NEAR EAST UNIVERSITY

Faculty of Engineering

Department of computer Engineering

Call management system

Graduation Project
Com- 400

Student: MUTAZ MOHAMMRD ABU HASHISH
(20000758)

Supervisor : Assist.prof.ERDAL ONURHAN

Nicosia – 2003

ACKNOWLEDGMENT

"This project was not possible to be prepared without the guidance and the support of my supervisor ***assist.prf.dr.erdal onurhan***

I am indebted to him for his complete support and guiding throughout all the stages of the preparation, and providing his constructive comments.

also I would like to take this opportunity to thank ***him*** *for help and support, not just throughout this project but also through the courses which he provides to the students in the Department of Computer Engineering, Through these courses I have gained a lot of knowledge, which helped me in the preparation of this project.*

I thank my family, specially my dad, ***Mohammed*** *for their constant encouragement and support during the preparation of this project.*

I would like also to thank my friends specially , ***khaled Taher ,Ansam*** *for their advice and support."*

ABSTRACT

This project is generally about a database It has four parts.

The first one gives a good explanation about introduction and definition of database and database management systems.

The second one briefly explains about Architecture and Operations of Microsoft Access2000.

The third one gives flowcharts and all things about the project.

The fourth one gives tables ,forms,reports,and code.

The main program of **call management system** by using Microsoft access2000, gives programming that controls and know all calls, to and form a company.

TABLE OF CONTENTS

CHAPTER THREE: FLOW CHARTS

CHAPTER FOUR: OPERATION OF MICROSOFT ACCESS

INTRODUCTION

A data base **(DB)** is developers who want to use Microsoft access 2000 to build custom application it covers the features that makes access2000 that you are most likely to use in your programming projects.

Access is a popular development platform in large measure because it is part of the Microsoft office suite many clients want their access system to interoperate with the rest of office and they want system that are transparent and easy to maintain without developer assistance.

Longtime access developers will find the essential information on the advances introduction in access2000 another target readership is developers who primarily use non Microsoft technologies such as dbase or paradox they know all of the development concepts but they do not necessarily know to implement those concept with access.

CHAPTER ONE : INTRODUCTION TO DATABASE

1.1 Introduction to data base

Most programmers aren't qualified database administrators, but were often thrust into the role of "Acting DBA". This article will cover some basic techniques of database design and modeling. As usual I will use real world examples and expose certain pitfalls of not following these guidelines. This project is designed for the programmer who is familiar with client server databases, and SQL (however much of this applies to ISAM databases such as Paradox also)

This project will allow you to design databases that are flexible, scalable and easy to maintain. They should also be of the caliber that when an actual DBA or another programmer inherits your system they will not want to go find another job. This is not intended to train you to be a full fledged DBA, or cover advanced design or performance issues. This is a crash course in "etiquette and procedure" for those of you that already know how to translate real world concepts into tables and relationships.

There are also some useful tips in here for DBAs. Too often DBAs do not receive input from developers, and have no idea what makes thing hard or easy for us. And in some cases there are DBAs who are so stuck in "acadamia" that they don't have a taste of reality and the great pains that must be endured to work with their data models. If you have to work with a "Tenth Normal Form" DBA or one who thinks the 3 in 3NF is "cubed", I advise you plaster this article on his cubicle wall.

1.2 Database Overview

1.2.1 What is a Database? What are they used for?

A database stores electronic data in an organized and accessible manner. The size and scope of databases can vary wildly, from a small database used by an individual to file phone numbers, to an extremely large enterprise database that stores several terabytes of information accessed by tens of thousands of individuals. A database's tables and rows hold data in much the same way as a file cabinet or an address book, but they allow users more flexibility in what kinds of searches they can perform on the data as well as how they want to view the data.

1.2.2 What is a DBMS?

A Database Management System (DBMS) is a software system that is used both to create databases and manage the information stored within them. The architecture of the DBMS will frequently determine or limit the possible uses of the databases it creates. Some DBMS's work best for creating single-user databases, while others can build databases that accommodate multiple users in larger corporate environments.

1.2.3 What is a DBA?

A Database Administrator (DBA) is crucial to any organization that has mission-critical information stored in its databases. DBAs are responsible for the design and administration of databases, and for ensuring that the highest level of data integrity is maintained. Database maintenance includes performance tuning, monitoring the logs for errors, performing backup and recovery procedures as well as tasks associated with the data stored in the database, which include data transfer, data replication and data cleansing.

1.3 Database Architectures—RDBMS, OODBMS, and ORDBMS

A database consists of one or more tables, each containing data stored as individual records. Different database architectures determine how the tables and records are organized or related to one another. The first database architectures - Hierarchical and Networked - have largely been superseded by the Relational, Object-Oriented and Object-Relational architectures.

The Relational Data Model, developed by Todd Codd in 1969, allows multiple tables to be related to one another within a database For example, one customer's information could be recorded in separate tables such as "Personal Information", "Marketing Efforts", and "Service Requests". The information stored in these tables will then relate back to the customer's main record. A relational database management system (RDBMS) also offers flexibility in terms of how the customer's data can be viewed. To access the information stored in relational databases, users can either build queries using the Structured Query Language (SQL), or they can utilize a user interface that translates their requests into SQL and displays the results. While the American National Standards Institute (ANSI) approved an early version of SQL as a standard, many RDBMS's also use customized, proprietary forms of the language.

The Object-Oriented database model emerged in the mid-1980s due to the dissatisfaction of some database users with the structural limitations of RDBMS's. The Object-Oriented model defines each piece of data and its associated processes as an individual object. According to the basic tenets of this model, all information about an object is stored in one place instead of being stored across multiple tables, as is done in the relational model. An Object-Oriented Database Management System (OODBMS) also integrates more easily with applications that have been written with an Object-Oriented programming language such as C++ or Java. Despite the advantages of the Object-Oriented approach, no standard model for the construction of an OODBMS yet exists. For this reason, at least in part, RDBMS's still dominate the database market.

One effort to combine the best parts of the RDBMS and OODBMS is the Object-Relational Database Management System (ORDBMS). This model allows developers to incorporate the best parts of an RDBMS and an ODBMS. An ORDBMS works with objects like an ODBMS,

1.4 The Future of Databases

Databases play an important role in both data management and data storage in today's Information Age. The high value placed on information-gathering by companies as well as individuals requires efficient methods of storing and accessing information. Database architectures and products will need to be even more highly scalable to accommodate and support this increasing production of data in the future.

but also allows SQL-based querying like an RDBMS.

1.5 Database Designer

1.5.1 General Description: Under the general supervision of the Director of the Digital Library Program department and in close collaboration with other members of the Design & Development Unit, this position designs and manages databases by developing data models, database schemata, and management tools which effectively support the Library's information architecture.

1.5.2 Duties:

1. Analyzes and defines information distribution processes and data requirements.
2. Develops data models and data schemata.
3. Designs databases at both the logical and physical levels.
4. Conceptualizes and implements data migration from existing structures into new ones.

5. Develops and implements database management tools.
6. Creates appropriate documentation and training materials for new products.

1.5.2.1 Strongly preferred qualifications:

- Strong knowledge of and experience with development and delivery of high-quality information services in an academic or research setting

1.5.3 Single and multi-file databases

A database can contain a single table of information, such as the phone book), or many tables of related information. An order entry system for a business, for example, will consist of many tables:

- an orders table to track each order
- an orders detail table for tracking each item in an order
- a customer table so you can see who made the order and who to bill
- an inventory table showing the goods you have on hand
- a suppliers table, so you can see who you need to re-order your stock from
- a payments table to track payments for orders

Each of these tables will be linked to one or more of the other tables, so that you can tie information together to produce reports or answer questions about the information you have in your database.

Multi-file databases like this are called *relational databases*. It's relational databases, as we'll see later in this series, that provide exceptional power and flexibility in storing and retrieving information.

1.6 Database programs

To create and maintain a computer database, you need a database program, often called a database management system, or DBMS. Just as databases range from simple, single-table lists to complex multi-table systems, database programs, too, range in complexity.

Some, such as the database component of Microsoft Works, are designed purely to manage single-file databases. With such a product you cannot build a multi-table database. You can certainly create numerous tables for storing different types of information, but there's no way to link information from one table to another. Such programs are sometimes called flat-file databases, or list managers.

Other database programs, called relational database programs or RDBMSs, are designed to handle multi-file databases. is a relational database that's easy to use and fairly inexpensive.

The most popular relational databases are the offerings from the big three software companies. Lotus, Corel and Microsoft each produces a full-featured relational database application available both as a standalone program or as part of its integrated suite. Lotus has Approach, Corel has Paradox and Microsoft has Access.

1.6.1 Database program tools

A database program gives you the tools to:

- design the structure of your database
- create data entry forms so you can get information into the database
- validate the data entered and check for inconsistencies
- sort and manipulate the data in the database
- query the database (that is, ask questions about the data)
- produce flexible reports, both on screen and on paper, that make it easy to comprehend the information stored in the database.

Most of the more advanced database programs have built-in programming or macro languages, which let you automate many of their functions.

1.6.2 Using a database

If the mention of programming languages makes you feel you're getting out of your depth, don't worry! Most of the database programs you're likely to encounter can be used at a variety of levels.

If you're a beginner, you'll find built-in templates, sample databases, 'wizards' and 'experts' that will do much of the hard work for you. If you find the built-in databases don't quite work for you, it's easy to modify an existing database so it fits your needs, and it's not at all difficult to learn to create your own simple database structure from scratch.

For more advanced users, the more powerful database programs enable you to create complete, custom-built, application-specific systems which can be used by others in your organisation or business.

1.7 When not to use a database

Even though you can use a database program to do anything from managing the inventory of a parts supply warehouse to managing your personal finances, sometimes the smart option is to not use a database at all. That's because there's no point in reinventing the wheel: If you want a personal financial manager, you're far better off spending money on one of the commercial programs, such as or, than slaving for weeks creating your own version of the same thing.

The same goes for vertical market applications. Before you spend months designing a church contribution application or that parts inventory system, take a look around the Web or at your local software supplier to see if something similar has already been created. Shareware libraries such as are littered with such specialist applications.

1.8 Introduction to VLDB, Very Large DataBases

Welcome to the discussion on VLDB. The purpose of this series, these white papers is to present and discuss VLDB regarding it's implementation, application, design and architecture needs. The focus here will be centered on the techniques necessary to setup, maintain, and grow a database of 1 to 80 Terabytes in nature. The target audience is those who have a background in systems architecture, database administration, or systems administration. This series will not discuss the business benefits nor pro's and con's of VLDB from a business perspective.

What the series promotes are the following goals of understanding:

- Impacts of VLDB on Hardware and Software
- Sizing and scooping of Hardware
- Parallel Processing, multi-threaded Architecture in relation to VLDB

The other areas of discussion will focus on: networks, partitions, clustering files and indexes, database sharing, CPU speed and quantity, RAM speed and quantity, and an introduction to

MPP possibilities (such as NUMA-Q). A new series will introduce new structures capable of assisting in housing VLDB data sets: a structure called a Data Collection, and a new data architecture model called the Data Vault™ will be presented as a means for massive data storage in a detailed normalized form. Watch for this series in the coming months.

Much of the theories and hypothesis presented here are supported by mathematical functions, however mathematical theory is not the focus of this series; therefore the mathematical functions may be referenced in certain places. However the proof of these theorem's or math functions will not be demonstrated.

1.8.1 What is VLDB? VLDW?

Different definitions exist of each term, and the terms themselves conjure different images for everyone. However, the standard definitions in the industry are as follows:

- VLDB = Very Large DataBase, an environment or storage space managed by a relational database management system (RDBMS) consisting of vast quantities of information. The definition of what exactly a VLDB is changes every day as hardware and software adapt, become faster and faster – and are capable of handling more and more loads. Usually though, VLDB's "don't fit on laptops". But wait, this changes tomorrow to. IBM is scheduled to produce a disk that is capable of holding 3x the gigabytes on your laptop within the next year.
- VLDW = Very Large Data Warehouse. A data warehouse can be made up of both physical (hardware/software), and virtual (electronic information) content. The warehouse being the structure or architecture serving a specific business purpose (or several). The VLDW is based on VLDB technology. The two usually go hand in hand

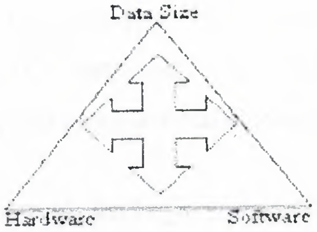
1.8.2. Technical Theory

The database or data warehouse will grow in the specific areas consisting of transactional information. The information describing the transaction itself is usually not as frequently changing, and is capable of being termed: Meta Data – or data about data. Most of the transactions around the world consist of: date and time (when did the transaction occur?), who (the individual conducting, initiating, or entering the transaction), and what (the content of the transaction itself). Beyond these three components, the transactions typically include many descriptive elements, and relationships between information – all of which must be recognized, but should be stored in a minimalist fashion in the database. Therefore, compression ratio's, growth ratio's, and metrics ranging from implementation time, to overall sizing should be computable attributes following a mostly accurate profiling activity. The keys are: how many types of transactions, and how many total transactions are being entered in to the system?

The theory goes on to say: the relationship between hardware, software, speed, and RDBMS, and data set growth appears to be closely related – as well as consistent in nature. Therefore most common techniques included in mathematical algorithms should hold true for splitting data in to chunks while holding other elements constant. As a result, speed should only change if one of the other variables change.

The suggested very broad formula for this theory might be: speed = [(hardware)*(software)] bytes per second / (size of data in bytes)

With this formula, if the size of the data increases, while the hardware and software stay constant, the speed decreases. Or vise versa if the hardware increases and the size of the data stay the same the speed *should* increase. Of course the dependency is on the bottleneck of the system. A delicate balance is achieved in keeping these three "points" moving in ever larger directions in equal portions. Think of three sides of a triangle:



The pressure is in the middle of the triangle, pushing outward. Speed is the size of the arrow set inside the triangle. Increasing one, doesn't necessarily increase speed. A number of factors must be addressed. If you picture a pyramid, the 3d dimension might be the number of parallel processes capable of executing the requests. Consider the volume of a pyramid as the resulting speed, the more volume, potentially the more speed for more data. For this example, the items considered for each piece are discussed below:

1. Hardware = Size + Speed

Size = Amount of RAM, # of parallel CPU's, # of controllers, # of disks

Speed = RAM speed, Disk Speed, Bus Speed, Network Speed (bandwidth), and removing the # of idle cycles per second per CPU, as well as the # of blocked I/O processes per second per CPU.

2. Software = # of parallel processes + # of partitions + operating speed of software (throughput)

If the formula holds true, then there are a number of things that can be done to achieve better performance.

1.8.3 Why is all of this important?

We are seeing rapid change in both technology and the requirements for analysis of massive sets of information. More and more businesses are looking at the related issues and recognizing that patterns, trends, and analysis of these items is becoming critical to the success of the business. Ok – enough business talk. What about the technical end of it? Because of the business, the IT specialists left to implement these complex problems are beginning to deal with physical problems they've never had to think about before. OLTP simply "deleted" or rolled off the unused portion of the data, Data Warehouses physically were unable (until recently) to hold such massive sets of information. The focus has simply changed, the name of the game for IT is to get the most accurate answer from a compilation of as much information as possible – and do it as quickly as possible, providing that much desired competitive edge in business.

In recognizing speed as a foremost goal, we begin to realize that one of IT's responsibilities is integrating, understanding, and tuning all of these different components. Not to mention the mountains of data that exist within an enterprise. Technically, having a VLDB situation without the expertise to predict it, manage it, grow it, or architect it – is like having the information aboard the best aircraft carrier in the world without having the aircraft. What's the point? More simply put – it's like a rowboat without the oars.

More and more IT personnel need to understand the varying levels in order to tune the correct component. This is analogous to improving a car with the best carburetor money can buy, but then putting the same old fuel in to the engine, maybe the engine is too small? Maybe the fuel is bad? Maybe there isn't enough horsepower to take advantage of the best carburetor? Anyhow, by understanding only a single component of this particular triangle, only certain levels of performance and management of VLDB can be achieved. Provided the amount of money it costs a business to house, build and deploy VLDB this would be a grave mistake on the business's part to not properly train IT employees.

So where does one go to achieve such knowledge? They don't teach it, and it's something that's not readily available on the web. This series will try to assist in unraveling some of the complications and necessities of what to think about when getting in to VLDB for the first time. For those advanced with VLDB, please send any comments, omissions, errors, or thoughts to the author directly. We always welcome input. This series is meant to be dynamic in nature.

1.8.4 Who's involved in VLDB?

Now that the discussion has opened, who is actually involved in VLDB? Who needs to know? What's the weakest link? Ok, sorry for the bad pun. We're just checking to see if you're awake. The answer is: everyone. Everyone who interacts from a business perspective is interested in VLDB even if they don't know it. They need to understand that the questions they are beginning to ask about the content (specifically the historical patterns of the content) require more and more analysis. Which in turn, most statisticians will tell you that to predict for 6 months (most accurately), should have at least 12 months behind it. In other words: twice as much data for half the amount of prediction over time. All of these statistics prove true when coming to most accurate conclusions in VLDB. Sure sample sets can be implemented, but what if that key factor is missed? There are examples of these written up in various books such as Oracle8i – DBA Handbook, and some other sparsely populated writings.

From an IT perspective, everybody that touches, maintains, architects/designs, or builds a data warehouse is interested (or should be) in VLDB. The list of roles include: systems architect, systems administrator, enterprise architect, database administrator, ETL operator, dss/olap (Business Intelligence individual), operations for backup/restore and disaster recovery.

The next series will cover the concepts of VLDB such as parallel processing, architectures, and introductions to hardware/software requirements.

1.8.5 VLDB / VLDW Concepts

This next section discusses the concepts behind VLDB and VLDW. What to think about when creating these situations or considering them for implementation practices. There are many concepts regarding VLDB, which due to time and space constraints will not be covered here. If there is a desire to see a write up on certain concepts pertaining to VLDB that has been left out of this series please feel free to contact Core Integration Partners and let us know. We will attempt to include them in a section of post-series write-ups.

