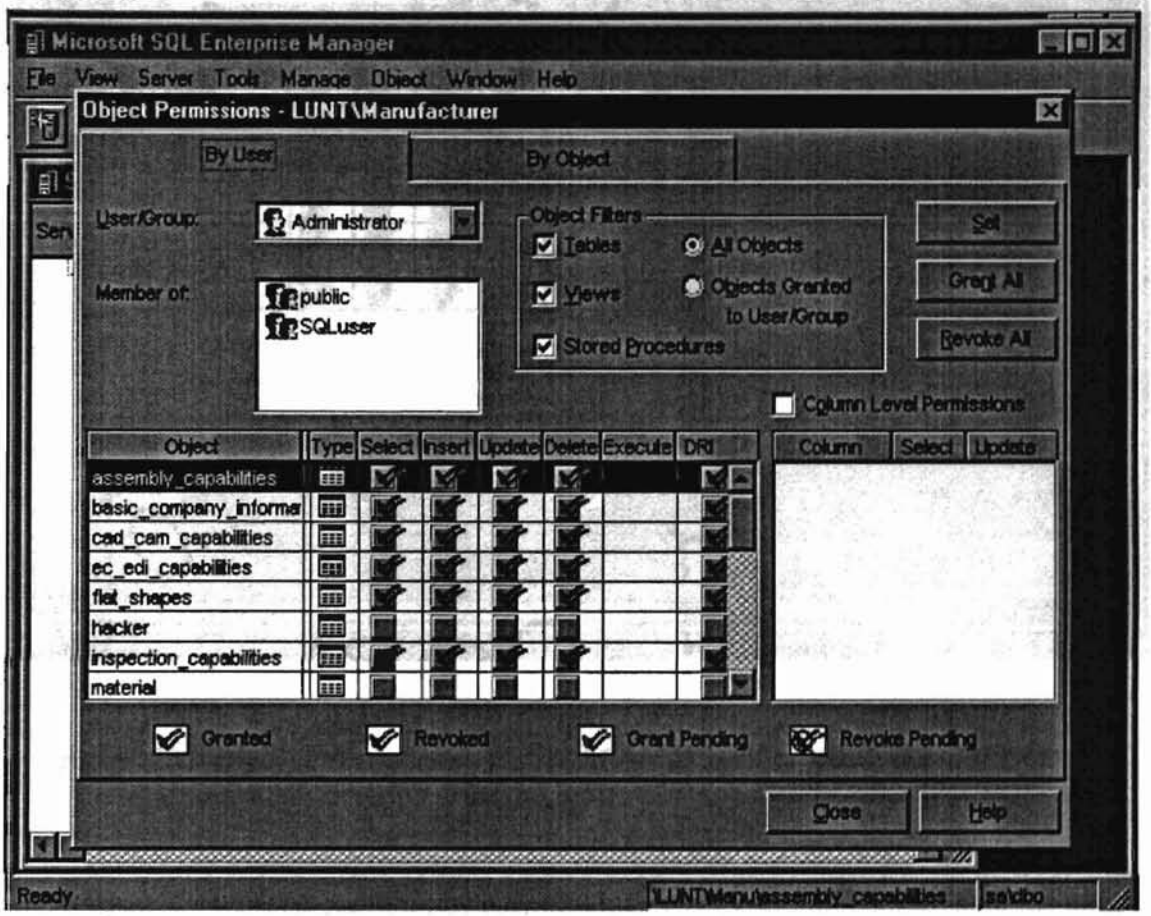

Figure 19: "Object Permissions - LUNT\Manufacturer" with the tab "By User"

Move mouse to the scroll bar in the "User/Group" field in Figure 19. Select a user or group name. Notice that there is a table in the lower left portion of the window. In this table, a check mark in a box in the table means the user/group has the permission for an action (select, insert, etc.) granted for a particular relation in the "Object" column.

describes setting up Database Source Name (DSN), which is

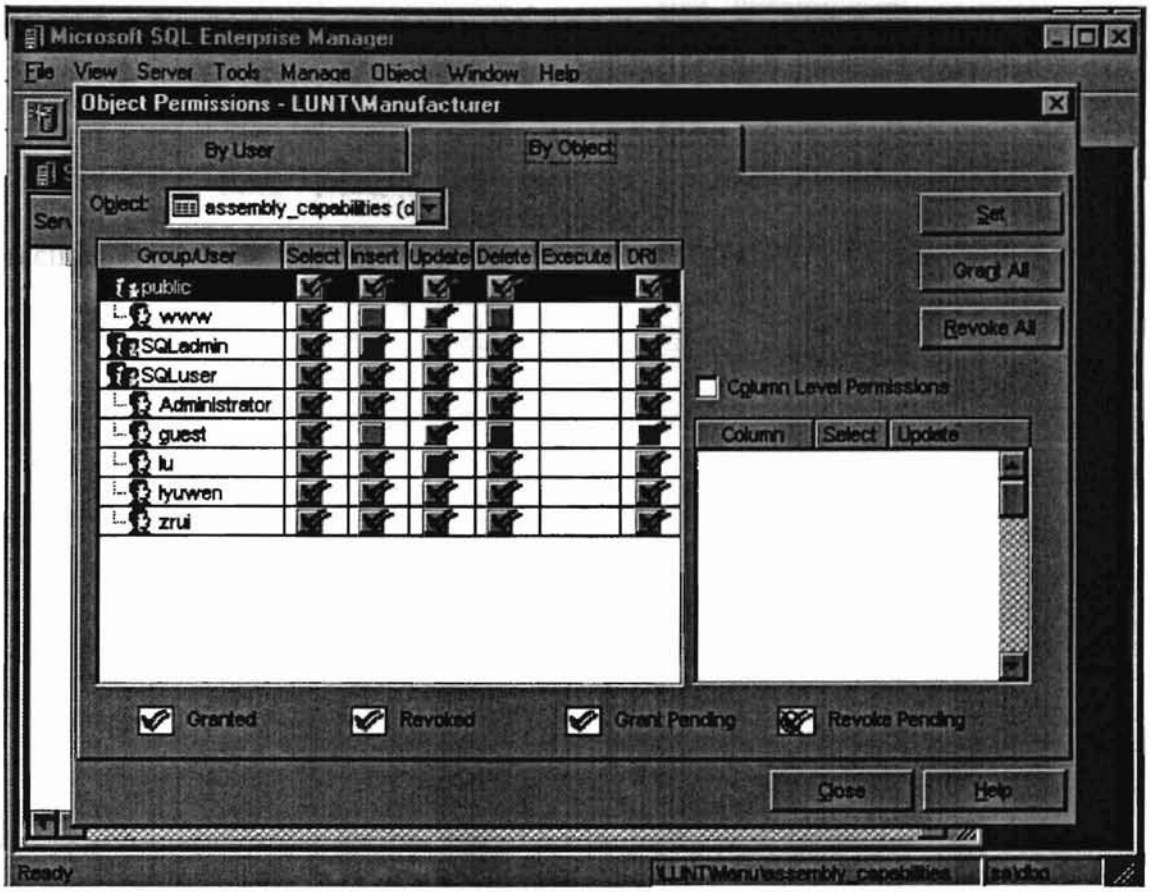

Figure 20: "Object Permissions - LUNT\Manufacturer" with the tab "By Object"

Figure 20 shows the "Object Permissions - LUNT\Manufacturer" window with the tab "By Object" activated. Click the scroll bar in the "Object" field to select an object (tables name in the database "LUNT\Manufacturer), then put a check mark (☑) in the table for a group or a user. It means the user or the group in the "Group/User" column has permission for a particular action (select, insert, update, etc.) for the selected table.

The next section decribes setting up Database Source Name (DSN), which is "CattManu" in this case. Any DSN needs two parameters, a ODBC driver, and a database name. The ODBC driver has been included in our "SQL Server". We use the "Manufacturer" (LUNT\Manufacturer) as the database name which is created in this section.

Setting up a Database Source Name (DSN) for the Database "LUNT\Manufacturer"

- The ODBC manager is a dynamically linked library (ODBC32.DLL) that loads ODBC drivers and provides a single entry point to ODBC functions for different drivers. We have set up the database "Manufacturer" in MS SQL Server "LUNT" (LUNT\Manufacturer), and need to create a Database Source Name (DSN) "CattManu" for the database "LUNT\Manufacturer". We will use DSN "CattManu" in the next section to access and update the database "LUNT\Manufacturer".

The procedures for setting up DSN are as follows:

Open the "Control Panel" window under "Setting" menu of Windows 95. Then click the "32 ODBC" icon, The "ODBC Data Source Administrator" window will be opened. The window shows all current ODBC data source names on the system (Figure 21). "ODBC Data Source Administrator" (ODBCAD32.EXE) allows a programmer to add, delete, or configure a DSN by click on Add, Remove or Configure button respectively.