The concepts included in this discussion focus on the hardware, software, and architecture sides of VLDB. There are separate chapters on both hardware and software included in this series. For that reason these concepts are interwoven with all such ideas, and should be read as precursory knowledge before launching in to the other respective areas. Included in the discussions are the following items:

1.9 Architecture Overview

Architecture plays a much larger role in VLDB than most care to think about. In order to understand VLDB/VLDW we must consider the architecture and the concepts behind what makes it all work. The architecture is like the set of blueprints for a house, without the overview of what to put where, and the guidelines or steps to build it – it becomes difficult to create correctly. All too often most projects begin with the right intentions, but forget to include the architecture components. This leaves the foundations of VLDB extremely weak, and many problems begin to crop up as data set size grows rapidly.

An example paradigm might be as follows:

Suppose you had a lumber yard, and in that yard you had 1 stack of 500 foot boards, 8 feet high, and 6 feet deep. These boards are stacked neatly on one side of the lumberyard. Your objective is to move the boards across to the other side of the yard. Do you simply start by grabbing a board, and attempting to move it by hand, one by one across to the other side? Most often all the options of how the board(s) can be moved are considered. In doing so, you find out that there's a forklift of standard size is available. Does this mean you can move all the boards now? Not necessarily. The boards are too long, and too heavy for a single forklift, so additional options must be considered.

In this paradigm, if the boards represent data, the forklift represents the database/query or software, and possibly a CPU (the engine of the forklift). If the boundaries of the lumberyard aren't considered, there could be trouble. Again, the architecture or the entire process (all the options) must be considered before the best option for the situation at hand is chosen. If the worst option is chosen (because the entire architecture was not considered), then the boards may break – or it may take a very long time to move the boards, or halfway through, no more boards could be moved. Maybe some of the other boards have now been placed in the way of completing the task.

Four other co-workers join you, each with their own forklift, and your boss has just told you these boards need to be moved within the next 2 hours. This further complicates the problem. Now there are five people, five forklifts, and still 500 ft long boards. After discussing it, everyone agrees that there are two options: cut the boards in to 5 portions of 100 feet each and each move the stacks independently, or put all five forklifts at the same time under the boards and try to move the entire stack all at once. Of course this would require great synchronization so as not to disturb the stack or have it fall over.

After further consideration the option for cutting the boards is chosen, the stacks are moved efficiently across the yard, and the job is done in 1 hour and 45 minutes. Just in time for the boss to come out, and start congratulating you – explaining how he can't move the boards in trucks anyway if they were still longer than 100 ft. Or maybe he begins to yell, for the lack of asking about the fact that these 500 ft boards were specifically ordered at that length, and you and your co-workers should have asked first.

What happened here? You solved the problem with two possible outcomes (probably more outcomes available). But, was the architecture right to solve the problem? The problem itself was solved, but can be considered a success only if the requirements were discovered before delving in to the actual application of solving the problem. Let's examine one more possibility…

You and your co-workers go back, ask the boss hey, do you want these in 500 ft lengths, or do you mind if we cut them? If we cut them we can move them with our 5 forklifts, if you want them in one length, we need to use the crane to haul the whole stack across the yard. The boss thinks for a minute and responds, the crane is more expensive, but the customer wants the boards in 500ft lengths. Go ahead and use the crane to move the boards. Successfully, the boards are moved across the yard in 45 minutes, in a single stack – uncut.

Interesting, now there's another solution to the same problem: an introduction of a high-priced item, but one that can accomplish the task successfully according to the requirements. The items in this real-world case should be thought of as follows: the boards are data in a table. Cutting the boards is like partitioning them. The forklifts are like CPU's with RAM for a carrying capacity, and they have a certain speed. The more forklifts, the faster the stacks can be moved, at a maximum of 5 stacks – other forklifts would simply sit idle. Each forklift operating independently is like a parallel process, capable of accessing a partition. If the boards are in a single stack – the forklifts must operate together at the same time (as if they were one large forklift).

The crane is a high-priced item that can meet the requirements, but it can do it faster. It's like having more CPU's, more RAM, faster machines, faster bus speeds, faster disks, more controllers, etc.. Higher cost get's the job done faster than lower cost, but it's over-kill unless this kind of operation will take place regularly. If the data set grows (more boards, more stacks), then it would make sense to invest in a crane, but only after the forklifts are overwhelmed or the time frame is squeezed to tightly.

Ok, enough of the theory. Let's get down to brass tacks. The bottom line is: architecture, engineering, and requirements are extremely important in considering VLDB. As this series will now delve in to the technical sides of all of this. Here we go....

1.10 What is a thread?

A thread is typically defined as an independent process which can execute asynchronous to other processes. Usually a thread is capable of operating independently of other threads, or processes. When a thread has to wait on the results of another thread, it will sit in "idle mode" until a token is passed (semaphore/mutex). Ok, in English – a thread can be likened to a car. Put four cars coming to a stop at a 4 way stop sign, each car is running independently of the other cars. They all arrive at the stop at different times. Then each car proceeds to go in turn and continue on their way. Think of each car being a thread, the stop sign being a semaphore or shared lock, by which all cars must stop first before proceeding, and each car can only proceed when the intersection is cleared.

Threads can also be thought of in terms of different users connecting to the database. Each has their own connection, and each can execute different queries at the same time, or they can execute the same query at the same time. The threads allow for multiple processes to occur at the same time. This concept is very important in understanding how multi-threaded architectures work. A multi-threaded architecture provides the system with the ability to run multiple threads or processes at the same time. The system performs management of the 4 way stop (with stop lights), so that accidents and traffic jams are avoided as best as possible.

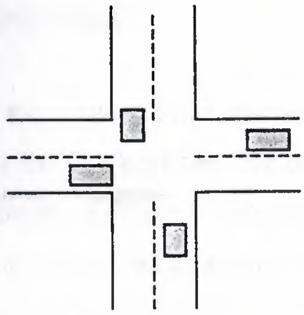


Figure 2-1 Four Way Stop Intersection

Each car represents a thread. All cars must stop before proceeding, but all cars are in motion independent of one another.

Why is this important? It is important to recognize threads as a basis of architecture for systems with VLDB, so that larger tasks or requests (such as massive inserts, selects, deletes or updates) can be split in to smaller multiple tasks that can run at the same time – thus shortening the overall time frame necessary to complete the task. Going back to the analogy used earlier, it's similar to having N# forklifts. Each thread is a forklift, the operators (people) are like the system that communicates and manages what each thread is doing. It can take lot's of little threads working together to complete what otherwise is a huge task, in a short period of time. However, without the Operating System providing parallel processing capabilities, threads still execute in order (serially). It would be like having five forklifts, but only one key that fits all, you can only operate one forklift at a time.

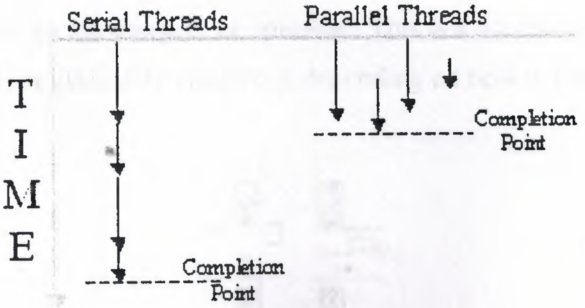


Figure 2-2: Serial vs Parallel Thread Execution

The time line increases, for serial execution are multiples longer than that of parallel execution. The parallel threads all executetTogether at the same time, while the serial threads execute one after the other.

1.10.1 What is Parallel Processing?

Parallel Processing is the ability of the system to run, manage, and maintain multiple threads at the same time (synchronously). It's the five keys and five operators needed to run each forklift independently. Degree of parallelism is a measure which indicates how many parallel processes can run at once before the system is overloaded or maximized. For instance, if you have five stacks of lumber, and seven forklifts, you can still only operate five forklifts at a time, indicating a degree of parallelism of five. This is a very simplistic explanation of the parallel processing theories. There are many books in the marketplace that describe in great detail how parallel processing and degree of parallelism is reached. It is not the scope of this series to discuss each in extreme detail, only to introduce the concepts as they relate to VLDB.

Parallel processing enables speed, and division of tasks across CPU's. This is also where the concept of load balancing comes in to play. Load balancing is the process by which the operating system decides where to run the threads, how many threads to run, and which ones need to sit idle. It attempts to maximize the overall usage of the hardware resources available. Load balancing in the lumberyard would consist of deciding which forklifts are out of gas and need to sit idle. Maybe one of the forklifts can carry a larger capacity load than the other, so it may be used to carry a slightly larger pile in a shorter time frame. Maybe it carries two smaller piles more quickly to the other side of the yard. Load balancing is dynamic in nature. Threads are swapped to and from different CPU's unless they are "CPU BOUND" during execution. CPU Bound means that the threads have been tied to run on a specific CPU, and only that CPU. According to the load balancer, cannot be moved to run on another CPU. Very few programs allow this level of control by the designer or operator. This is a double-edged sword, and can be tremendously beneficial, or extremely dangerous depending on how it's set up.

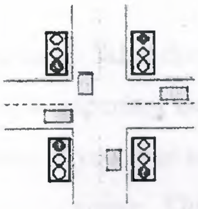


Figure 2-3 Four-Way Intersection

In this case, the stoplights represent the Operating System, and the capacities for executing or managing parallel processes. The threads (cars) are the processes.

Why is parallel processing important to VLDB? It allows multiple processes (SQL queries for instance) to execute at the exact same time frame, and with the same priority. It also provides the vehicle for very large tasks to be split in to multiple smaller tasks for the purposes of faster execution times. Without parallel processing, it becomes difficult if not impossible to handle systems of scale. Parallel processing also plays a huge role in the success of partitioning the data, which in the example provided here – are the stacks of lumber being cut in to shorter stacks. This concept of partitioning will be explained in one of the follow on chapters.

Ok, so now that threads exist, and parallel processes exist to manage threads, how does the system decide who takes priority and when? The answer is through a technique called pre-emptive multi-tasking.

1.10.2 What is pre-emptive multi-tasking?

Pre-emptive multi-tasking is the ability to handle parallel processes, but to grant priorities to certain processes so they may interrupt other processes at specific points in time, or in reaction to specific events. In the case of the Four-way intersection with stoplights, a pre-emptive task would be an ambulance, which changes all the lights to red, but is still (in most cases) able to go through the intersection. It's a random occurrence that's unpredictable, but when it happens, it takes priority over the cars on the road. The multi-tasking part is parallel processing, it allows multiple tasks to interrupt other tasks at the same time – for instance, three ambulances at three different intersections, all have the same effect and the same priority on the road. Hopefully not all the ambulances head towards the same intersection at the same time – this would cause collision, and the ambulances would have to slow down, stop and wait until each one makes it through the intersection. This concept is called Blocked-I/O or blocked processing, and it will be discussed later in this chapter.

Why is pre-emptive multi-tasking important? What does it mean to the VLDB world, and how does it play in the operating system? It's important because it allows the operating system to manage execution priorities. For instance, if you type a key on your keyboard the system must respond immediately, regardless of what it's doing. This is an example of a hardware interrupt, one of the highest priority tasks that can occur on a system. It allows the system to react to a series of unpredictable events at the time they happen. The system considers it less important to update the screen than to respond to a keyboard event. These priorities also run through different software in the operating system. In fact, in near real time systems or true pre-emptive multi-tasking systems such as Unix, these priorities can even be assigned. Unix term for this is "niceness". Setting the "nice" for software sets the priority of it's interrupt levels.

For instance, you can tell the system what's an ambulance, versus a fire-truck versus a car, etc.. This is important, to be able to set the priority in VLDB means that the database software can take precedence over other software running on the same system. Once priority is decided, the software makes the request to the operating system that enters a queue. When the system has time, or needs to execute the request it pulls it from the queue, assigns it to a processor, then it begins executing. Of course there are many more steps to this process which have not been described here but that's the general flow.

Unfortunately there are still systems out there that claim to be multi-threaded. This may be true, but when it comes down to brass tacks they are not true pre-emptive multi-tasking. These systems block processes from executing on an order of magnitude more frequently than those systems with true pre-emptive multi-tasking. It's one of the primary differences between Unix, and Microsoft's Windows platforms.

1.10.3 What is a blocked process?

A blocked process is a thread that is sitting in idle mode, waiting for another of its requests to complete. It can be likened to going to a gas station, finding out they are filling the gas tanks below the surface – so you have to wait maybe five minutes before you can fill up your car (when they turn the pumps back on). But, before you can get gas, these events such as turning the pumps back on must happen first. This sequencing of events causes you to wait. This is equivalent to "idle state" for the thread that was executing.

The thread sitting in idle, waiting for something else to happen or complete is called a blocked process or blocked thread. Blocked I/O simply means that the thread is waiting for a disk operation to complete which is the most common event to block threads. There are many different methods for threads or processes to become blocked, if you want to find out more about these things you can find books on multi-threaded processing which walk through all the details.

Are blocked processes bad? Not necessarily, unless there are too many processes waiting in the wings to execute. Sometimes multiple threads are waiting on each other, this could be equated to deadlock situation in a database. Deadlocks cause machines to freeze up – frequently requiring cold boots. Blocked processes are a problem for the resource monitors, because the

CPU utilization rates drop during processing while the CPU suspends the thread in "idle" or wait state until the blocked lock releases and continues to execute. The problem is, the actual utilization of the CPU may drop, but 100% of all available resources are taken, causing CPU load to be significantly higher than utilization rates.

The less blocking that the processes or threads do, the faster the execution of the overall processing. The problem is, no matter what's done, there will always be interrupting threads with higher priorities, some of which require one or more lower priority processes to become blocked. Why is this important to VLDB? When dealing with massive sets of information, it is most optimal to have the process broken up in to smaller more manageable processes, but also to have them be able to be independently executed. When the dependencies from the separate threads have been removed they can execute in parallel and have less "blocked" action. As long as each thread is going after independent data sets on disk, again this leads to partitioning the large data set in the database. If you have four threads going after the same data, then you'll end up with dueling threads. It would be similar to having four forklifts trying to lift the same stack of lumber at the same time. It makes no sense.

1.10.4 What are pipes and why do they matter?

Pipes are conceptual term used for throughput – at least in this document. The pipes are the ability of the system to perform movement of X amount of information in Y time frame through a series of circuits or connectivity. Ok – what was that again? Basically with the lumber yard, it may be the number of stacks or total amount of lumber that a delivery truck can carry over the course of a single tank of gas. It's the transport mechanism for the information inside the system. Without proper sizing of the transport mechanism the job takes too long, costs too much, or produces too much waste or excess. In technical terms, that relates to over-utilization of available resources, under-utilization of available resources, or too tight a time frame in which to move too much information.

Just like everything else, pipes come in all shapes, sizes, and colors. Virtual pipes can have any attribute you wish to assign to it. Anyhow the throughput of these pipes is what matters most. How fast can a pre-determined amount of information travel from point a to point b? It may be a network connection, or disk connection, or disk controller, bus speed on the CPU board. It could even represent the speed of the CPU – yes it's all about speed and performance. As mentioned earlier, throughput is a critical success factor to VLDB. Especially since users are less and less willing to wait for responses from the systems they access, they need/want immediate answers from mountains of data.

The pipes come in to play, with how many pipes are available, their bandwidth (diameter & volume). These are frequently the most expensive components to "get right" in the complex world of servers and connectivity, but in most cases, they are the most beneficial to increasing throughput. Going back to the triangle of power introduced in the first series, the connectivity between the points is the pipe system.

CHAPTER TWO : WHAT'S NEW WITH ACCESS 2000

2.1 What's new with access 2000?

Access2000 makes major strides in many areas Microsoft has created a profoundly new product that still feels like the access you know the new with access 2000 is active x data object (ado), enhanced sql server interoperability visual basic for application (vba) and packaging enhance ments Microsoft jet engine improvements and improved web interoperability

2.2 Microsoft Access 2000 Is a Database

If you're a serious user of a personal computer, you've probably been using word processing or spreadsheet applications to help you solve problems. You might have started a number of years ago with character-based products running under MS-DOS but subsequently upgraded to software that runs under the Microsoft Windows operating system. You might also own some database software, either as part of an integrated package such as Microsoft Works or as a separate program.

Database programs have been available for personal computers for a long time. Unfortunately, many of these programs have been either simple data storage managers that aren't suitable for building applications or complex application development systems that are difficult to learn and use. Even many computer-literate people have avoided the more complex database systems unless they have been handed a complete, custom-built database application. Microsoft Access, however, represents a significant turnaround in ease of use, and many people are drawn to it to create both simple databases and sophisticated database applications.

Now that Access is in its fourth release and has become an even more robust product in the second edition designed for 32-bit versions of Windows, perhaps it's time to take another look at how you work with your personal computer to get the job done. If you've previously shied away from database software because you felt you needed programming skills or because it would take you too much time to become a proficient user, you'll be pleasantly surprised at how easy it is to work with Access. But how do you decide whether you're ready to move up to a database system such as Access? To help you decide, let's take a look at the advantages of using database application development software.

2.3 Microsoft access database Capabilities

An RDBMS gives you complete control over how you define your data, work with it, and share it with others. The system also provides sophisticated features that make it easy to catalog and manage large amounts of data in many tables. An RDBMS has three main types of capabilities: data definition, data manipulation, and data control. All this functionality is contained in the powerful features of Microsoft Access. Let's take a look at how Access implements these capabilities and compare them to what you can do with spreadsheet or word processing programs.

2.4 Microsoft Access as an RDBMS

Microsoft Access is a fully functional RDBMS. It provides all the data definition, data manipulation, and data control features you need to manage large volumes of data.

2.5 Microsoft access Data Definition and Storage

While you're working with a document or a spreadsheet, you generally have complete freedom to define the contents of the document or each cell in the spreadsheet. Within a given page in a document, you might include paragraphs of text, a table, a chart, or multiple columns of data displayed with multiple fonts. Within a given column on a spreadsheet, you might have text data at the top to define column headers for printing or display, and you might have various numeric formats within the column, depending on the function of the row. You need this flexibility because your word processing document must be able to convey your message within the context of a printed page, and your spreadsheet must store the data you're analyzing as well as provide for calculation and presentation of the results.

This flexibility is great for solving relatively small, well-defined business problems. But a document becomes unwieldy when it extends beyond a few dozen pages, and a spreadsheet becomes difficult to manage when it contains more than a few hundred rows of information. As the amount of data grows, you might also find that you exceed the data storage limits of your word processing or spreadsheet program or of your computer system. If you design a document or spreadsheet to be used by others, it's difficult (if not impossible) to control how they will use the data or enter new data. For example, on a spreadsheet, even though one cell might need a date and another a currency value to make sense, a user might easily enter character data in error.

Some spreadsheet programs allow you to define a "database" area within a spreadsheet to help you manage the information you need to produce the desired result. However, you are still constrained by the basic storage limitations of the spreadsheet program, and you still don't have much control over what's entered in the rows and columns of the database area. Also, if you need to handle more than number and character data, you might find that your spreadsheet program doesn't understand such things as pictures or sounds.

An RDBMS allows you to define the kind of data you have and how the data should be stored. You can also usually define rules that the RDBMS can use to ensure the integrity of your data. In its simplest form, a *validation rule* might ensure that you can't accidentally store alphabetic characters in a field that should contain a number. Other rules might define valid values or ranges of values for your data. In the most sophisticated systems, you can define the relationship between collections of data (usually tables or files) and ask the RDBMS to ensure that your data remains consistent. For example, you can have the system automatically check to ensure that every order entered is for a valid customer.

With Access, you have complete flexibility to define your data (as text, numbers, dates, times, currency, Internet links, pictures, sounds, documents, spreadsheets), to define how Access stores your data (string length, number precision, date/time precision), and to define what the data looks like when you display or print it. You can define simple or complex validation rules to ensure that only accurate values exist in your database. You can request that Access check for valid relationships between files or tables in your database.

Because Access is a state-of-the-art application for Microsoft Windows, you can use all the facilities of Dynamic Data Exchange (DDE*)*, *object linking and embedding* (part of Microsoft's OLE technology), and *Active*X custom controls. DDE lets you execute functions and send data between Access and any other Windows-based application that supports DDE. You can also make DDE connections to other applications using macros or Microsoft Visual Basic for Applications (VBA). OLE is an advanced Windows capability that, in part, allows you to link objects to or embed objects in your Access database. Objects include pictures, graphs, spreadsheets, and documents from other Windows-based applications that also support OLE.

2.6 The Uses of Microsoft Access

Microsoft Access has all the features of a classic relational database management system (RDBMS)—and more. Access is not only a powerful, flexible, and easy-to-use RDBMS, it is also a complete database application development facility. You can use Access to create and run under the Microsoft Windows operating system an application tailored to your data management needs. You can limit, select, and total your data by using queries. You can create forms for viewing and changing your data. You can also use Access to create simple or complex reports. Both forms and reports "inherit" the properties of the underlying table or query, so in most cases you need to define such things as formats and validation rules only once.

Among the most powerful features of Access are the wizards that you can use to create tables and queries and to customize a wide variety of forms and reports simply by selecting options with your mouse. Access also includes wizards that help you analyze your table design, import spreadsheet or text data, improve database performance.

Access includes a comprehensive programming language, Microsoft Visual Basic for Applications (VBA) that you can use to create very robust "production" applications that can be shared by many users.

Finally, you get all of these development facilities not only for working with an Access database but also to attach to and work with data stored in many other popular formats. You can build an Access application to work directly with dBase files, with Paradox and Microsoft FoxPro databases, and with any SQL database that supports the Open Database Connectivity (ODBC) standard. You can also easily import and export data as text, word processing files, or spreadsheet files.

2.6.1 Creating a Microsoft access 2000 Database

Once you have started Access 2000, the computer prompts you to either open an existing database or create a new one.

You can choose from several pre-made databases, or you can use the Database Wizard to assist you in creating your database

2.6.2 Creating Tables-Table Wizard

Once the database is created, the table must be created next, since all other database objects are dependent on the information in the table. The table is created by clicking on the new button in the database window. Users must create fields for the table. They must also decide what sort of data can be entered in each of those fields. This information is called the DATA TYPE.

2.6.3 Creating Tables-Table Structure

It can become necessary to alter the structure of your table to accommodate the records and fields of your table. You have the ability to freeze columns in a table, resize columns, or move them to new locations.

2.7 Forms

Forms, like tables, allow you the opportunity to look at your records. A table, however, allows you to look at all of the records at once. (or as many as can fit on the screen) A form shows only one record at a time. The real purpose of forms is to allow data entry. Forms can be formatted with different setups, different ways to enter data, and can allow a more thorough look at individual records.

2.7.1 Forms-Design View

If you need to make changes to your form, you can click on the Design View button, much like you do for changing tables. You will then get a screen similar to this, including a Toolbox for adding items.

If you wish to move a field in the form, you can click and drag the item. You can also use the vertical or horizontal rulers to select several items in the form. When a field is selected, it will have boxes surrounding it called resize handles.

2.7.2 Auto Form

You can choose Auto Form out of the New Object list. This will automatically create a form based on whichever table or query was selected at the time.

2.7.3 Form Wizard

To have more design control over your form, choose Form from the New Object list and choose to use the Form Wizard, When working in a database, most users will only need a certain group of records at any one time. Some may want to look at sales transactions for the previous week, while others may only need to look at a list of addresses in downtown Chicago. Whatever the reason, database programs have the ability to search for any record a user needs and manipulate the data in many different ways. These are accomplished by creating a QUERY. Once tables have been established inside of a database, a person can develop a query to select a group of fields from those tables, select only records that adhere to a specific set of criteria, and ready those records for use in a report.

2.8 Wildcard Characters and Comparison Operators

In our query a record would only be selected if the field contained exactly Chicago. However, an exact match may be too restrictive of a search. Therefore, you can use either WILDCARD CHARACTERS or COMPARISON OPERATORS to assist you in finding close matches. The following two tables show the symbols used for each of these,

WILDCARD CHARACTERS		
SYMBOL &	WHAT IT DOES	SAMPLE

USAGE		
M*	Finds all records that begin with M, no limit on extra characters	Murphy, McCall, Martin
*M	Finds all records that end with M, no limit on extra characters	Tim, mom, I'm
M	Contains the letter M anywhere within it	Time, McDonalds, gemstone
?	Single character that can represent any other character	F?nd: Find, Fund, Fend, Fond
??	Two characters that can represent any two consecutive characters	F??d: Food, Ford, Fred

COMPARISON OPERATORS	
SYMBOL	WHAT IT REPRESENTS
>	Greater Than
<	Less Than
>=	Greater Than or Equal To
<=	Less Than or Equal To
<>	Not Equal To

2.8.1 Queries-And/Or Criteria

Queries in Access have the ability to analyze many criteria at the same time. However, a decision must be made: does a record have to meet all criteria to be selected or only one? This is where AND and OR criteria come in. When you have two or more criteria, you must separate one criterion from another with either the word AND or the word OR. If you separate them with an AND, this indicates to the computer that ALL criteria must be met for the record to be selected. If only one criteria is met, the record will NOT be selected. On the other hand, if you separate them with an OR, this indicates to the computer that ANY of the criteria can be met for the record to be selected. The only way a record would not be selected in an OR is if the record match no criteria at all.

2.8.2 Working with Records-Finding Records

Most people will need to find certain records in a table at some point. Although a query is normally used for this, a user can also do a search inside the table if they are only searching for records that meet a single standard.

2.8.3 Working with Records-Editing Records

Once you have found the record you wish to edit, you make changes in much the same way as a word processor: you click after the area you wish to correct, use the [BACKSPACE] key to erase the old information, and type in the new information. You can also use your arrow keys to have an entire cell highlighted if you wish to replace the entire contents of that cell.

The entire Description field cannot be seen. It would not do much good to resize the column, since this field has a Memo data type. Therefore, you should move to the cell you wish to edit, then hit the [SHIFT] + [F2] keys to zoom in on the cell.

You can use the [DELETE] key to delete the contents of one cell in the record. Also, like in many applications, Access has a Find and Replace feature to make the same correction several times easily.

2.8.4 Working with Records-Moving Around in a Table

When you are in a table, you will need to move back and forth in the records to view them or make editing changes. In the table, like many application programs, the mouse will allow you to click on any record or field that you want. There are also some keystrokes that can move you around in a table. These keystrokes can move you between cells in the table or within the same cell if you are editing the cell.

KEYSTROKE SEQUENCE	ACTION
[ENTER]	Advances the active cell one to the right or to the next record if at end of the record
[TAB]	Same as [ENTER] key
[SHIFT]+[TAB]	Moves the active cell back one to the left or to the previous record if at the beginning of a record
Arrow Keys	Move to the next active cell or record in that direction
[HOME]	Moves to the first field in a record or to the beginning of

	the active cell when editing
[END]	Moves to the last field in a record or to the end of the active cell when editing
[CTRL]+[HOME]	Moves to the first field in the first record of the table
[CTRL]+[END]	Moves to the last field in the last record of the table
[F5]	Allows user to type the record number to go to
[SHIFT]+[F2]	Zooms in on the active cell to see its contents

There are also **NAVIGATION BUTTONS** at the bottom of the table. These buttons allow a user to enter a new record, advance one record, advance to the end of the table, go back one record, or go back to the beginning of the table. NOTE: if the mouse is held down on either button to advance or go back one record, the active cell will move forward or back several records rapidly. Near the Navigation buttons, the computer also indicates the current record number and the number of total records in the table.

2.8.5 Working with Records-Sorting Records

Since the only location that records can be added is at the end of the table, it can become necessary to rearrange the records. This is easily accomplished in Access with the use of the SORT ASCENDING and SORT DESCENDING buttons on the toolbar. To sort the records based on the order of a particular field, you simply have to have the active cell somewhere in that field, and then click on the desired button. If you wish to sort by more than one column, highlight the columns by which you want to sort, then click on either button. (NOTE: the columns MUST be next to each other. If they are not, either move them or use a query to perform the sort.) The computer will sort by the data in the column to the left.

2.8.6 Working with Records-Adding and Deleting Records

The only location where a new record can be added to a table is at the end of the table. It is usually very easy to reach the end of the table unless the table has thousands of records in it. The end of a table can be reached instantly by clicking on the new button on either the toolbar or at the bottom of the table. When you add a record, like editing records, you can cancel the changes provided the active cell stays on the record. Once it moves off, the computer saves the database.

KEYSTROKE SEQUENCE	**ACTION**

[CTRL] + [']	Inserts the same value as the cell directly above it (Duplicate Command)
[CTRL] + [;]	Inserts the current date into the cell
[CTRL] + [SHIFT] + [:]	Inserts the current time into the cell
[CTRL] + [ALT] + [SPACEBAR]	Inserts the default value of the field into the cell

2.8.7 Deleting Records

If there are records that need to be deleted, you have the ability to select one or several rows of records. To do this, place your pointer in the row selector boxes to the left of the record(s) to be deleted, then click (or click-and-drag) so that those rows are highlighted. Then, simply hit the [DELETE] key on your keyboard. NOTE: Delete operations CANNOT be undone. The computer will prompt you to verify you want the records deleted.

2.9 ONE IMPORTANT NOTE REGARDING TABLES IN A DATABASE

Since most databases contain hundreds, even thousands, of records, it is critical that work be saved as often as possible in case there is a problem with the computer. To that end, Access has an automatic save feature. A user can enter or edit a record as often as they wish and still reject the changes made, provided the active cell remains on the same record. Once the user moves up or down to another record, the computer automatically re-saves the database. Since the only location that records can be added is at the end of the table, it can become necessary to rearrange the records. This is easily accomplished in Access with the use of the SORT ASCENDING and SORT DESCENDING buttons on the toolbar. To sort the records based on the order of a particular field, you simply have to have the active cell somewhere in that field, and then click on the desired button. If you wish to sort by more than one column, highlight the columns by which you want to sort, then click on either button.ord become permanent.

CHAPTER THREE : FLOW CHARTS

3.1 Main menu

In these flow charts you can see all form in the mean menu

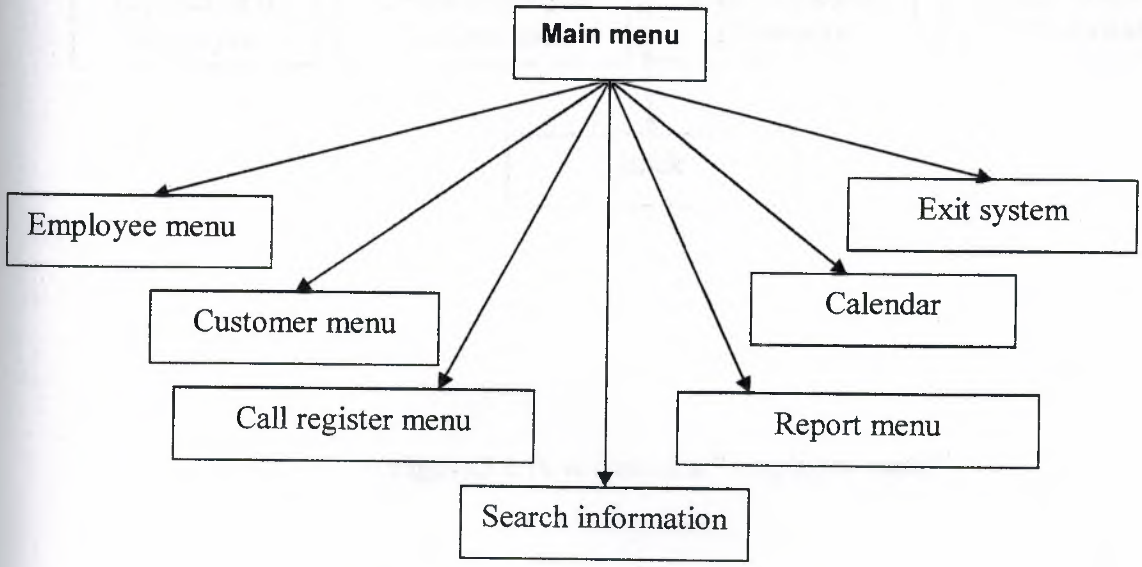


Figure3.1 flow chat of a " main menu"

3.2 Employee menu

In these flow charts you can see all form in the employee menu

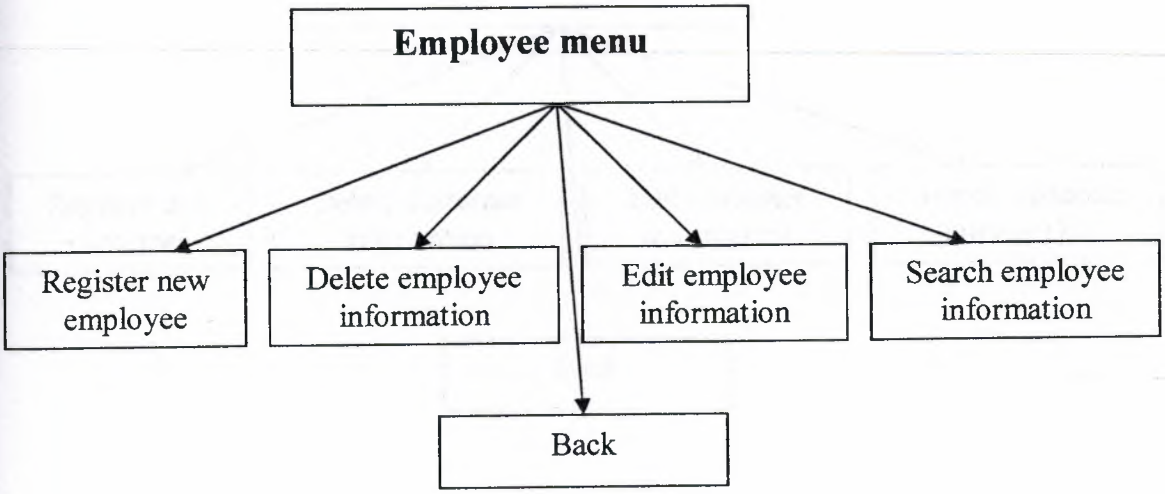

Employee menu
Register new employee
Delete employee information
Edit employee information
Search employee information
Back

Figure3.2 flow chat of a " employee menu"

3.3 Customer menu

In these flow charts you can see all form in the customer menu

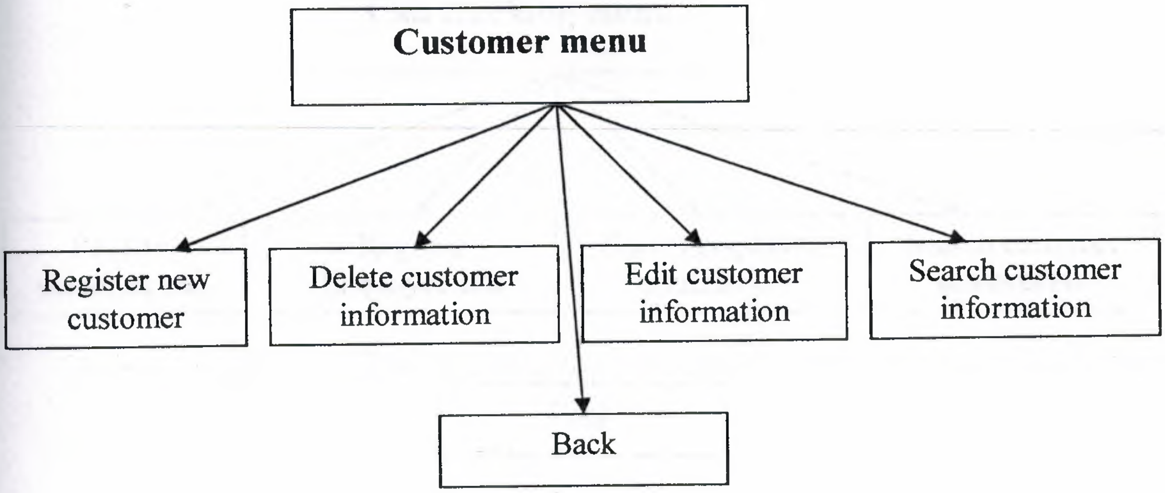

Customer menu
Register new customer
Delete customer information
Edit customer information
Search customer information
Back

Figure3.3 flow chat of a " customer menu"