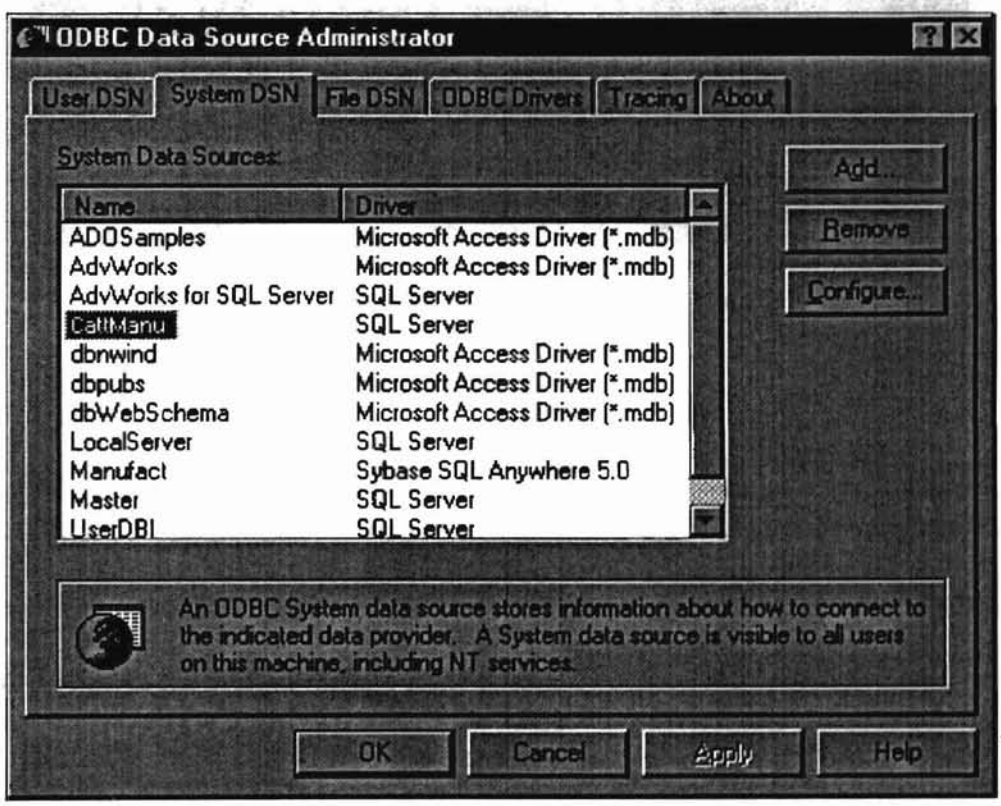

Figure 21: The ODBC Data Source Administrator

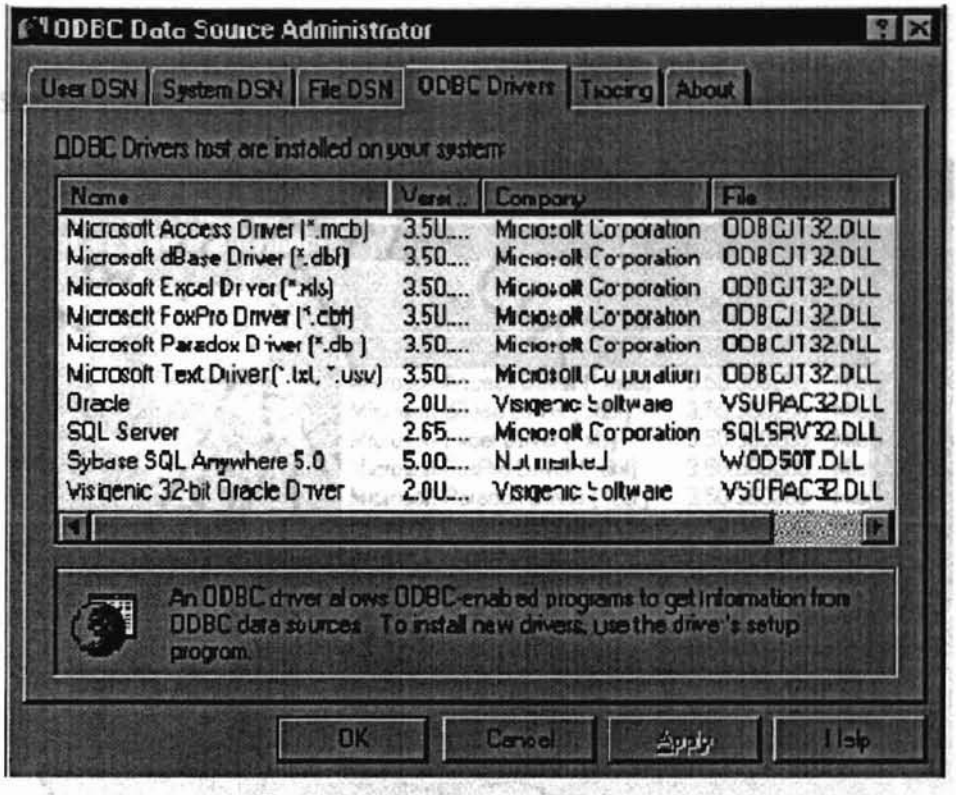

Figure 22: The ODBC Driver available

Click on "ODBC Drivers" tab in the "ODBC Data Source Administrator" window; The window will open a driver list window which shows all currently installed ODBC drivers in the system (Figure 22).

Now, create an ODBC Data Source Name (DSN) "CattManu" for the database "LUNT\Manufacturer". Click "System DSN" tab, make sure that the ODBC Data Source Administrator" window is back to the status shown in Figure 21. Since a "System DSN" is a data source name visible to all users, the web server will be able to find it if a code points to the DSN. System data sources are available only in the 32-bit ODBC administrator that comes with MS Windows NT Server 4.0.

Figures 23 and 24 illustrate how to set up a new data source "CattManu" using "SQL Server" driver to connect the database "LUNT\Manufacturer".

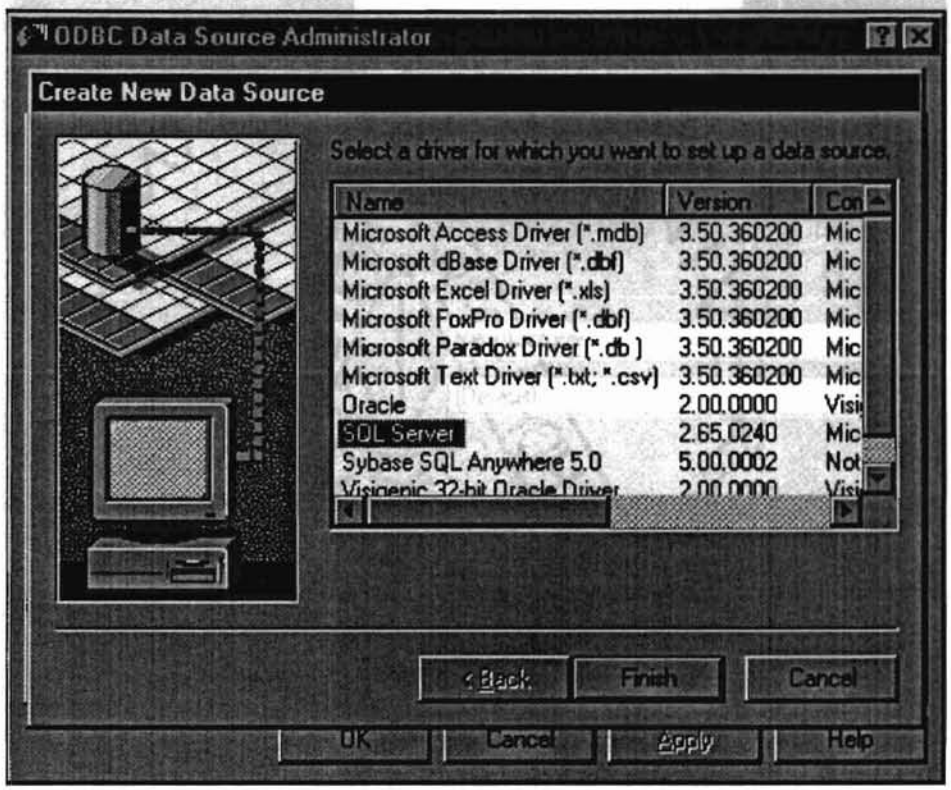

Figure 23: Select a driver when create a new data source

Click the "Add" button in Figure 21 to add a new data source name. The window shown in Figure 23 will be opened. It shows all of the drivers available. If the MS SQL driver is not in the list, the user will need to install it. Do a customized installation and add the MS SQL driver.