3.4 Call tracking menu

In these flow charts you can see all form in the call tracking menu

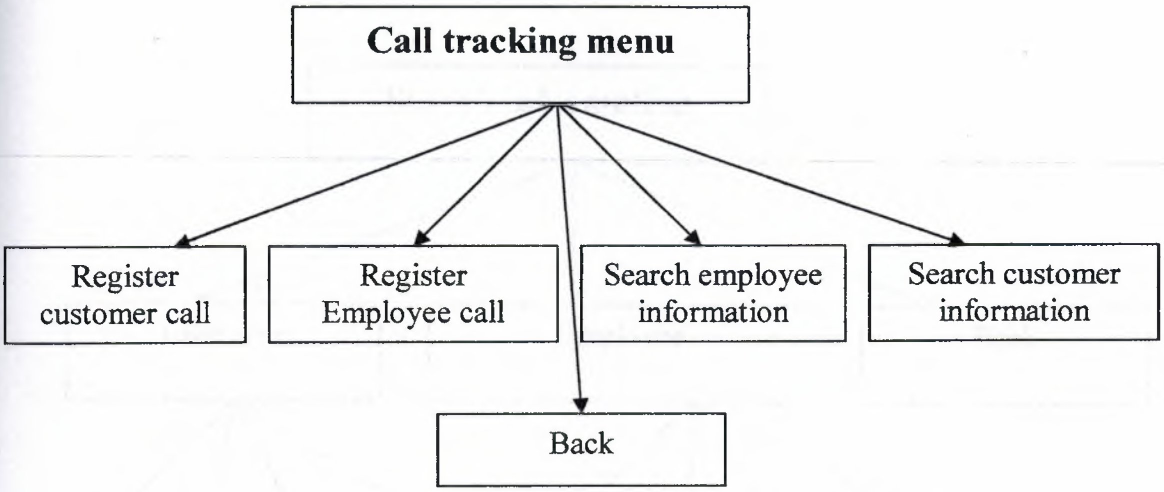


Figure3.4 flow chat of a " call tracking menu"

3.5 Search menu

In these flow charts you can see all form in the search information

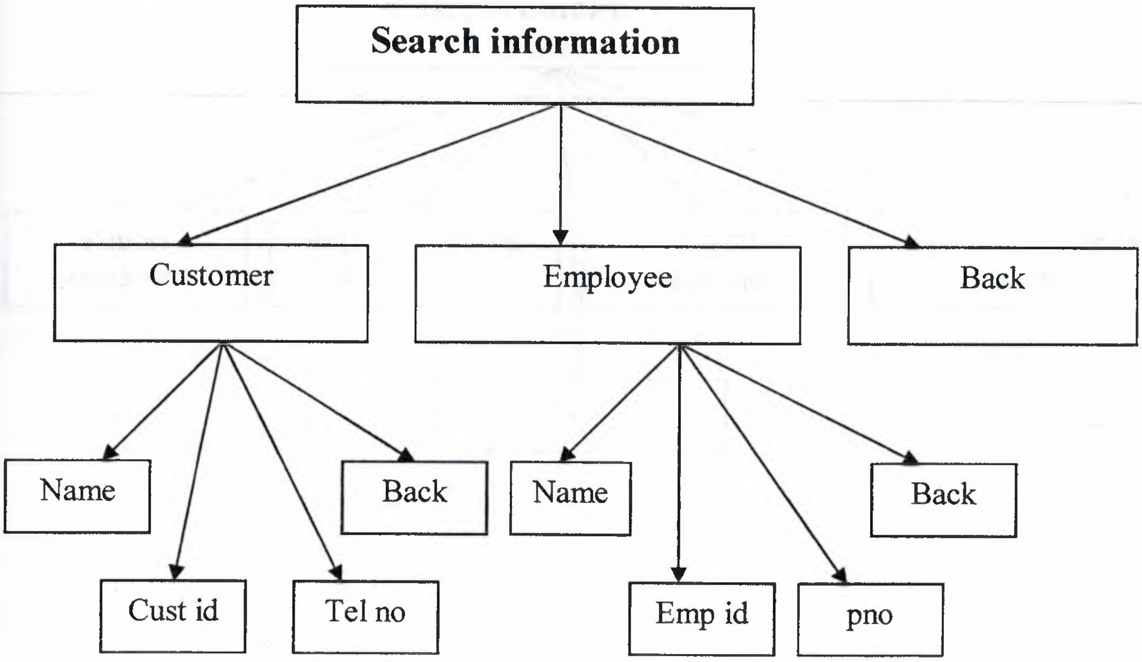


Figure3.5 flow chat of a " search information"

3.6 Report menu

In these flow charts you can see all form in the report menu

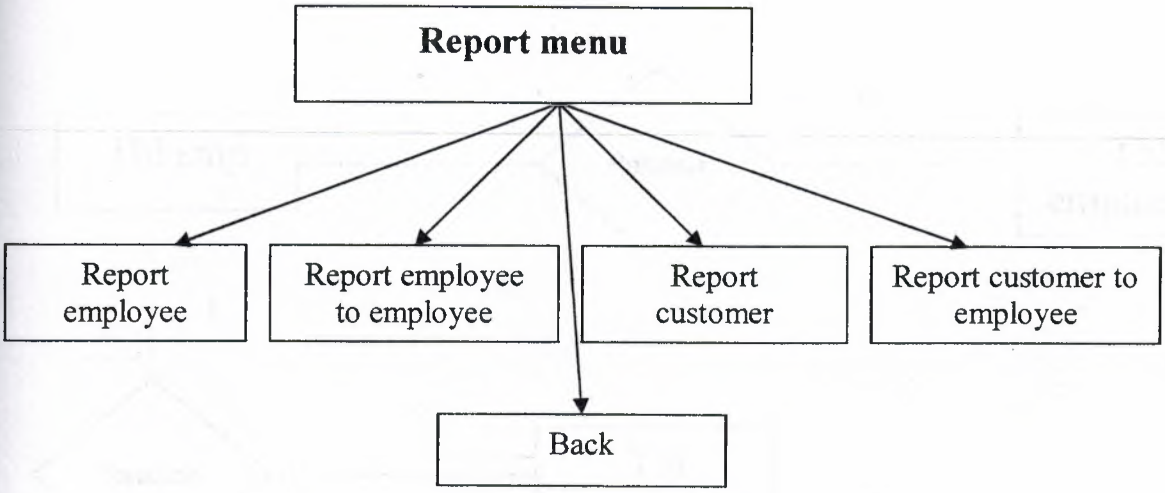


Figure3.6 flow chat of a " report menu"

3.7 Relation ship

In these flow charts you can see the relation ship of project "call management system"

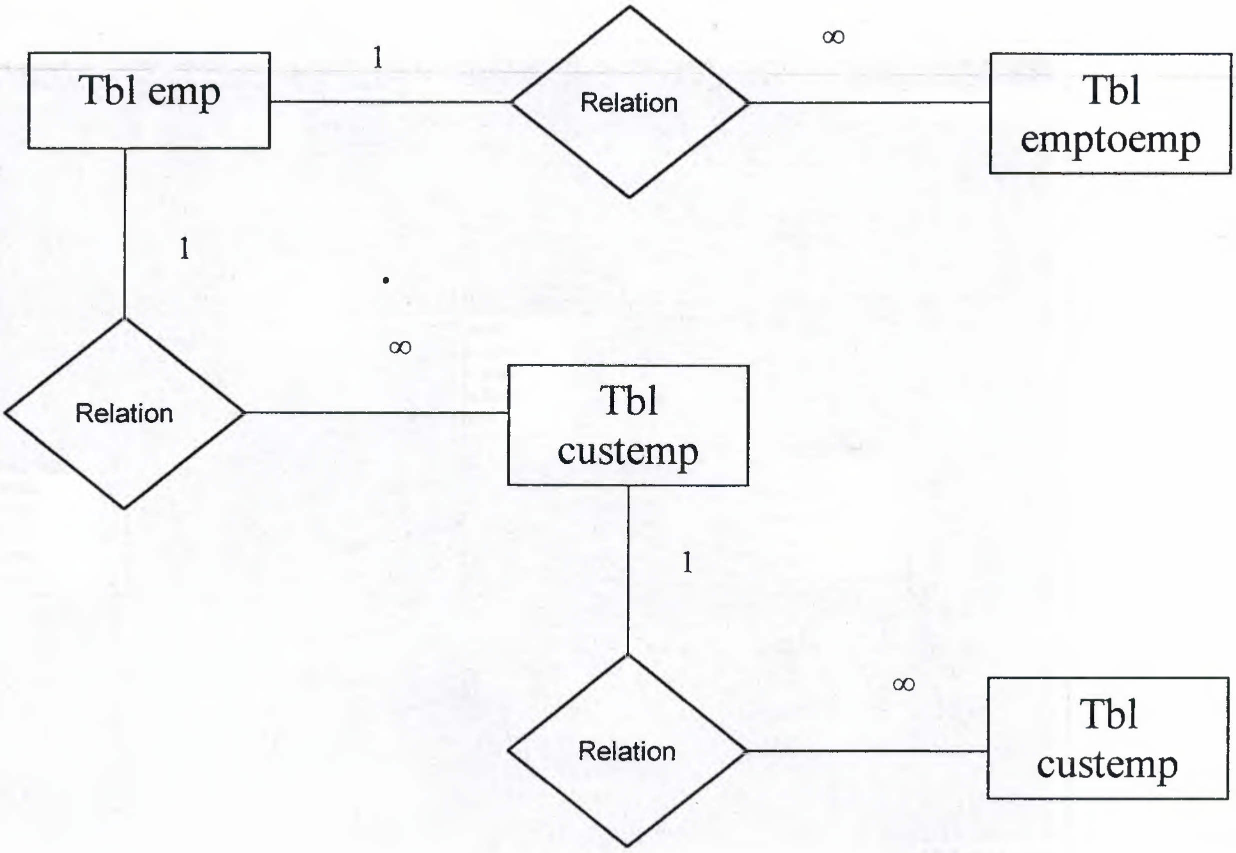


figure3.7 flow chat of a " relation ship"

3.7.1 Relation call management system

This is a relation ship of project “call management system”

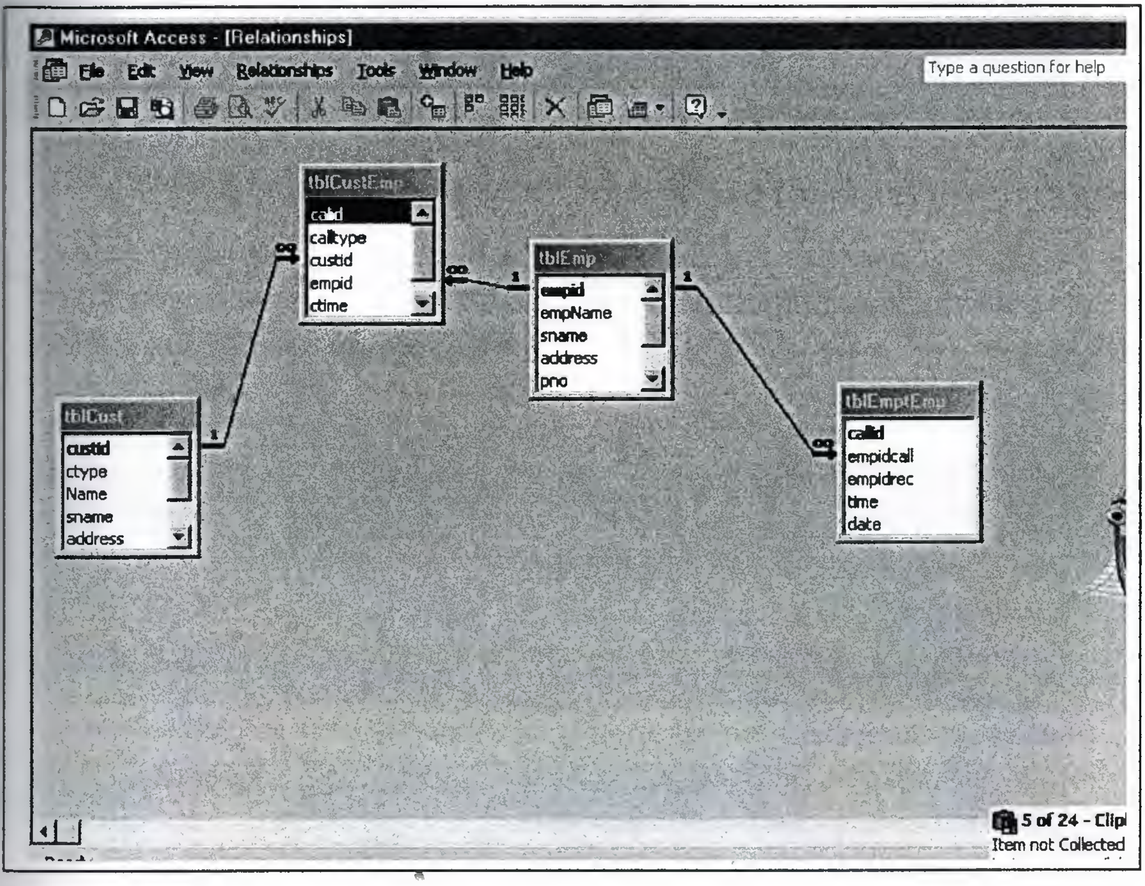
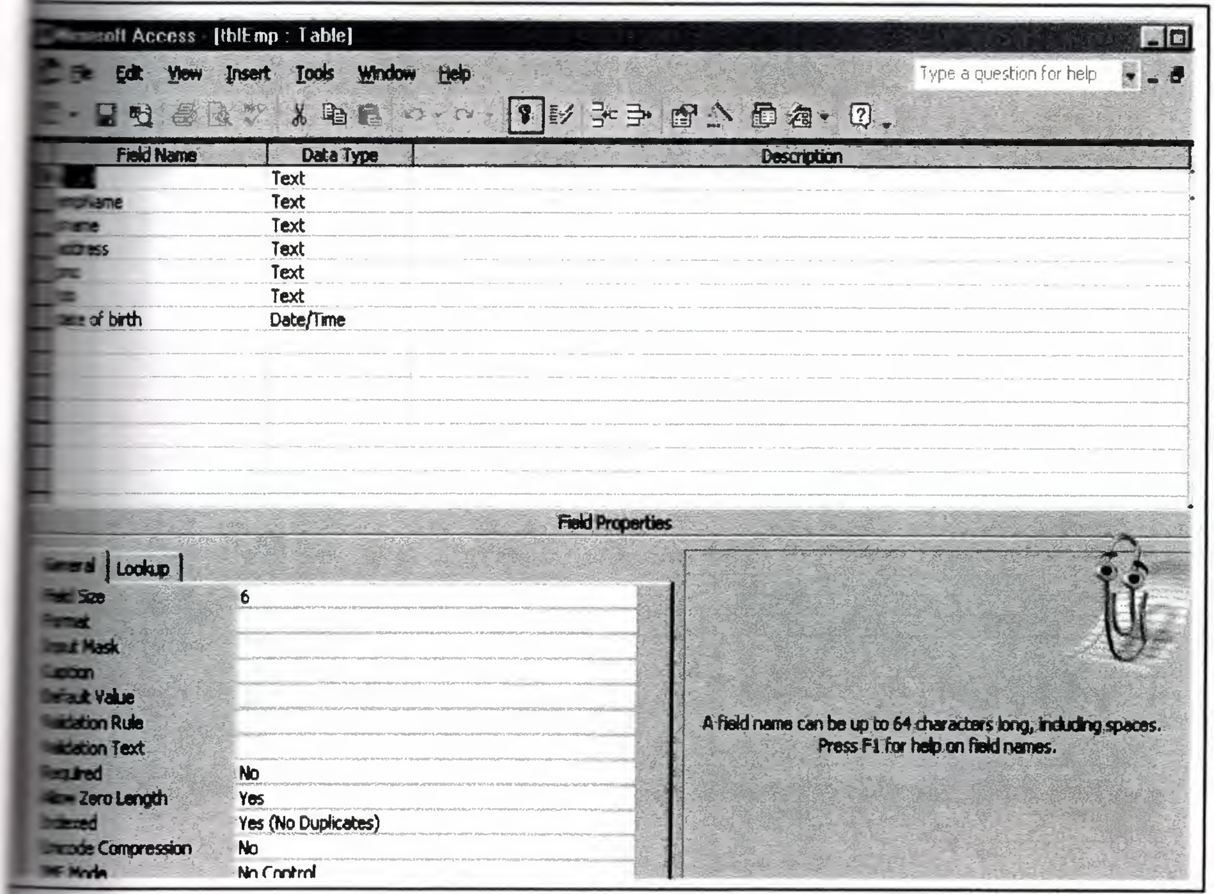


figure3.7.1 relation ship of a ” call management system”

CHAPTER FOUR : OPERATION OF MICROSOFT ACCESS

4.1.1 Table Employee

In this table you can see every thing about employee information



Microsoft Access - [tblEmp : Table]
File Edit View Insert Tools Window Help
Type a question for help

Field Name	Data Type	Description
[illegible]	Text	
empName	Text	
name	Text	
address	Text	
[illegible]	Text	
[illegible]	Text	
date of birth	Date/Time	

Field Properties

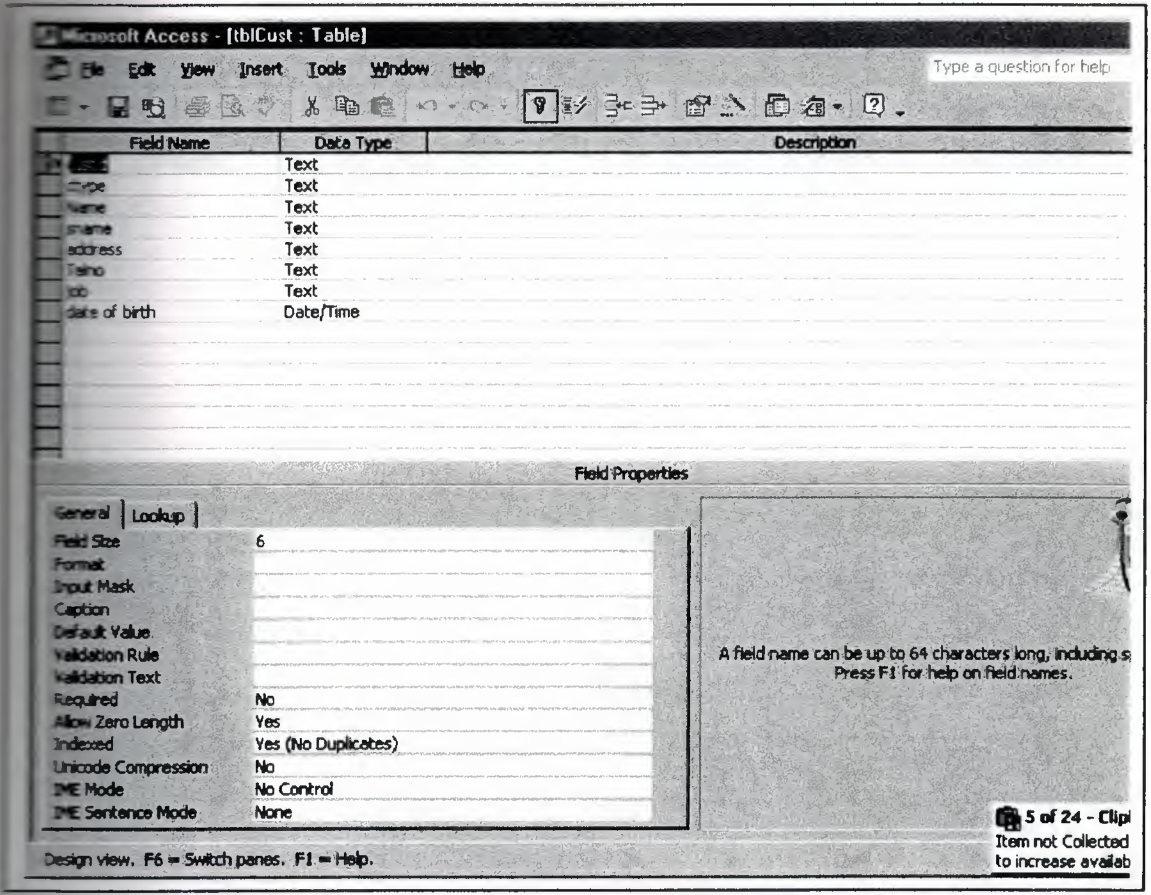
Field Size	6
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	Yes
Indexed	Yes (No Duplicates)
Unicode Compression	No
IME Mode	No Control

A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.

Figure 4.1.1 table "employee"

4.1.2 Table Customer

In this table you can see every thing about customer information



Field Name	Data Type	Description
[illegible]	Text	
ctype	Text	
Name	Text	
sname	Text	
address	Text	
Telno	Text	
job	Text	
date of birth	Date/Time	

Field Size	6
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	Yes
Indexed	Yes (No Duplicates)
Unicode Compression	No
IME Mode	No Control
IME Sentence Mode	None

A field name can be up to 64 characters long, including s
Press F1 for help on field names.

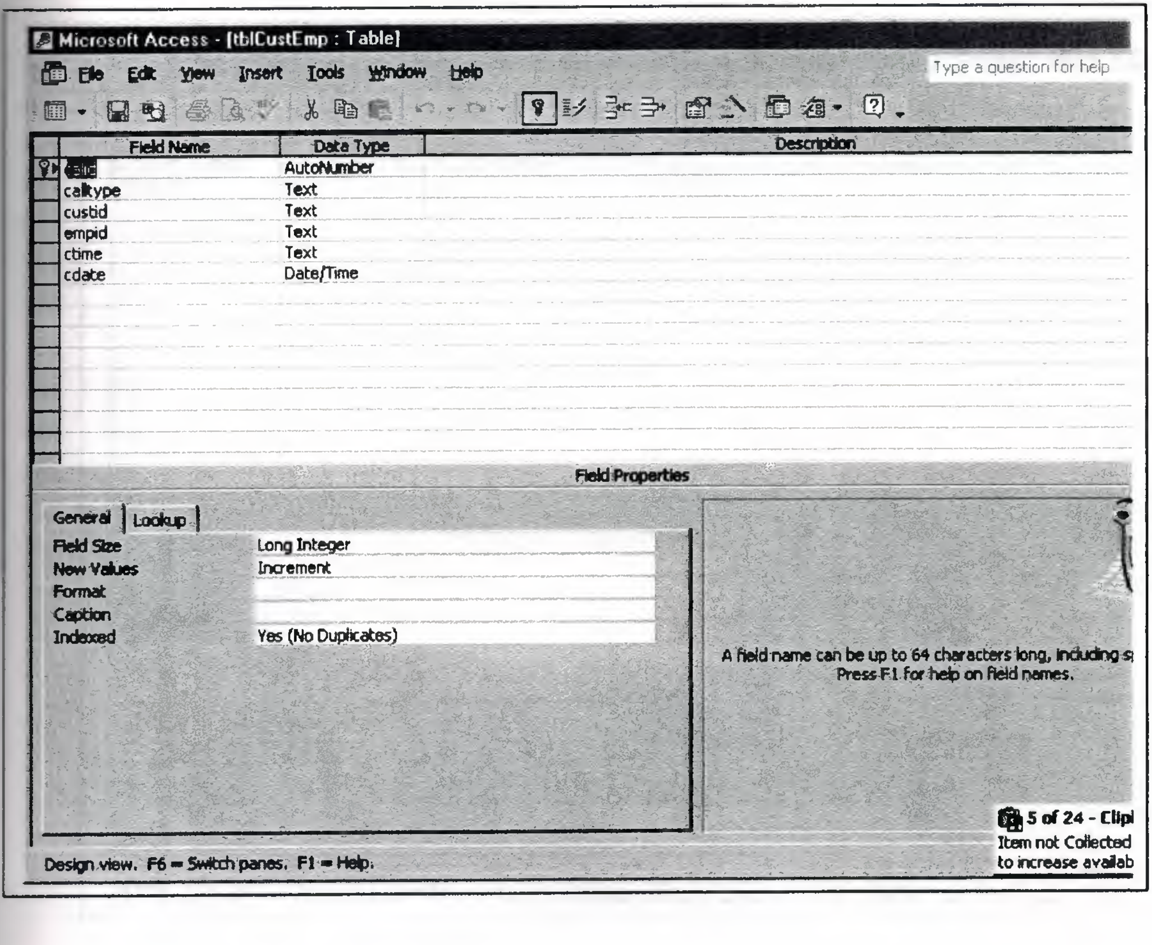
5 of 24 - Clip
Item not Collected
to increase availab

Design view. F6 = Switch panes. F1 = Help.

Figure 4.1.2 table "customer"

4.1.3 Table Customer To Employee

In this table you can see every thing about employee information



Microsoft Access - [tblCustEmp : Table]

Field Name	Data Type	Description
callid	AutoNumber	
calltype	Text	
custid	Text	
empid	Text	
ctime	Text	
cdate	Date/Time	

Field Properties

General	
Field Size	Long Integer
New Values	Increment
Format	
Caption	
Indexed	Yes (No Duplicates)

A field name can be up to 64 characters long, including s
Press F1 for help on field names.

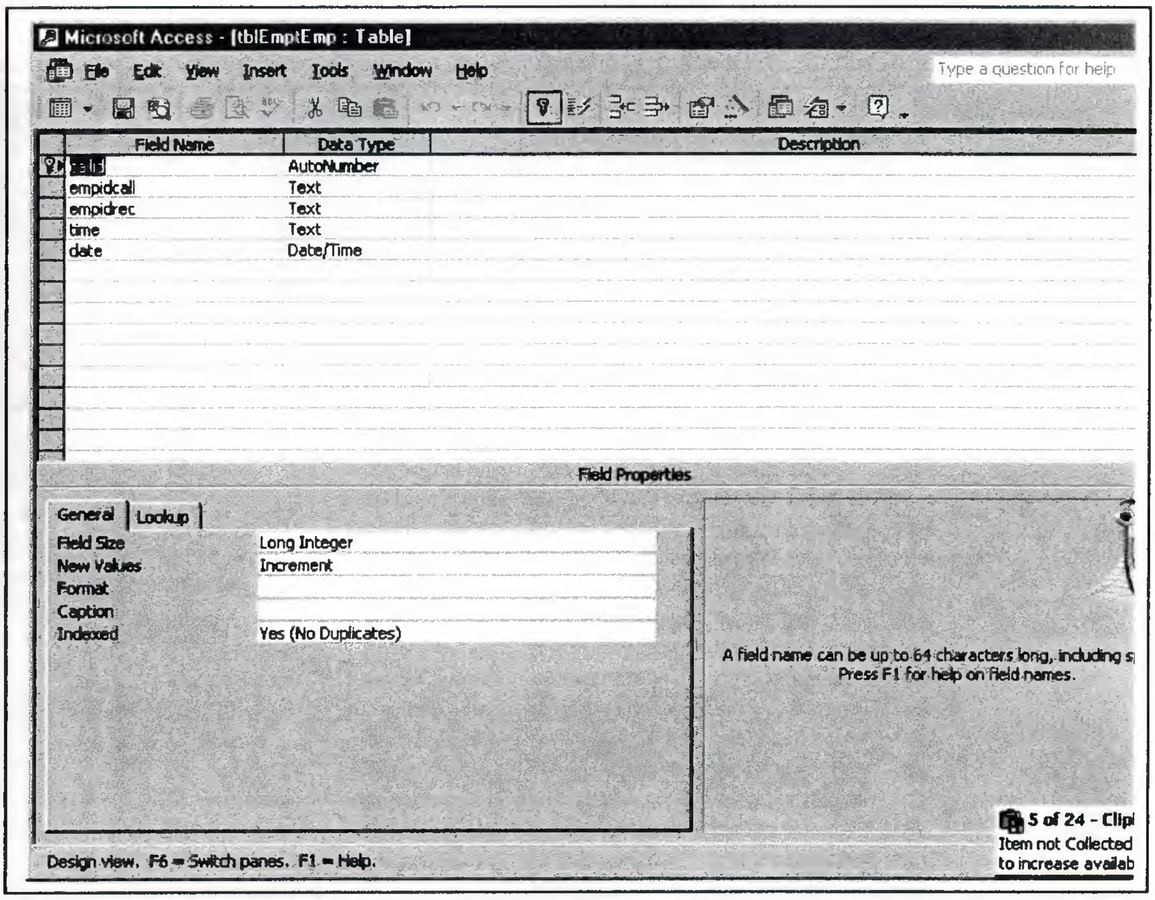
5 of 24 - Clip
Item not Collected
to increase availab

Design view. F6 = Switch panes. F1 = Help.

Figure 4.1.3 table “customer to employee”

4.1.4 Table Employee To Employee

In this table you can see every thing about employee to employee information



Microsoft Access - [tblEmptEmp : Table]
File Edit View Insert Tools Window Help
Type a question for help

Field Name	Data Type	Description
[illegible]	AutoNumber	
empidcall	Text	
empidrec	Text	
time	Text	
date	Date/Time	

Field Properties
General
Lookup

Field Size	Long Integer
New Values	Increment
Format	
Caption	
Indexed	Yes (No Duplicates)

A field name can be up to 64 characters long, including s
Press F1 for help on field names.

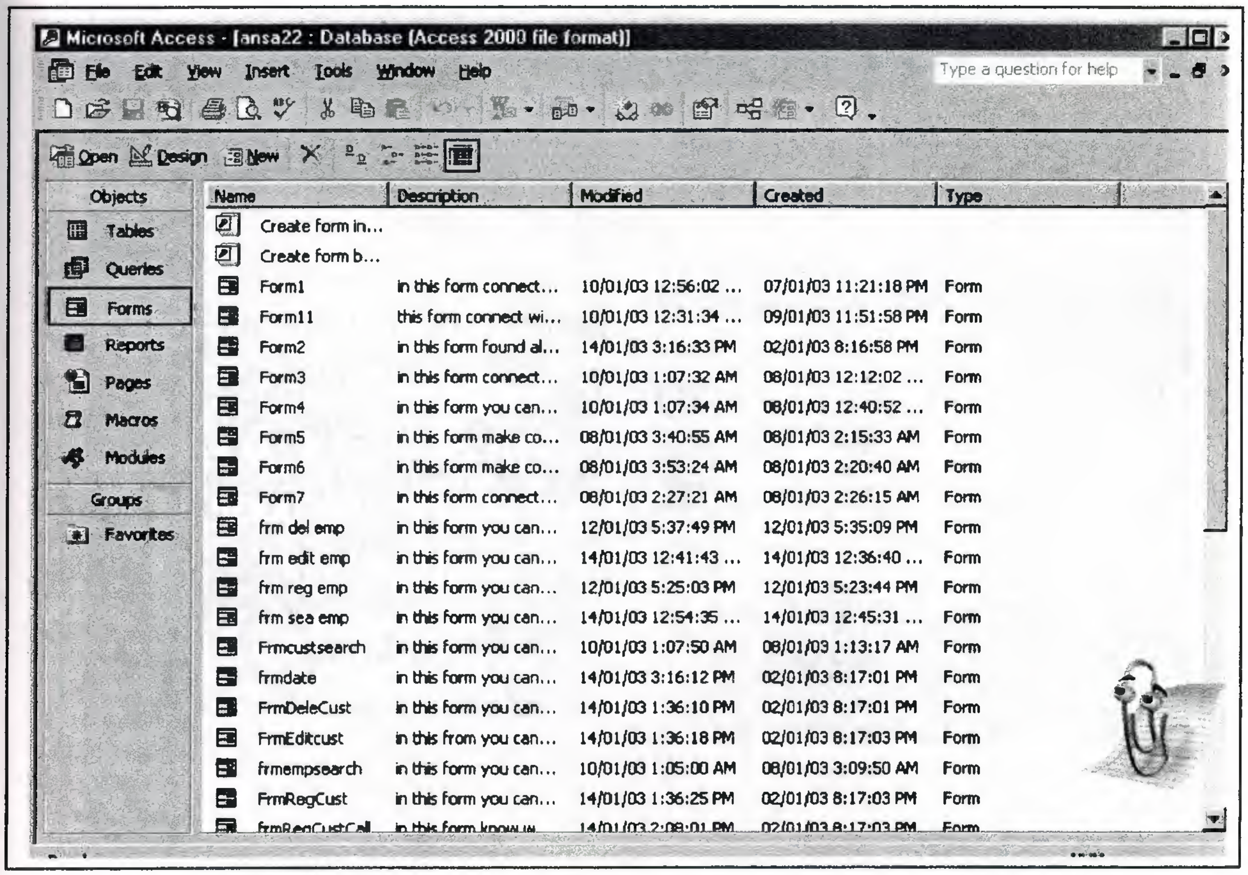
5 of 24 - Clip
Item not Collected
to increase availab

Design view. F6 = Switch panes. F1 = Help.

Figure 4.1.4 table “employee to employee”

4.2. Form main

In this form you can see all information about the form like (name, modified, created, type)



Name	Description	Modified	Created	Type
Create form in...				
Create form b...				
Form1	in this form connect...	10/01/03 12:56:02 ...	07/01/03 11:21:18 PM	Form
Form11	this form connect wi...	10/01/03 12:31:34 ...	09/01/03 11:51:58 PM	Form
Form2	in this form found al...	14/01/03 3:16:33 PM	02/01/03 8:16:58 PM	Form
Form3	in this form connect...	10/01/03 1:07:32 AM	08/01/03 12:12:02 ...	Form
Form4	in this form you can...	10/01/03 1:07:34 AM	08/01/03 12:40:52 ...	Form
Form5	in this form make co...	08/01/03 3:40:55 AM	08/01/03 2:15:33 AM	Form
Form6	in this form make co...	08/01/03 3:53:24 AM	08/01/03 2:20:40 AM	Form
Form7	in this form connect...	08/01/03 2:27:21 AM	08/01/03 2:26:15 AM	Form
frm del emp	in this form you can...	12/01/03 5:37:49 PM	12/01/03 5:35:09 PM	Form
frm edit emp	in this form you can...	14/01/03 12:41:43 ...	14/01/03 12:36:40 ...	Form
frm reg emp	in this form you can...	12/01/03 5:25:03 PM	12/01/03 5:23:44 PM	Form
frm sea emp	in this form you can...	14/01/03 12:54:35 ...	14/01/03 12:45:31 ...	Form
Frmcustsearch	in this form you can...	10/01/03 1:07:50 AM	08/01/03 1:13:17 AM	Form
frmdate	in this form you can...	14/01/03 3:16:12 PM	02/01/03 8:17:01 PM	Form
FrmDeleCust	in this form you can...	14/01/03 1:36:10 PM	02/01/03 8:17:01 PM	Form
FrmEditcust	in this from you can...	14/01/03 1:36:18 PM	02/01/03 8:17:03 PM	Form
frmempsearch	in this form you can...	10/01/03 1:05:00 AM	08/01/03 3:09:50 AM	Form
FrmRegCust	in this form you can...	14/01/03 1:36:25 PM	02/01/03 8:17:03 PM	Form
frmRegCustCall	in this form know w...	14/01/03 2:08:01 PM	02/01/03 8:17:03 PM	Form

Figure 4.2. form “main’