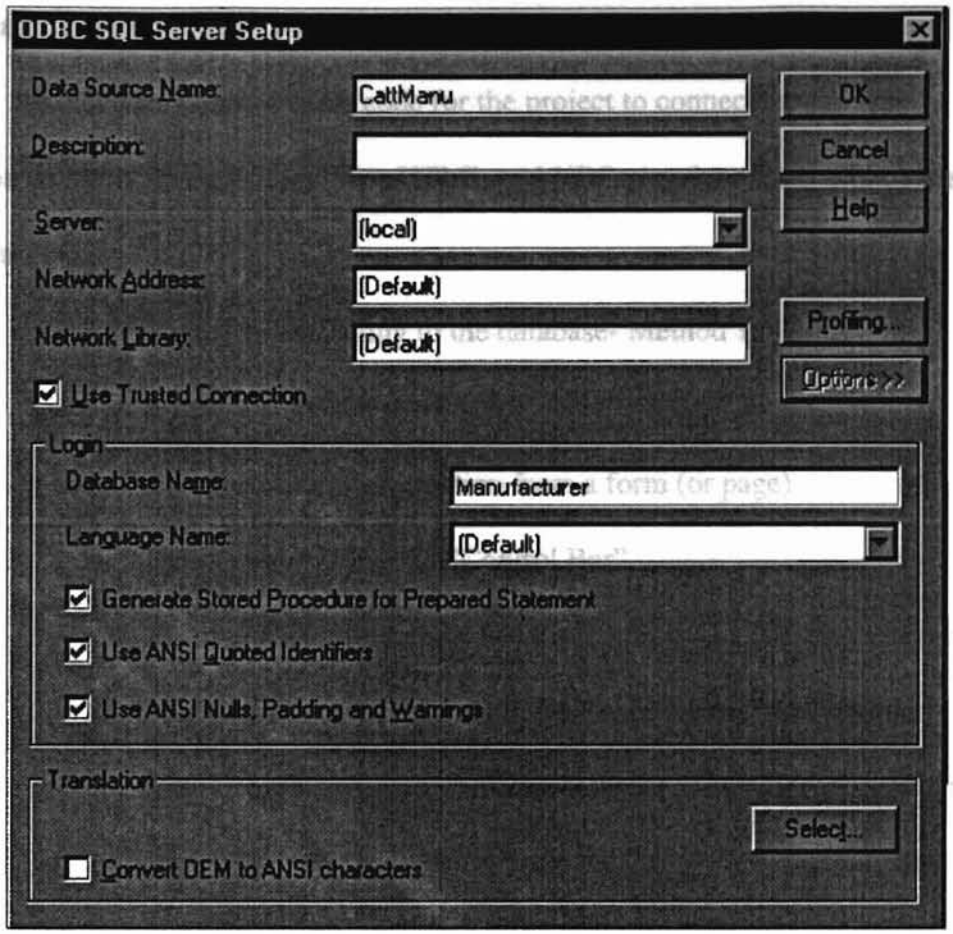

Figure 24: The ODBC SQL Server Setup

Choose the "SQL server" driver in Figure 23. The "ODBC SQL Server Setup" window will be opened (Figure 24). In Figure 24, define a Data Source Name (DSN) "CattManu". Add a description. Then type in "Manufacturer" in the "Database Name" field for the database "LUNT\Manufacturer". Click OK, and the ODBC data source name (DSN) "CattManu" will be created by the "ODBC DSN Administrator".

Implementation details

This section shows a basic code for the project to connect and update the database using ASP which combines HTML and VBScript. It consists eight parts. They are :

Parts 1 : Connecting to the database- Method 1

Parts 2 : Connecting to the database- Method 2

Part 3 : Getting information from a form (or page)

Part 4 : Generating the "Control Bar"

Part 5 : Inserting records into the database "LUNT\Manufacturer"

Part 6 : Editing records in the database "LUNT\Manufacturer"

Part 7 : Updating records in the database "LUNT\Manufacturer"

Part 8 :Deleting records from the database "LUNT\Manufacturer"

1. Connecting to the Database - Method 1

We have developed a database "LUNT\Manufacturer" in section IV and added a DSN entry "CattManu" in section V in order that the database can be accessed using ODBC.

One major problem that will impede the user of such a system is the ability to find the relevant information needed to answer a specific query and to update the database. One could design a browser that would allow the user to explore the search space of available information and then have the user manually put together the information found. This approach might require a great deal of user time and effort to find the desired information. Another approach is to use SQL in SQL server to determine which relations should be accessed to answer a given query. We need to create a form to add, delete, and edit information for the database. We may add one line to those forms that gives users an additional link to another script that displays other information.

The ASP offers two methods to access a database. An example is given in Table3 for the first method. Table 3 illustrates a connection to a database with "CattManu" as the DSN, obtaining a record set based on a SQL query. Once the script has done with the data, the record set and the connection to the database are closed. Each access to the database would be connected to the DSN entry "CATTManu" (Lines 1 and 2 in Table 3); once the connection has been established, SQL statements can be used to manipulate data (lines 3 through 9 in Table 3); once completed, all related objects are closed (lines 10 and 11 in Table 3). This method is used in the project.

Notation used in the Table 3:

In Table 3, the "Memo" is a table in the database "LUNT\Manufacturer";

line 3 is to assign a "Record Set" (RS) to a string variable "RS";

RS.EOF means end of the record set;

RS.MoveNext means moving the record set to the next record;

RS.Close means closing the record set;

Cattdb.Close means closing the object.

```
<%
        Set CATTdb= Server.CreateObject("ADODB.Connection")
        CATTdb.Open "CattManu"
        Set RS = CATTdb.Execute("SELECT * FROM Memo")
        <Select Name = "Company" Size=1>
        Do While Not RS.EOF
           <Option><% = RS("Company_Name") %> </Option>
        RS.MoveNext
        Loop
        </Select>
        RS.Close
        CATTdb.Close
%>
```

Table 3. Connecting to a database - method 1

2. **Connecting to the Database - Method 2**

The other option is to maintain a connection throughout the user's session. The connection is closed when the session ends. This is controlled by the Global.asa file. Each ASP-Based internet application can have one global.asa file located in the root directory of the application. The global.asa has four events - Application-Start, Session-Start, Application-End, and Session-End. The connection to the database for the session is in the Session-Start event, with Session-End being the event used to close the connection (Table 4). The language or script being used is VBScript. The .ASP files run on the server side.

As indicated in Table 4, "CATTdb" is connected to the database "LUNT\Manufacturer" that can be used throughout the session to the DSN entry "CattManu".

```
SUB Session_OnStart
                    CATTdb.Open "CattManu"
                                '---- Open ADO connection to database
END SUB
```

Table 4. Connecting to a database - method 2

The first post-startup request is made to the web server for any *.asp file in an application that causes the Global.asa to be read. So at the moment a request is made to any *.asp in the directory in which the internet application is stored, a connection is established with the DSN "CattManu". Table 5 shows the default default.asp processed.

```
CATTdb.Open "CattManu"
```

Table 5: Connect string in global.asa

3. Getting Information from the Form

In ASP based applications the programming logic, variables and HTML can be maintained in a single file. Commonly used functions across an ASP application can be in one file, which can be included in different pages using the "include" statement such as the command "<!-- #include FILE="filename" -->. With regards to the logic of the example we have used a state space model to determine the state of the ASP page - e.g., an addition, deletion or update taking place or not.

The information of the current state of the page is dictated by the contents of the form element named "Action". The value of the element "Action" is obtained from the form in VBScripts with the statement in Table 6.

```
Action = Request.Form("Action")
```

Table 6: Getting a form element value.

4. Generating the Control Bar

Depending upon the value of "Action", different control bars (Add, Edit, Delete etc.) can be displayed and different HTML can also be sent back to the

client. Which controls need to be displayed are handled by the function ControlBar depending upon the current value of the variable "Action". For example, the code in Table 7 indicates that if the "Action" is blank, or has the variable "OK", or has the variable "CANCEL", the controls bar content Add, Edit and Delete will be displayed. The result of Table 7 is shown in Figure 34 (See Page 62).

```
if Action = "" or Action= "OK" or Action = "CANCEL" then
        CBar=CBar & " < td valign=top><input type=submit name=Action value=Add></td>"
        CBar=CBar & "< td valign=top><input type=submit name=Action value=Edit></td>"
        Cbar=CBar & "<td valign=top><input type=submit name=Action value=Delete></td>"
end if
```

Table 7: Generating the control bar

5. Inserting Records into the Database

The function Check makes sure that values have been entered in the respective fields before being added to the database. If all the information has been filled out, the information is inserted into the database using the SQL statement. An example is given in Table 8, which inserts the values entered in the fields of the form into the database (insert a record with CodeKey "100001" and Memo "AB").

```
sqlstr= "INSERT INTO Memo "
                sqlstr= sqlstr & "(CodeKey, Memo )VALUES "
                sqlstr= sqlstr & "('100001','AB')"
```

Table 8: Inserting a record.

6. Editing a Record

To edit a specific user's information we created two combo boxes. The first combo box has up to fourteen forms for each particular company. The second combo box has all of the company names in the database, as shown in Figure 34. The second box is created by querying the database for all the companies in the database using the SQL statement in Table 9.

```
sqlstr="SELECT * FROM Basic_Company_Information  ORDER BY CodeKey ASC"
                    set rsUsers=CATTdb.Execute(sqlstr)
```

Table 9: Selecting records

7. Updating a Record

An example is given in Table 10 using SQL statement. The value of the Memo will be changed from old value to "New Memo Value".

```
Memo = "New Memo Value"
sqlstr= "UPDATE Memo "
        sqlstr= sqlstr & "SET CodeKey ='100001', "
        sqlstr= sqlstr & "Memo ='New Memo Value', "
        sqlstr= sqlstr & " WHERE CodeKey='100001';"
```

Table 10: Updating a record

8. Deleting a Record

The ASP logic can be embedded in the HTML to generate HTML according to certain conditions. In this case the HTML code is controlled by the value of Action.