4.3 Form Mean Menu

In this form you can see the form of main menu

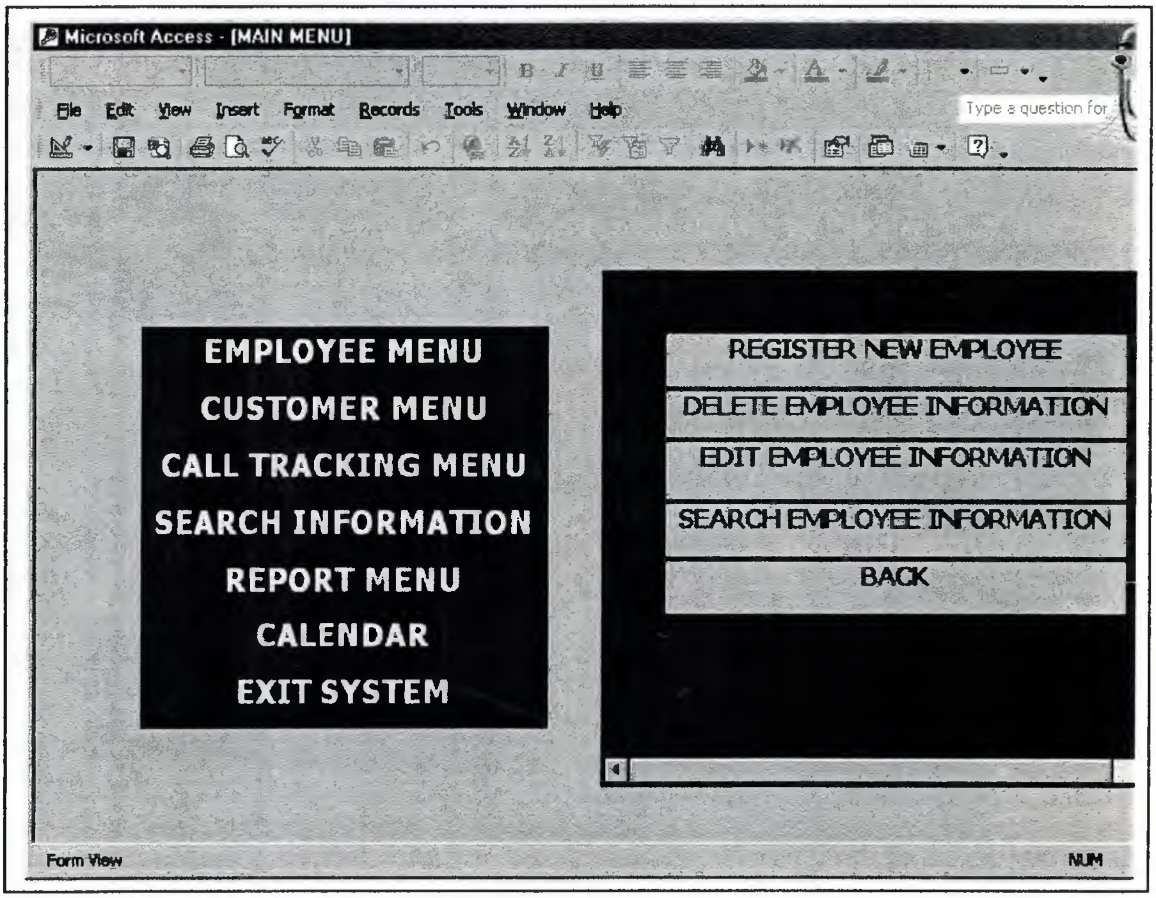
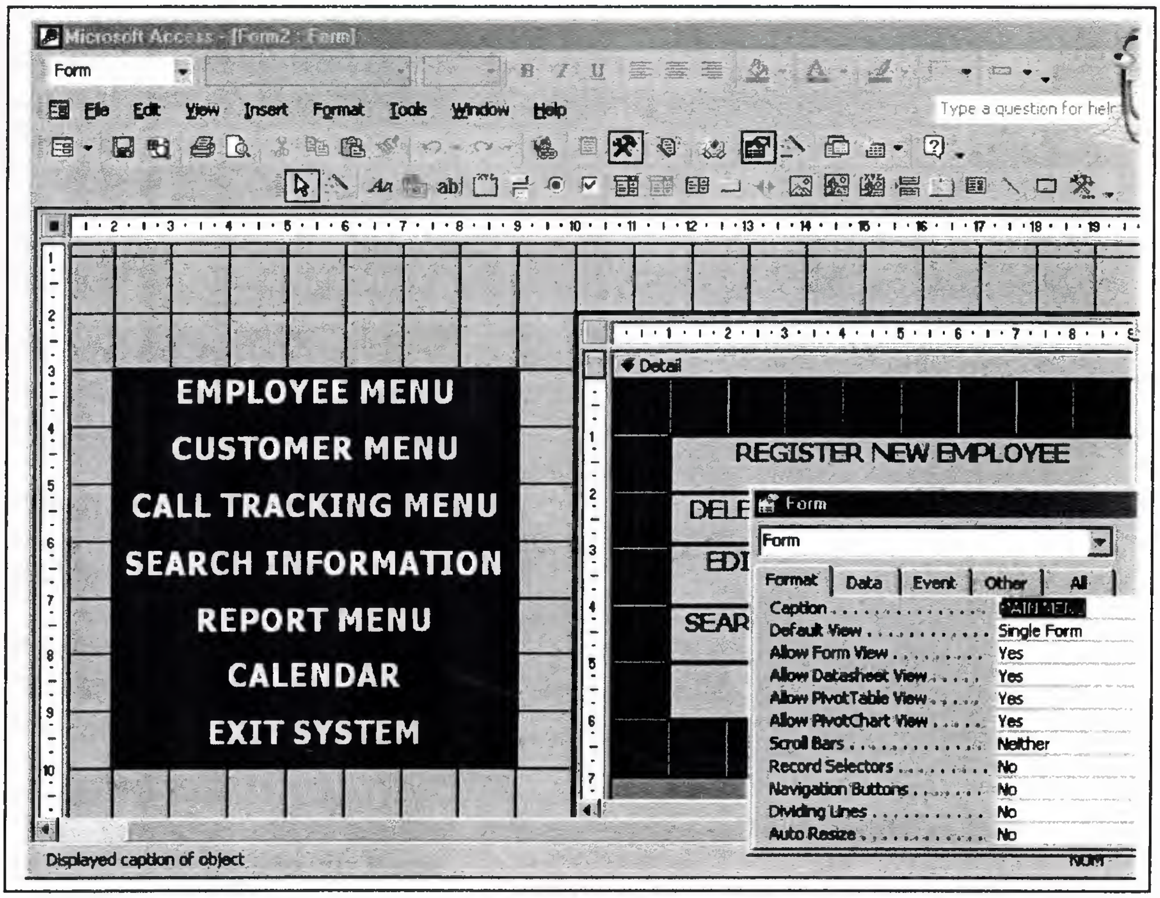

Microsoft Access - [MAIN MENU]
File Edit View Insert Format Records Tools Window Help
Type a question for
EMPLOYEE MENU
CUSTOMER MENU
CALL TRACKING MENU
SEARCH INFORMATION
REPORT MENU
CALENDAR
EXIT SYSTEM
REGISTER NEW EMPLOYEE
DELETE EMPLOYEE INFORMATION
EDIT EMPLOYEE INFORMATION
SEARCH EMPLOYEE INFORMATION
BACK
Form View
NUM

Figure 4.3 form "main menu"

4.3.1 Design Form Mean Menu

In this form you can see the design of main menu



Caption . . . . MAIN MENU
Default View . . . . Single Form
Allow Form View . . . . Yes
Allow Datasheet View . . . . . Yes
Allow PivotTable View . . . . . Yes
Allow PivotChart View . . . . . Yes
Scroll Bars . . . . Neither
Record Selectors . . . . No
Navigation Buttons . . . . No
Dividing Lines . . . . No
Auto Resize . . . . No

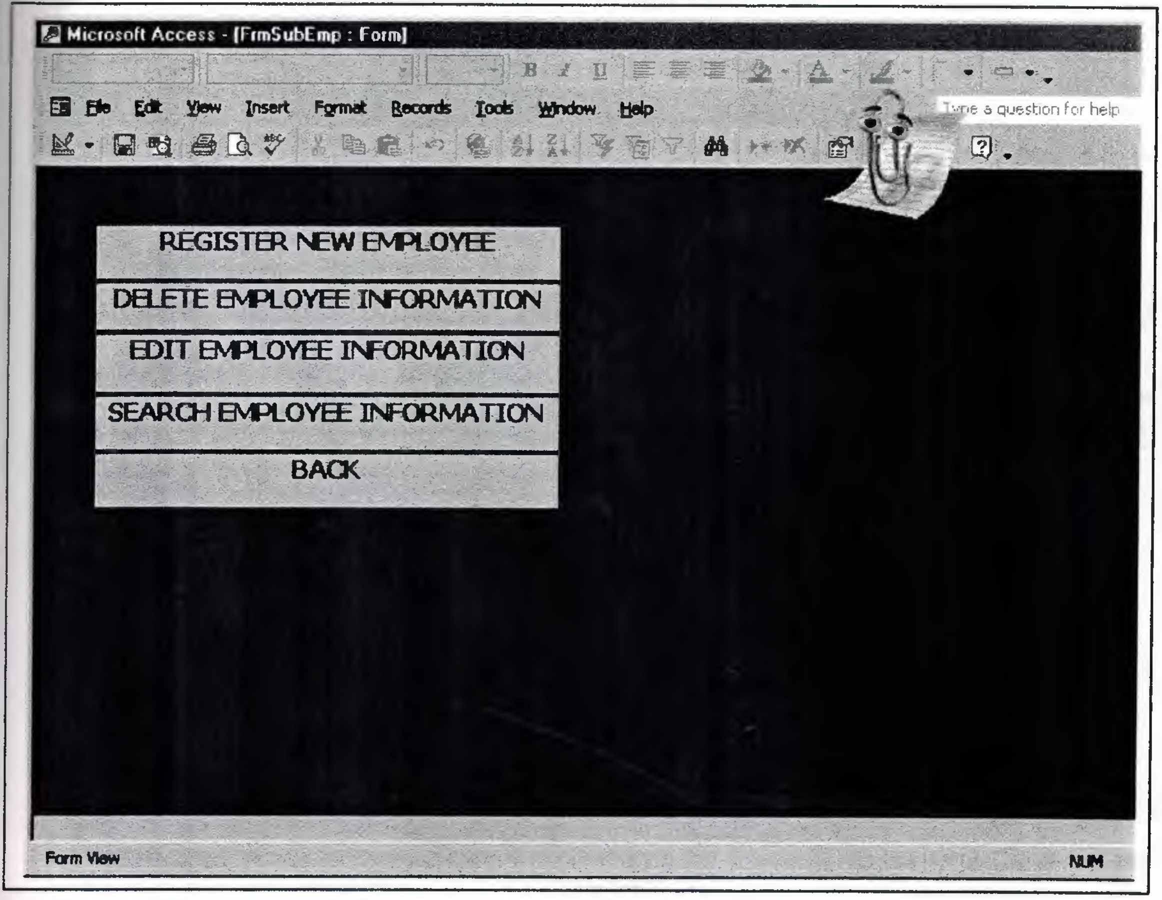
Figure 4.3.1 design form “main menu”

4.3.2 Code Maim Menu

```
Option Compare Database


Private Sub Label40_Click ()
Me.chld.SourceObject = "frmsubemp"
End Sub

Private Sub Label42_Click ()
Me.chld.SourceObject = "frmSubCust"
End Sub

Private Sub Label43_Click ()
Me.chld.SourceObject = "frmsubcall"
End Sub

Private Sub Label44_Click ()
Me.chld.SourceObject = "report"
End Sub

Private Sub Label45_Click ()
Me.chld.SourceObject = "search"
End Sub

Private Sub Label46_Click ()
Me.chld.SourceObject = "frmdate"
End Sub

Private Sub Label47_Click ()
If MsgBox("Dou want to exit this", vbExclamation + vbYesNo) = vbYes Then
DoCmd.Close acForm, Me.Name
 End If
End Sub
```

4.4. Employee Menu Form

In this form you can see the employee menu

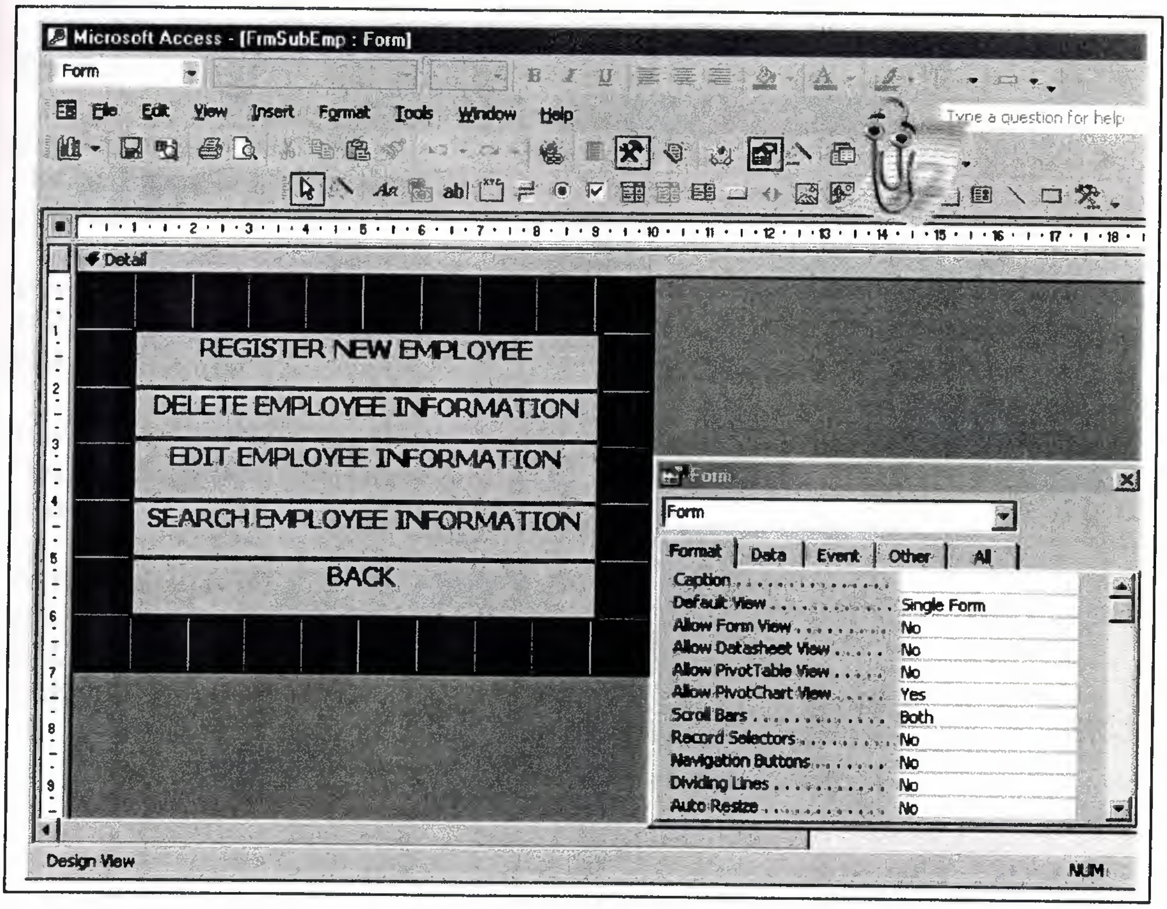


Microsoft Access - [FrmSubEmp : Form]
File Edit View Insert Format Records Tools Window Help
Type a question for help
REGISTER NEW EMPLOYEE
DELETE EMPLOYEE INFORMATION
EDIT EMPLOYEE INFORMATION
SEARCH EMPLOYEE INFORMATION
BACK
Form View
NUM

Figure 4.4. form “Employee menu”

4.4.1 Design Employee Menu Form

In this form you can see the design employee menu form


Microsoft Access - [FrmSubEmp : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
REGISTER NEW EMPLOYEE
DELETE EMPLOYEE INFORMATION
EDIT EMPLOYEE INFORMATION
SEARCH EMPLOYEE INFORMATION
BACK
Form
Form
Format Data Event Other All
Design View
NUM

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

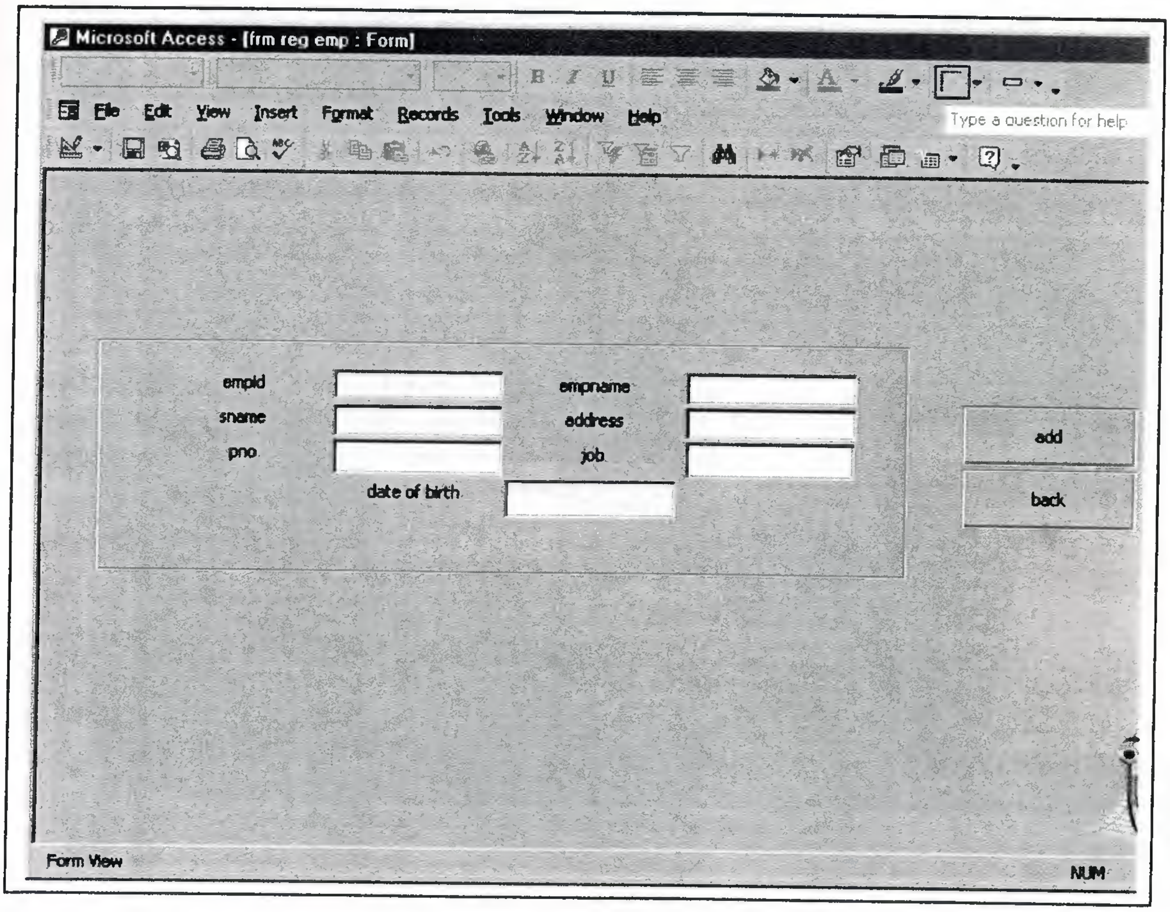
Figure 4.4.1 design “employee menu”

4.4.2 Code Employee Menu Form

```
Option Compare Database

Private Sub Label78_Click ()
DoCmd.OpenForm "frmregemp"
End Sub

Private Sub Label80_Click ()
DoCmd.OpenForm "frmdelemp"
End Sub

Private Sub Label81_Click ()
DoCmd.OpenForm "frmediemp"
End Sub

Private Sub Label82_Click ()
DoCmd.OpenForm "frmseaemp"
End Sub

Private Sub Label83_Click ()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, Me.Name
End Sub
```

4.5 Register New Employee Form

In this form you can see the employee menu



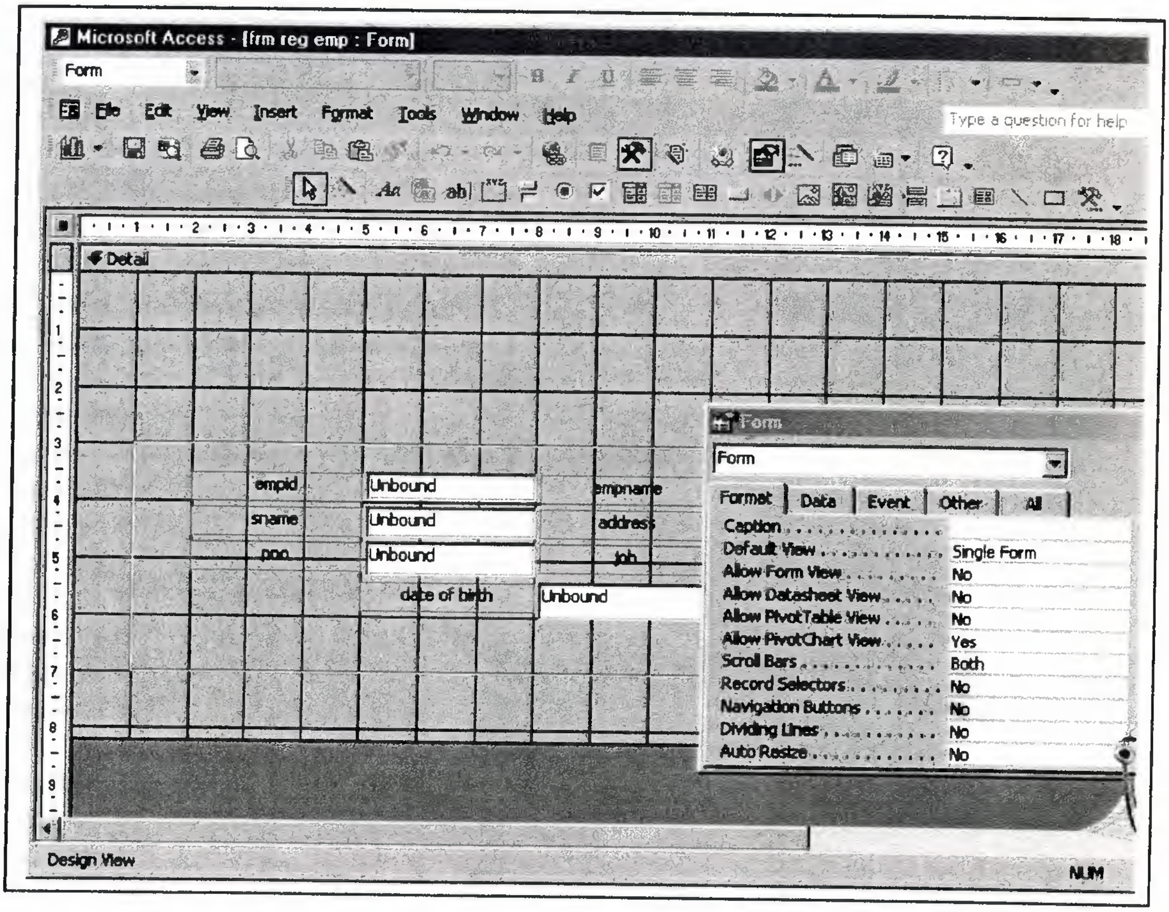
Microsoft Access - [frm reg emp : Form]

empid		empname	
sname		address	
pno		job	
date of birth			

Figure 4.5. form "Register new employee"

4.5.1 Design Register New Employee Form

In this form you can see the design register new employee



empid	Unbound
sname	Unbound
pno	Unbound
date of birth	Unbound

Form

Format | Data | Event | Other | All

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.5.1. design "register new employee "

4.5.2 Code Register New Employee Form

```
Private Sub Command30_Click ()
Dim db As DAO. Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select*from tblemp where empid='" & Me.txtcid & "'"
Set rs = db.OpenRecordset(s)
If rs.EOF And rs.BOF Then
rs.AddNew
rs.Fields("empid").Value = Me.txtcid
rs.Fields("empname").Value = Me.txtcn
rs.Fields("sname").Value = Me.txtcs
rs.Fields("address").Value = Me.txtca
rs.Fields("pno").Value = Me.txtcp
rs.Fields("job").Value = Me.txtcj
rs.Fields("date of birth").Value = Me.txtcd
rs.Update
MsgBox "new employee has been added", vbInformation
Me.txtcid = ""
Me.txtcn = ""
Me.txtcs = ""
Me.txtca = ""
Me.txtcp = ""
Me.txtcj = ""
Me.txtcd = ""
Else
MsgBox "this record is already present"
End If
End Sub

Private Sub Command33_Click()
DoCmd.OpenForm "frmsubemp"
DoCmd.Close acForm, Me.Name
End Sub
```

4.6 Delete Employee Information Form

In this form you can see the delete employee information

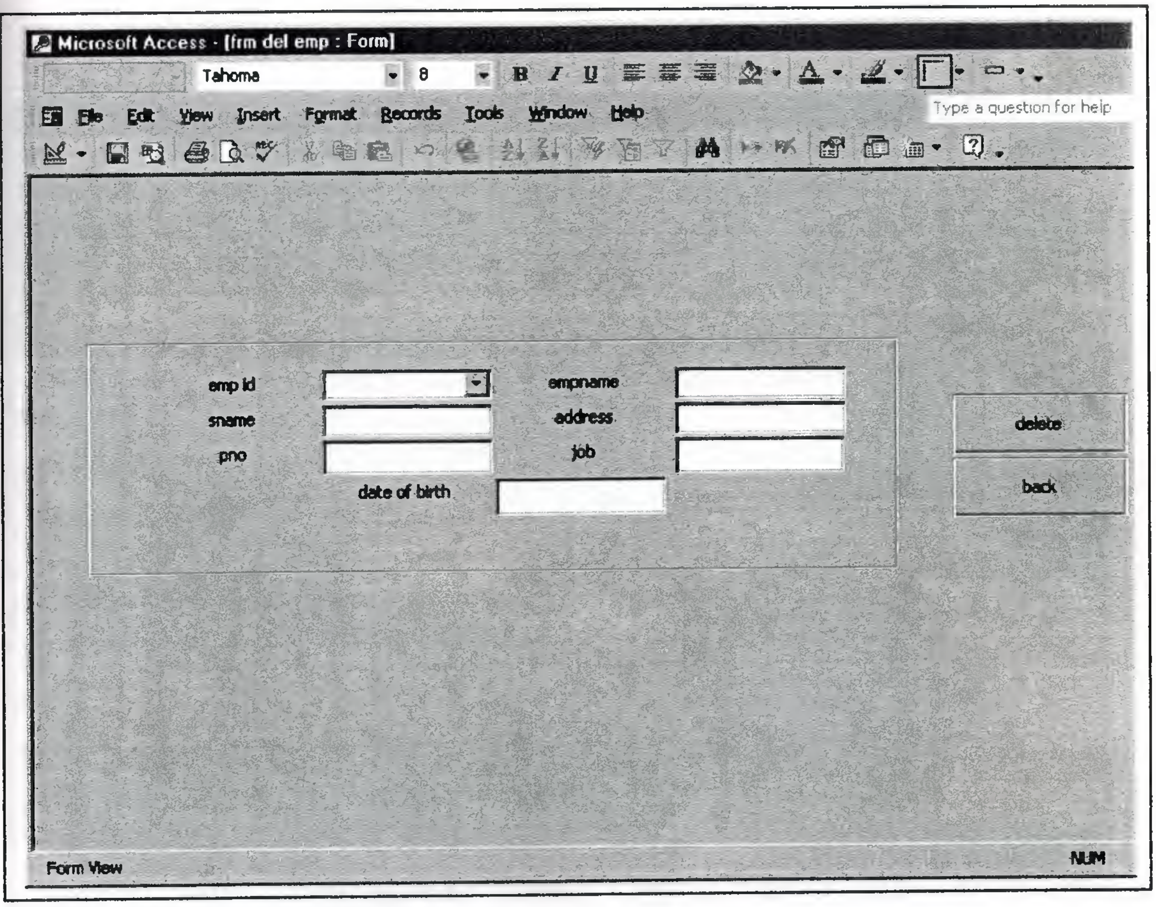
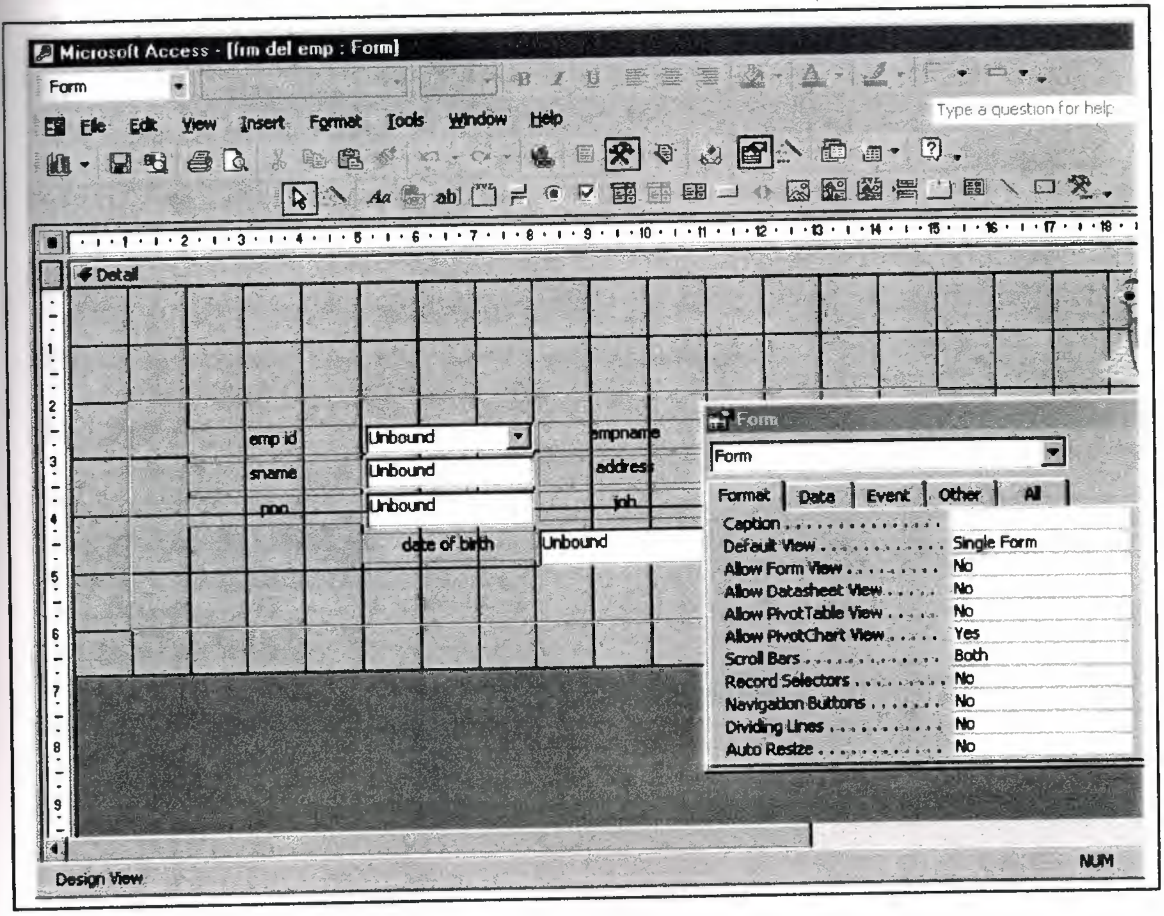


Figure 4.6. form “delete Employee information “

4.6.1 Design Delete Employee Information Form

In this form you can see the design delete employee information


Microsoft Access - [frm del emp : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
Design View
NUM

emp id Unbound

sname Unbound

pno Unbound

empname

address

job

date of birth Unbound

Form

Format | Data | Event | Other | All

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.6.1 design “delete employee information “

4.6.2 Code Delete Employee Information

```
Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select*from tblemp where empid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcn = rs.Fields("empname").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcp = rs.Fields("pno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth").Value
End Sub

Private Sub cmbcid_BeforeUpdate(Cancel As Integer)

End Sub

Private Sub Command24_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select*from tblemp where empid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
rs.MoveLast
rs.Delete
rs.MoveFirst
End Sub

Private Sub Command25_Click()
DoCmd.OpenForm "frmsubemp"
DoCmd.Close acForm, Me.Name

End Sub
```

4.7 Edit Employee Information Form

In this form you can see the delete employee information

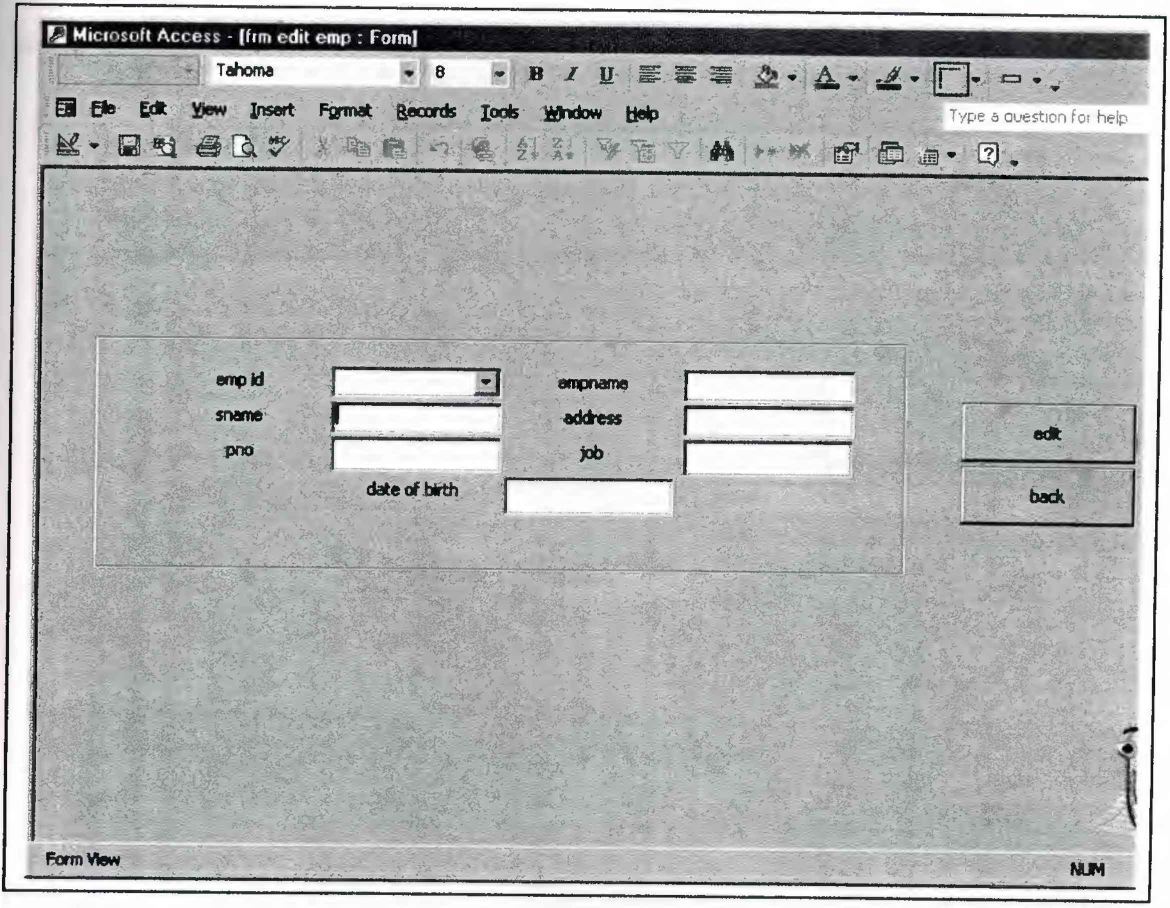
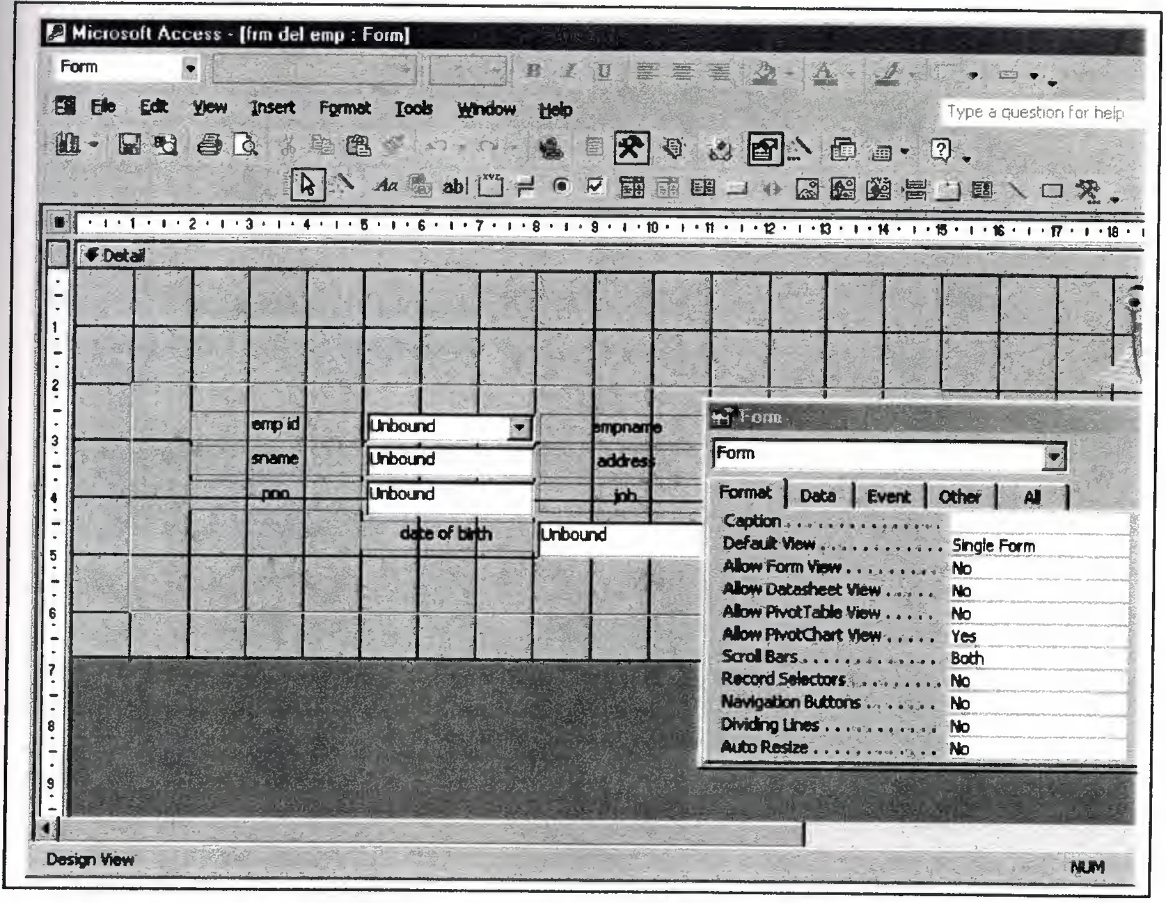