In order to delete a record, the CodeKey (a primary key) of the record to be deleted is needed. The HTML code is generated using logic condition on the value in the variable action. Since the value of Action is "DEL" HTML code is generated with information about the CodeKey that can be used by the "REMOVE" method, that does the actual deletion. An example is given in Table 11.

```
<%
        CodeKey = "100001"
        if Action="EDI" or Action="DEL" then       'Edit Mode
                do until rs.eof
                        strUsers=strUser & "
                        <tr><td valign=top>
                        <select name="CodeKey">
                                <%=CodeKey%>
                        </select>
                        </tr></td>"
        <% end if >
```

Table 11: Deleting a record with a CodeKey value "100001"

CHAPTER IV

Results and Discussion

The database "Manufacturing Resource Matrix" was used in this project. A authorized web user can view, search, update a particular information for a manufacturer. The interface and running results are listed below.

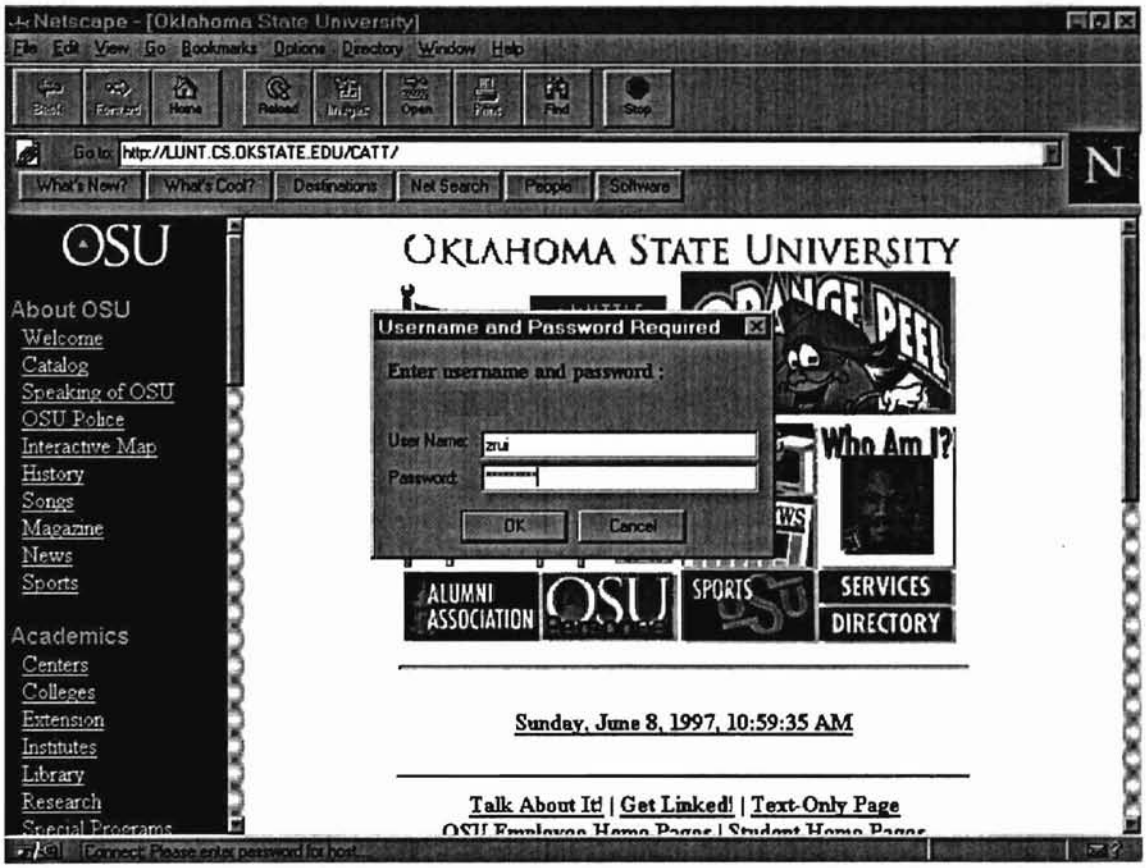

Figure 25 : A window for the user name and password

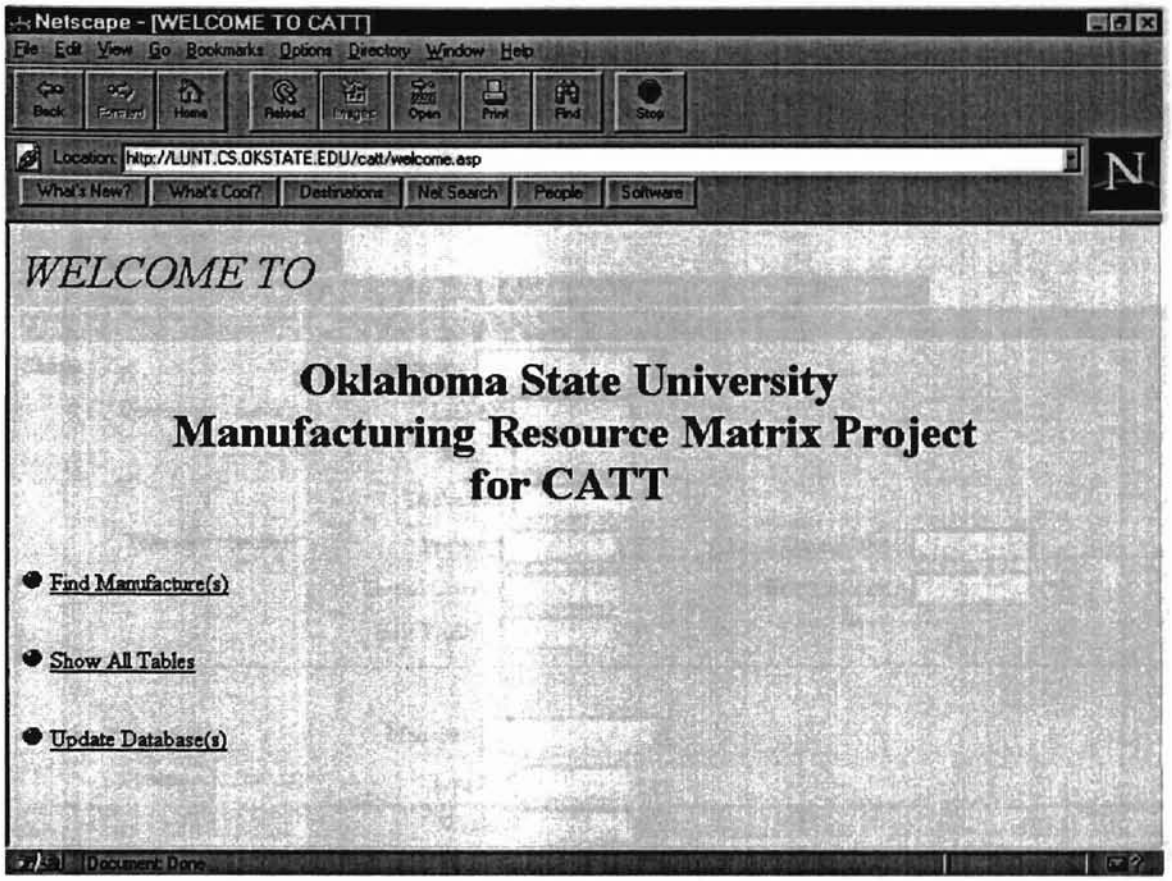

Figure 26 : Welcome and menu page

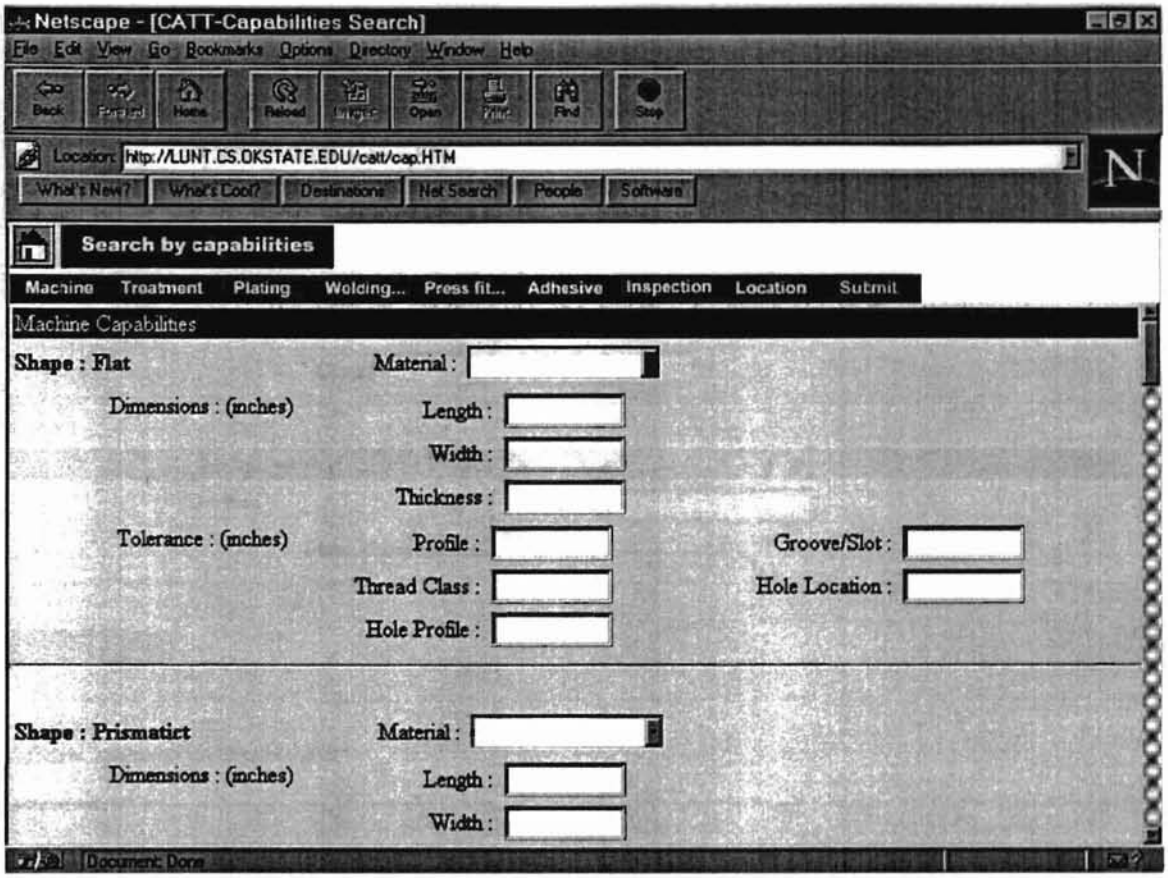

Figure 27 : Capabilities parameters for search company information

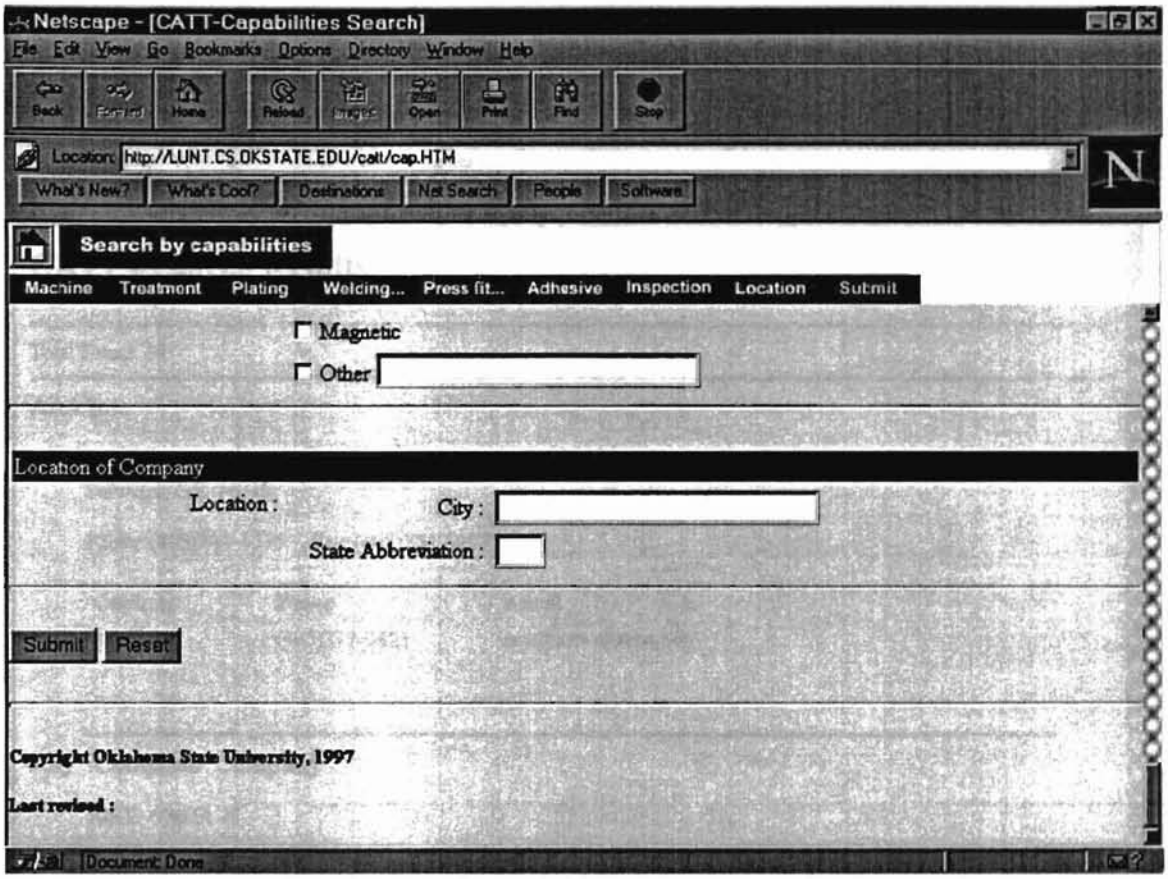

Figure 28 : A form of parameters for querying the database

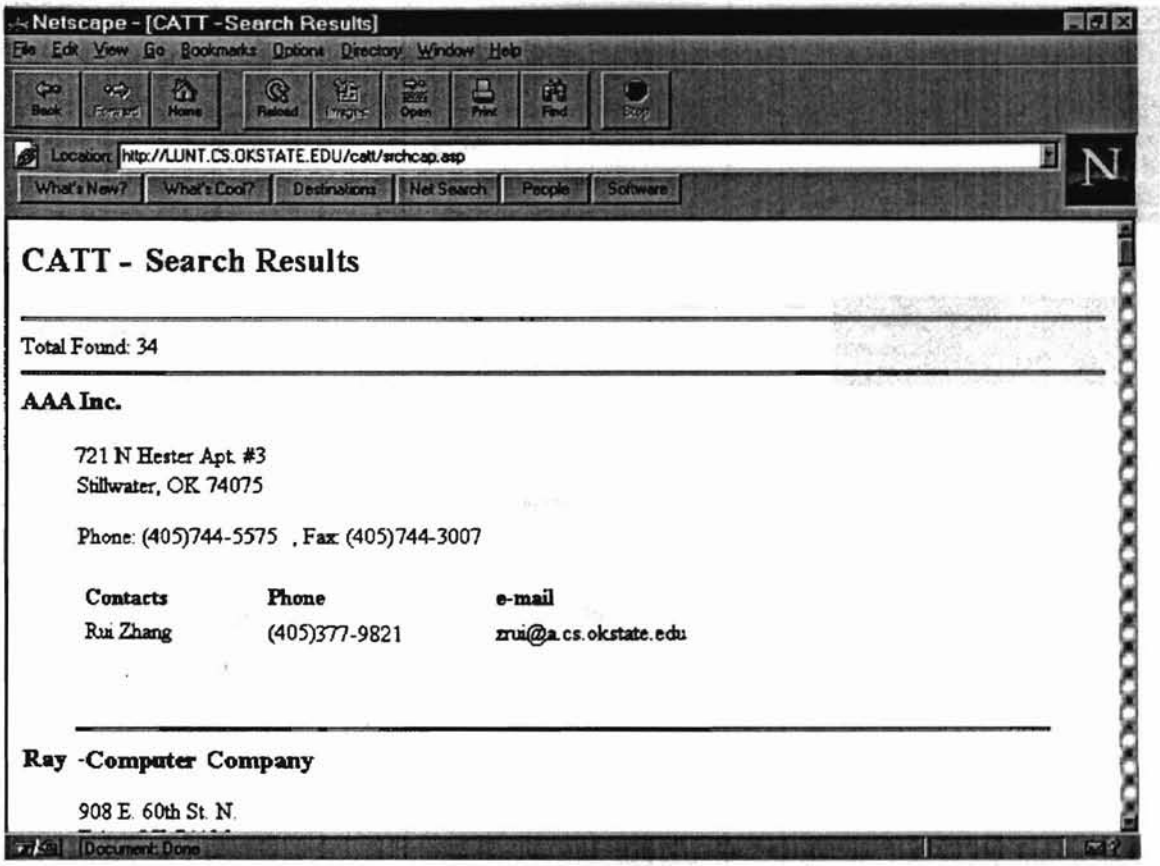

Figure 29 : An example of search results

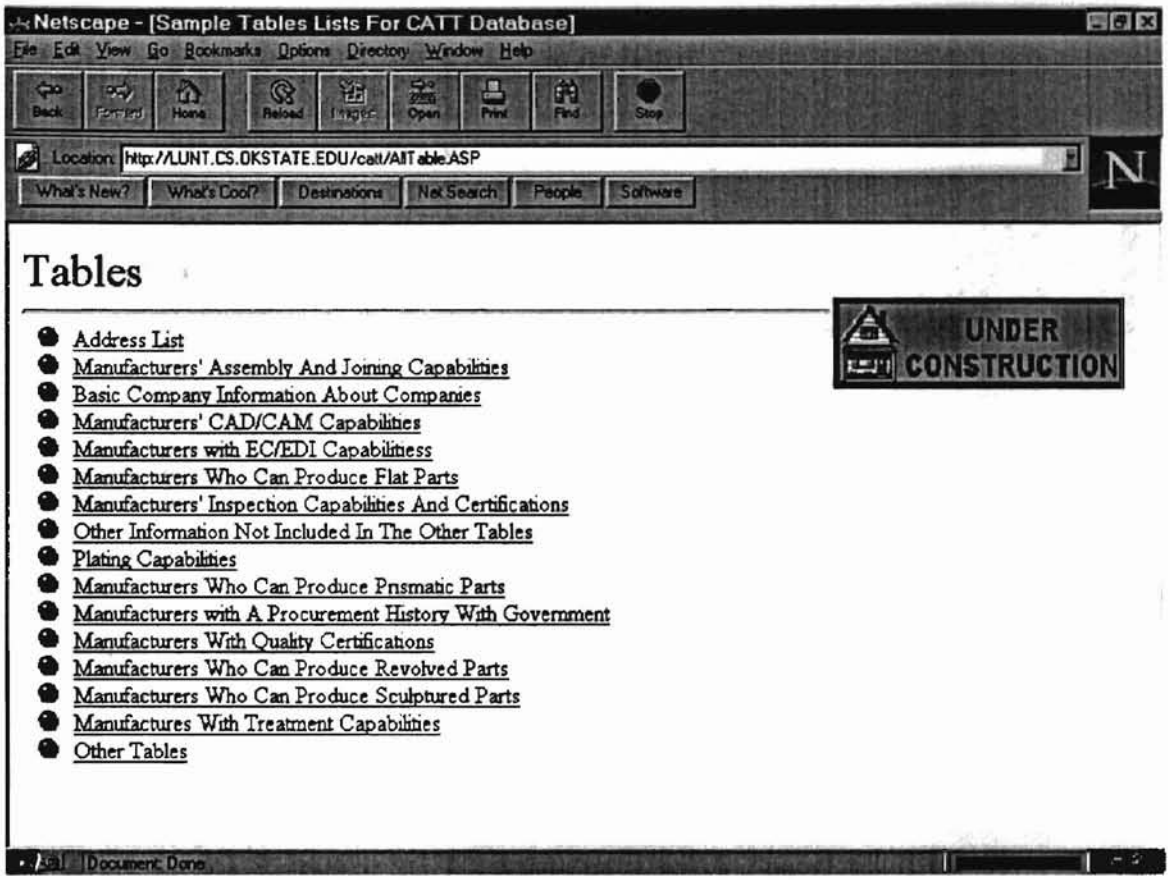

Figure 30 : Names of tables (relations) in the relational database

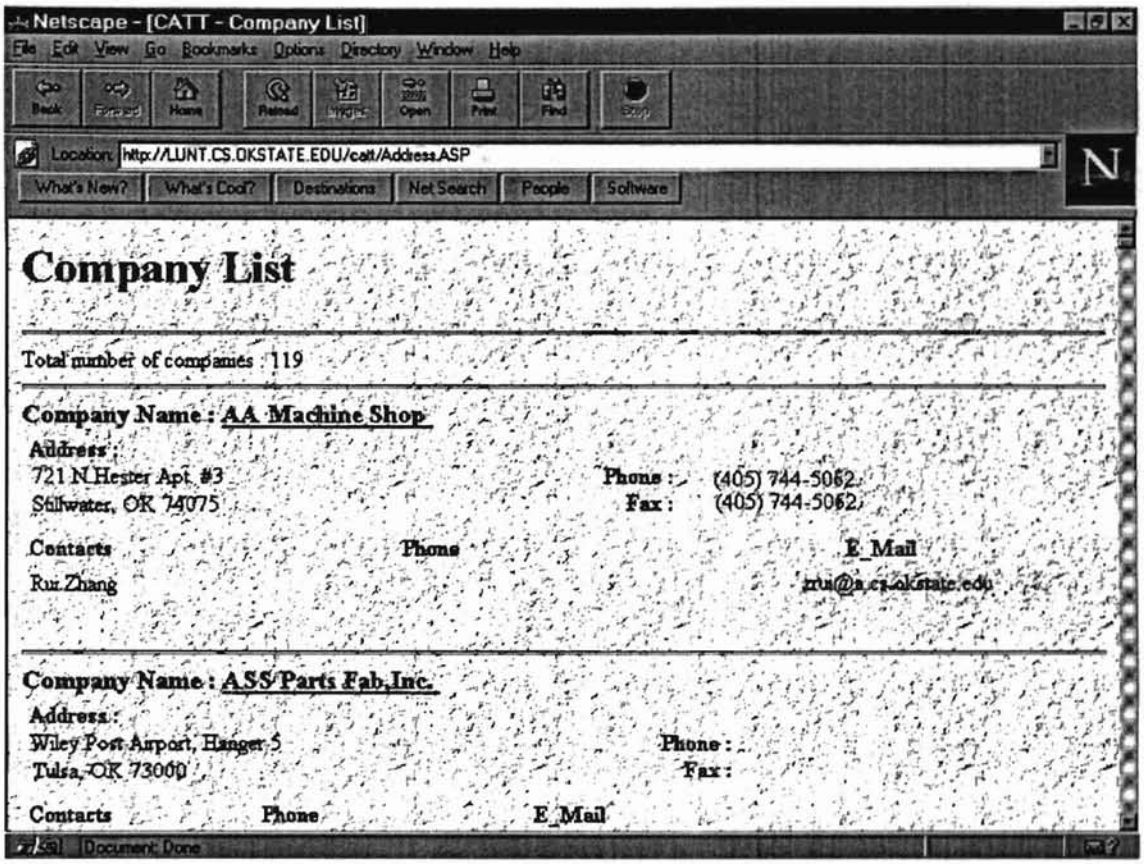

Figure 31 : A list of companies

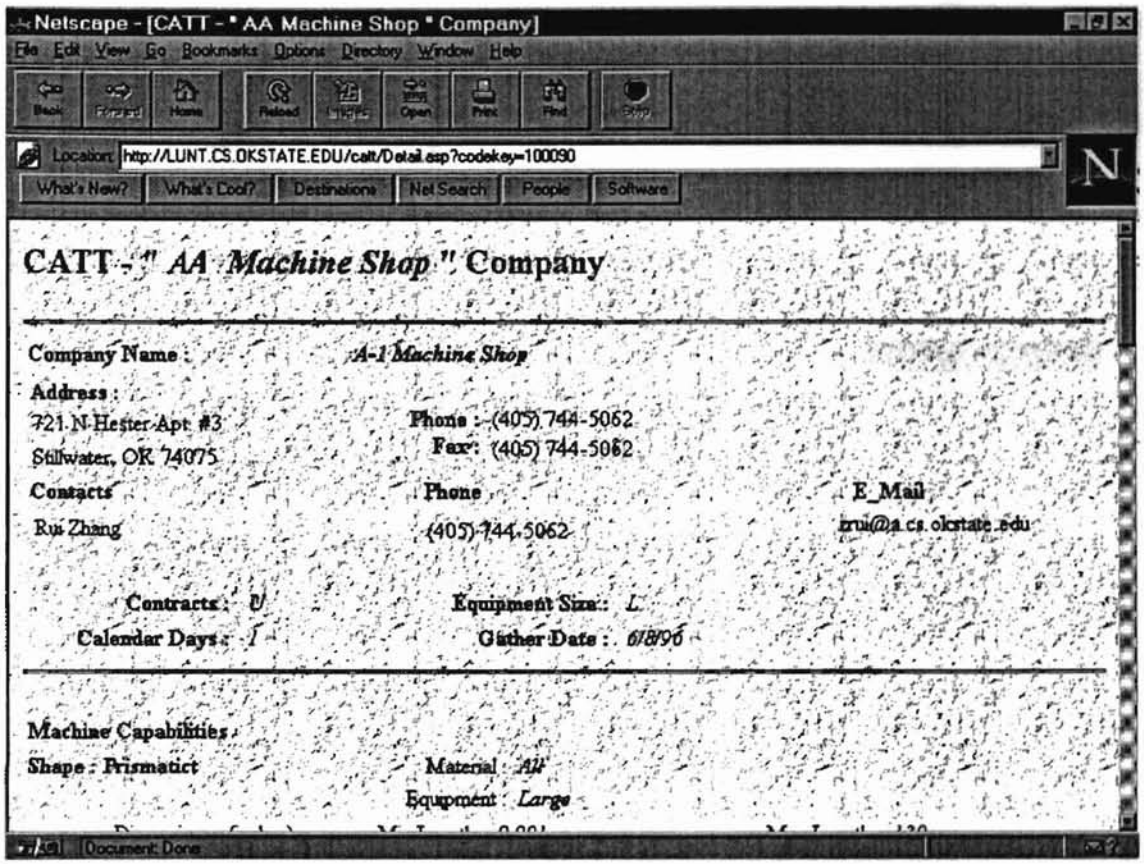

Figure 32 : Information related to an individual company

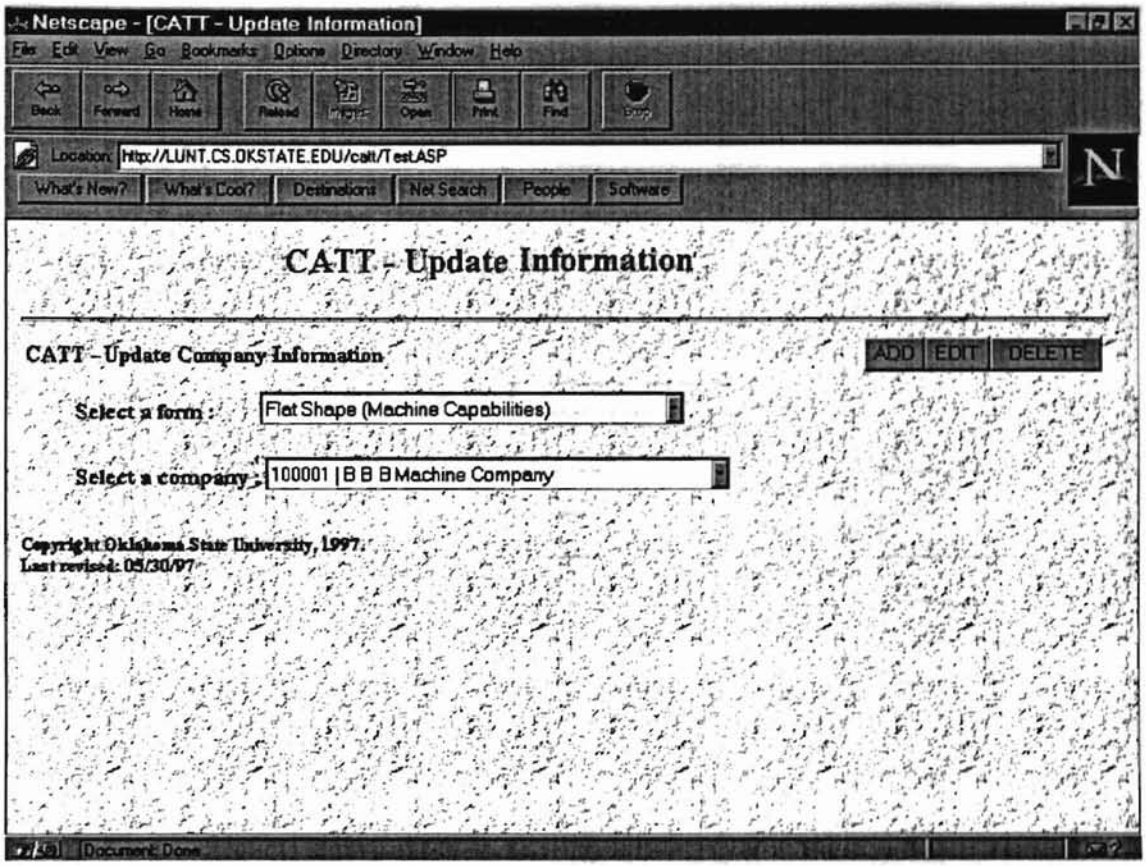

Figure 33 : A page for updating information

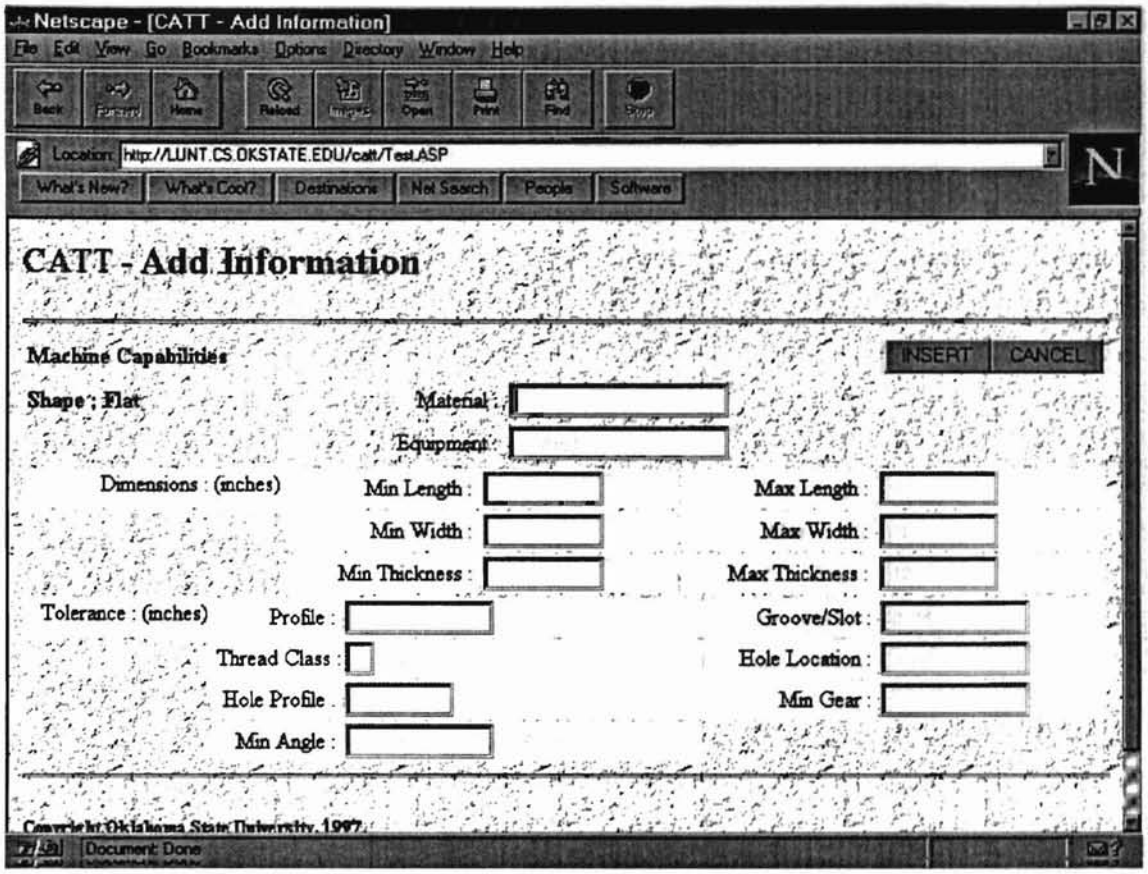

Figure 34 : Adding one company or one form for a company

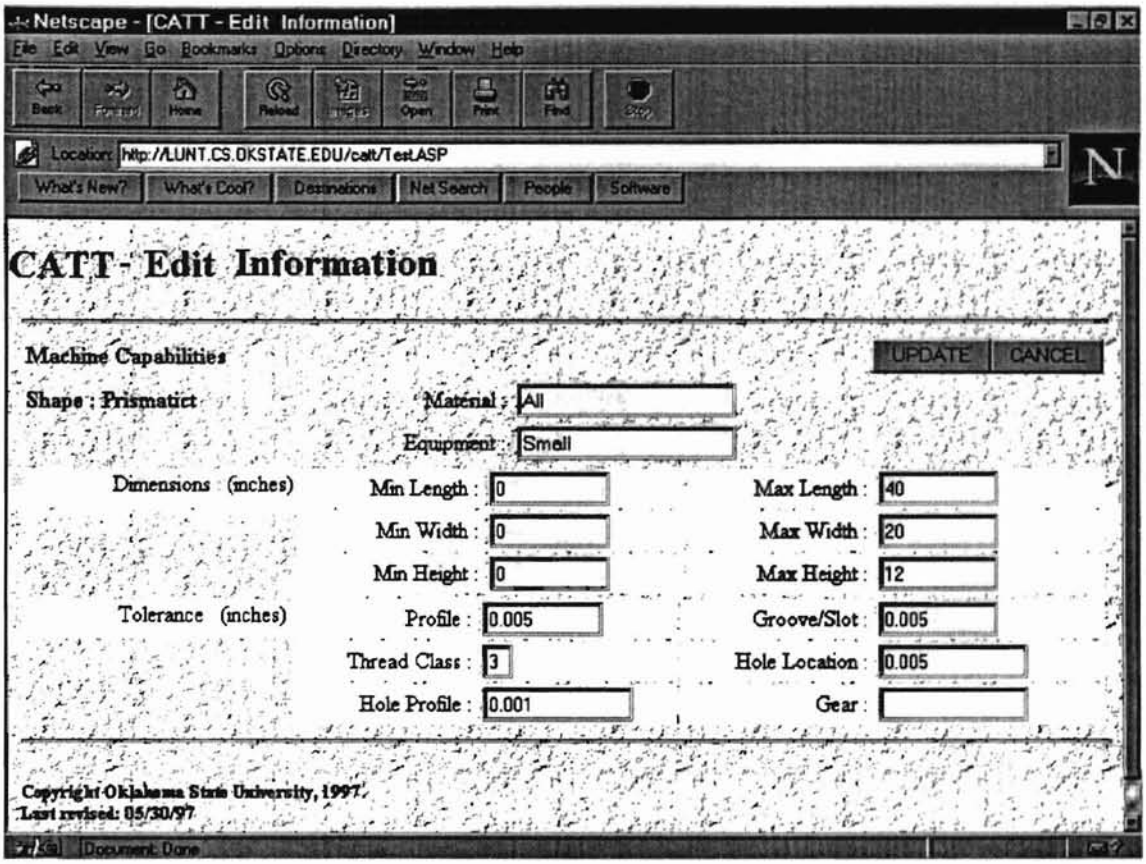

Figure 35 : Edit information for a company

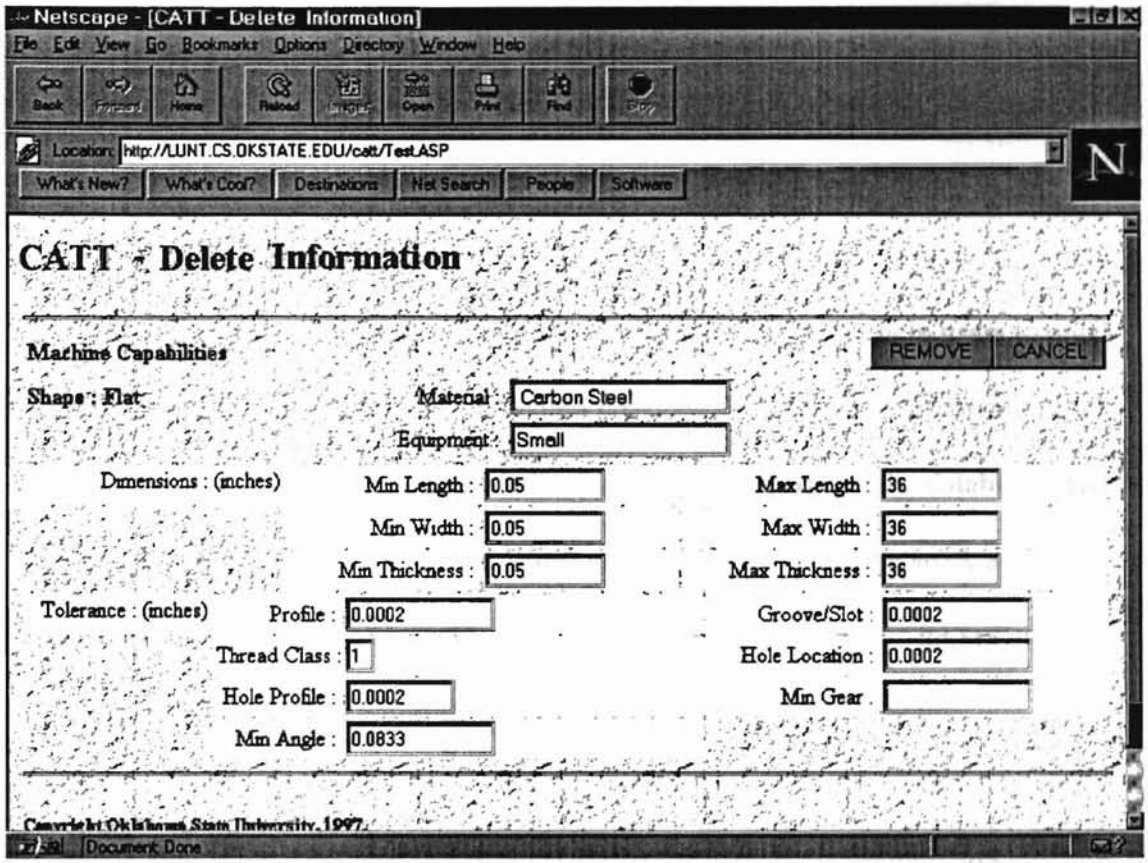

Figure 36: Delete information for a company

CHAPTER V

Conclusion and Future Work

In this thesis, a WWW user interface has been developed to help users to query and update a relational database. The system provides a GUI for the authorized user to update information and allows browsers to view the data and query the database. The software is implemented by using MS Windows NT to support the network and security, and a MS SQL Server used as a relational database management system.

From this project we find that the above method is a good way for communication between users and a management center. The authorized user can access the data at anytime, anywhere. This method is a great communication tool and enables users to save large amount of time and other resources. But due to the complicated relationships among these tables, and some flaws in the original table design, some relations have too many attributes with sparse data. In consideration of this situation, more tables with fewer attributes should be created to save response time, memory requirement, and secondary memory size.

BIBLIOGRAPHY

[AS, 1995] Ayer, Steve Object-Oriented Client/Server Application Development Using ObjectPAL And C++, *New York : McGraw-Hill, Inc.,* 50-149, 1995.

[BO, 1982] Bray, Olin H., Distribute Database Management Systems Lexington Mass, Lexington Books, 1982.

[CC, 1990] Chapman, C. J. A Visual Interface to Computer Programs for Linkage Analysis. *Am. J. Med. Genet.* 36, 115-160, 1990.

[CP, 1976] Chen, P. P. The Entity-Relationship Model-Toward a Unified View of Data. *ACM Trans. Database Syst.,* 1(1), 9-36, 1976.

[CP, 1991] Chen, Peter P. S. The Entity-Relationship Approach to Logical Database Design, *Wellesley, Mass. : QED Information Sciences,* 1991.

[DC, 1995] Date, C. J. An Introduction to Database Systems, *6th edition. Addison-Wesley Publishing Co., Reading, Mass.*, 63-74, 1995.

[DR, 1996] Dobson, Rick Make Access and the Web Work Together, *BYTE*, Vol. 0021, 71-72, October, 1996.

[GM, 1994] Gorman, Michael M. Enterprise Database in a Client/Server Environment, *New York : John Wiley & Sons,* 1994.

[JB, 1996] Jepson, Brian, World Wide Web Database Programming for Windows NT, *New York : John Wiley & Sons,* 1996.

[KHSA, 1991] Korth, Henry F. and Silberschatz, Abraham Database System Concepts, *2nd edition, New York : McGraw-Hill,* 437-450, 1991.

[LC, 1996] Lang, Curt., Database Publishing on the Web & Intranets, *Scottsdale, AZ : Coriolis Group Books*, 1996

[LD, 1995] Lowe, Doug. Client/Server Computing for Dummies, *Foster City, Calif : IDG Books Worldwide,* 1995.

[LM, 1995] Loomis, Mary E. S. Object Database : The Essentials, *Reading, Mass : Addison-Wesley*, 1995.

[MSCA, 1996] Administrator's Companion Microsoft SQL SERVER version 6.0 For the Microsoft Windows NT Server 4.0 Network Operating System, *Microsoft Corporation*, Redmond, Washington, 1996.

[MSCI, 1996] Internet Guide Microsoft Windows NT Server Version 4.0, *Microsoft Corporation*, Redmond, Washington, 1996.

[MSCS, 1996] System Guide Microsoft Windows NT Server Version 4.0, *Microsoft Corporation*, Redmond, Washington, 1996.

[NPCK, 1995] Nadkarni, P. M. , and Cheung, K. H. SQLGen : A Framework for Rapid Client-Server Database Application Development. *Computer and Biomedical Research, 28*, 479-499, 1995.

[RD, 1996] Rensin, David K., SQL Server 6.5 Secrets, *Foster City, CA : IDG Books Worldwide,* 1996.

[SR, 1995] Signore, Robert. The ODBC Solution : Open Database Connectivity Distributed Environments, *New York : Mc Graw-Hill,* 1995.

[SW, 1996] Stallings, Willian The Backbone of the Web, *BYTE,* 63-70, October, 1996.

VITA

Rui Zhang

Candidate for the Degree of

Master of Science

Thesis: DESIGN AND IMPLEMENTATION OF A WORLD WIDE WEB INTERFACE TO QUERY AND UPDATE A RELATIONAL DATABASE

Major Field: Computer Science

Biographical: Born in Beijing, China, March 22 1964, the second daughter of Zhang Chuqiao and Cai Yuying. Marred to Wen Shaokai, August 1, 1990.

Education: Graduated in July, 1982 from Beijing 124th high school in Beijing, China. Received a Bachelor of Mathematics Science degree in Mathematics Science from Capital Normal University in May, 1986, Beijing, China; completed the requirements of the Master of Science at Oklahoma State University, Stillwater in July, 1997.

Professional Experience: Software specialist from September 1986 - January 1991 in Beijing General Computer Company, Beijing, China. Software engineer January 1991 - December 1994 in Beijing General Computer Company, Beijing, China. Research assistant from August 1996 - July 1997 in Oklahoma State University, Stillwater, Oklahoma. Teaching assistant from August 1996 - May 1997 in Oklahoma State University, Stillwater, OKlahoma.