Figure 4.7. form “Edit Employee information “

4.7.1 Design Edit Employee Information Form

In this form you can see the design edit employee information



Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

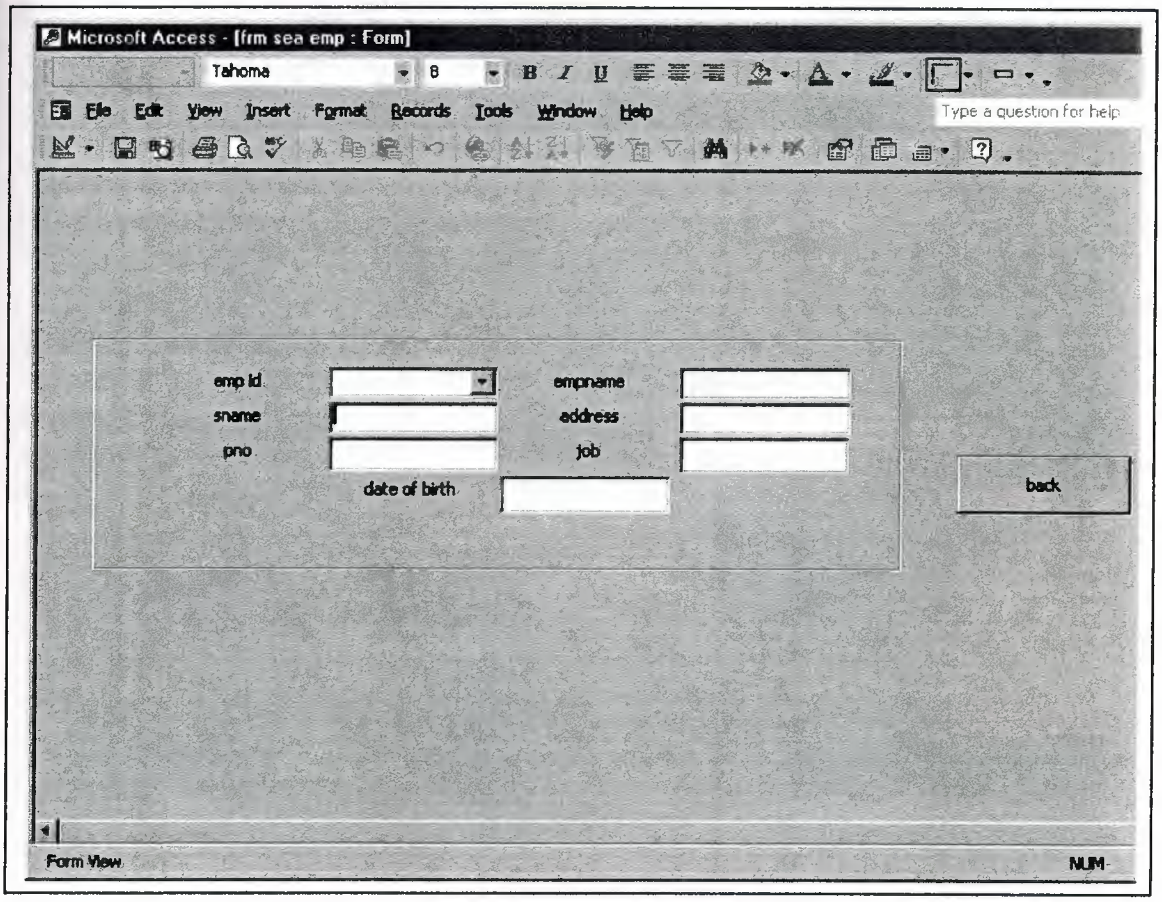
Figure 4.7.1 design "Edit Employee information "

4.7.2 Code Edit Employee Information

```
Option Compare Database
Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblemp where empid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcn = rs.Fields("empname").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcp = rs.Fields("pno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth")
End Sub

Private Sub Command15_Click()
DoCmd.OpenForm "frmSubEmp"
DoCmd.Close acForm, Me.Name

End Sub

Private Sub edit_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
s = "tblemp"
Set db = CurrentDb()
Set rs = db.OpenRecordset(s)
rs.edit
rs.Fields("empname").Value = Me.txtcn
rs.Fields("sname").Value = Me.txtca
rs.Fields("address").Value = Me.txtca
rs.Fields("pno").Value = Me.txtcp
rs.Fields("job").Value = Me.txtcj
rs.Fields("date of birth").Value = Me.txtcd
rs.Update
End Sub
```

4.8 Search Employee Information Form

In this form you can see the search employee information

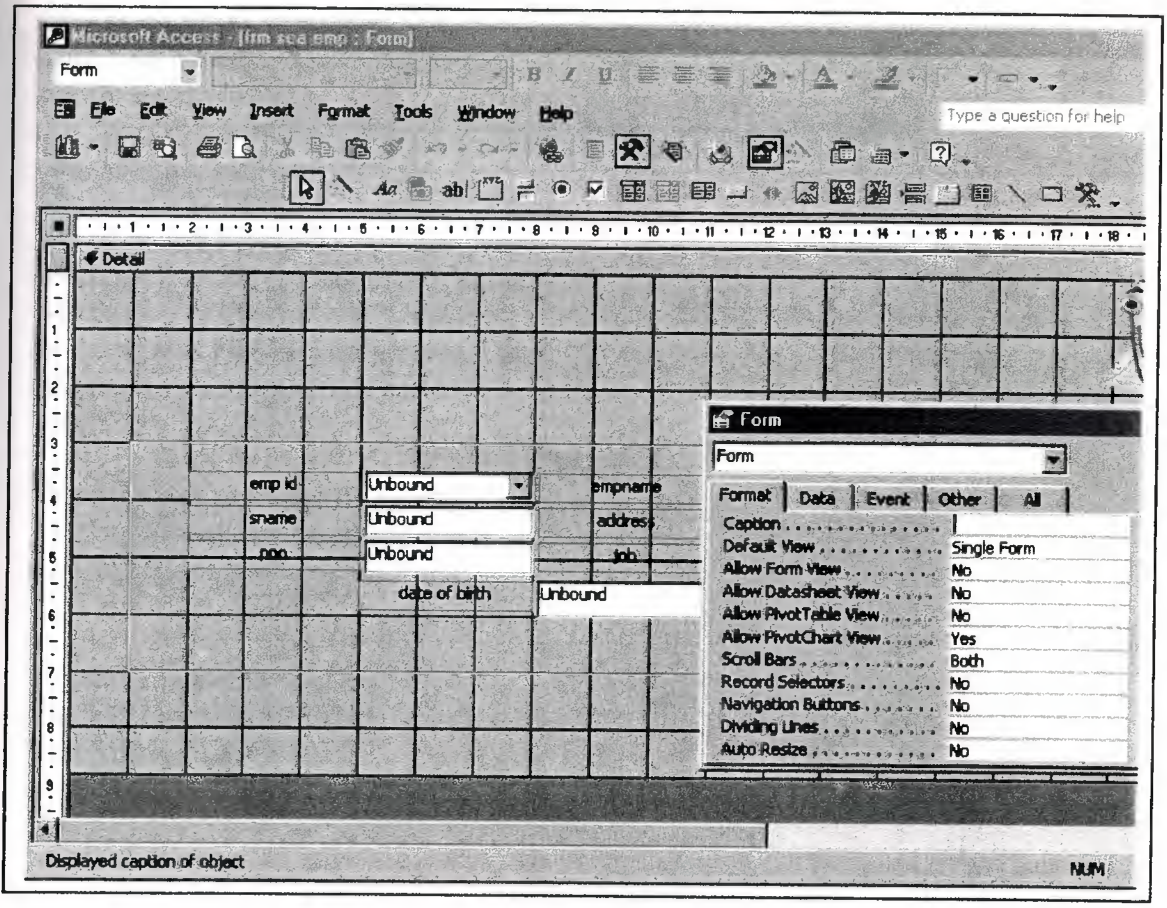


Microsoft Access - [frm sea emp : Form]
Tahoma
8
File Edit View Insert Format Records Tools Window Help
Type a question for help
emp id
empname
sname
address
pno
job
date of birth
back
Form View
NUM

Figure 4.8. form “search Employee information”

4.8.1 Design Search Employee Information Form

In this form you can see the design search employee information



Microsoft Access - [frm sea emp : Form]

emp id: Unbound

sname: Unbound

noo: Unbound

empname

address

job

date of birth: Unbound

Form

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Displayed caption of object

Figure 4.8.1 design “search employee information “

4.8.2 Code Search Employee Information

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblemp where empid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcn = rs.Fields("empname").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcp = rs.Fields("pno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth")
End Sub

Private Sub Command30_Click()
DoCmd.OpenForm "frmSubEmp"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Detail_Click()
DoCmd.OpenForm "frmsubEmp"
End Sub
```

4.9. Customer Menu Form

In this form you can see the customer menu

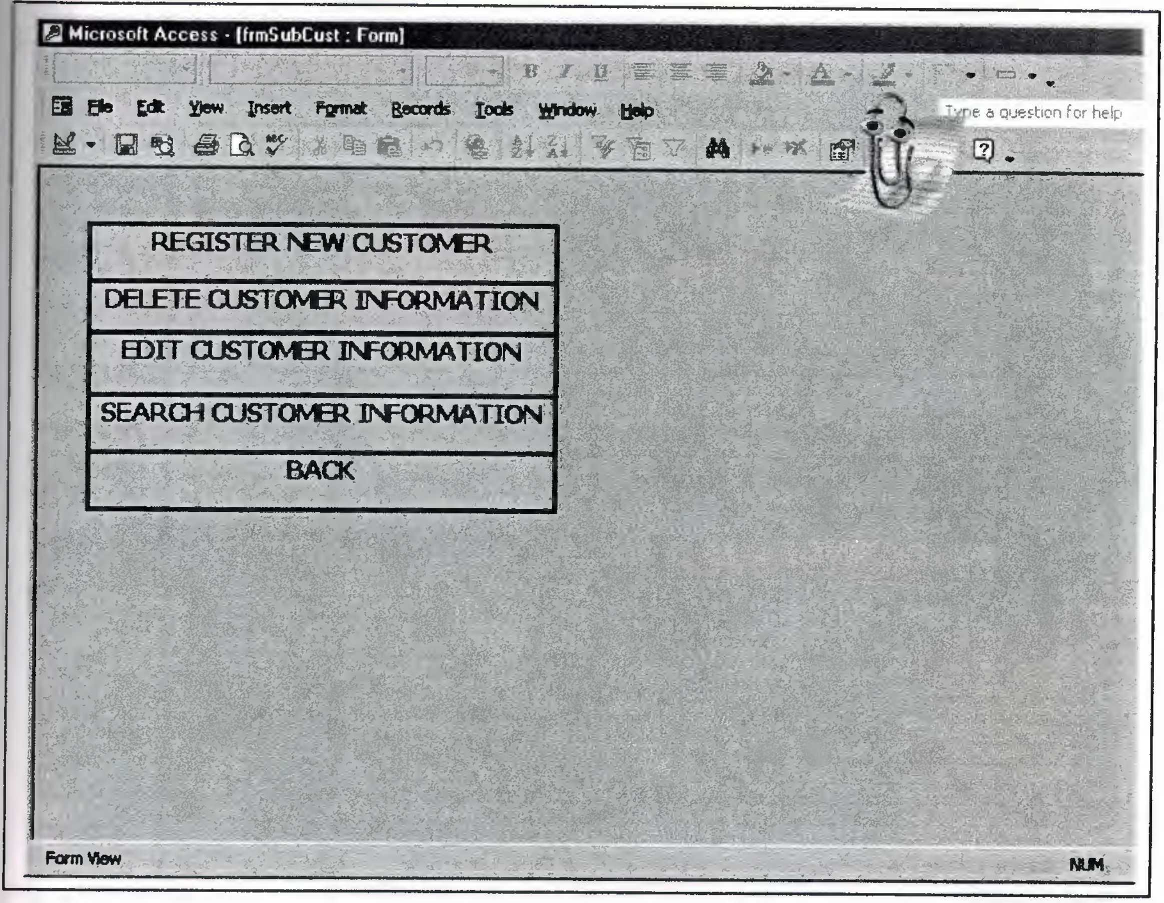
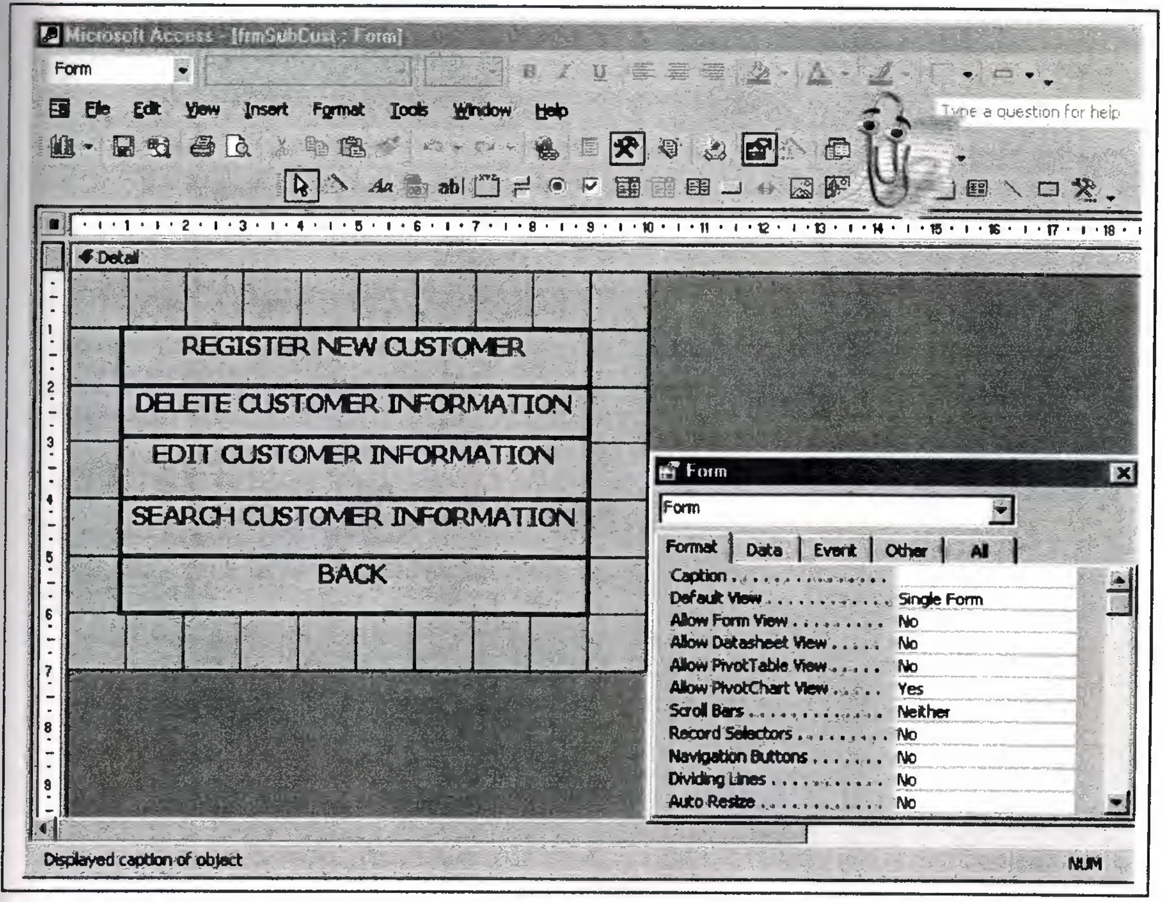

Microsoft Access - [frmSubCust : Form]
File Edit View Insert Format Records Tools Window Help
Type a question for help
REGISTER NEW CUSTOMER
DELETE CUSTOMER INFORMATION
EDIT CUSTOMER INFORMATION
SEARCH CUSTOMER INFORMATION
BACK
Form View
NUM

Figure 4.9 form "customer menu "

4.9.1 Design Customer Menu Form

In this form you can see the design customer menu


Microsoft Access - [frmSubCust : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
REGISTER NEW CUSTOMER
DELETE CUSTOMER INFORMATION
EDIT CUSTOMER INFORMATION
SEARCH CUSTOMER INFORMATION
BACK
Form
Form
Format Data Event Other All
Displayed caption of object
NUM

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Neither
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.9.1 design “customer menu “

4.9.2 Code Customer Menu

```
Option Compare Database

Private Sub Label78_Click ()
DoCmd.OpenForm "frmregcust"
End Sub

Private Sub Label80_Click ()
DoCmd.OpenForm "frmdelecust"
End Sub

Private Sub Label81_Click ()
DoCmd.OpenForm "frmeditcust"
End Sub

Private Sub Label82_Click ()
DoCmd.OpenForm "frmseacust"
End Sub

Private Sub Label83_Click ()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, Me.Name
End Sub
```

4.10. Register New Customer Form

In this form you can see the register new customer

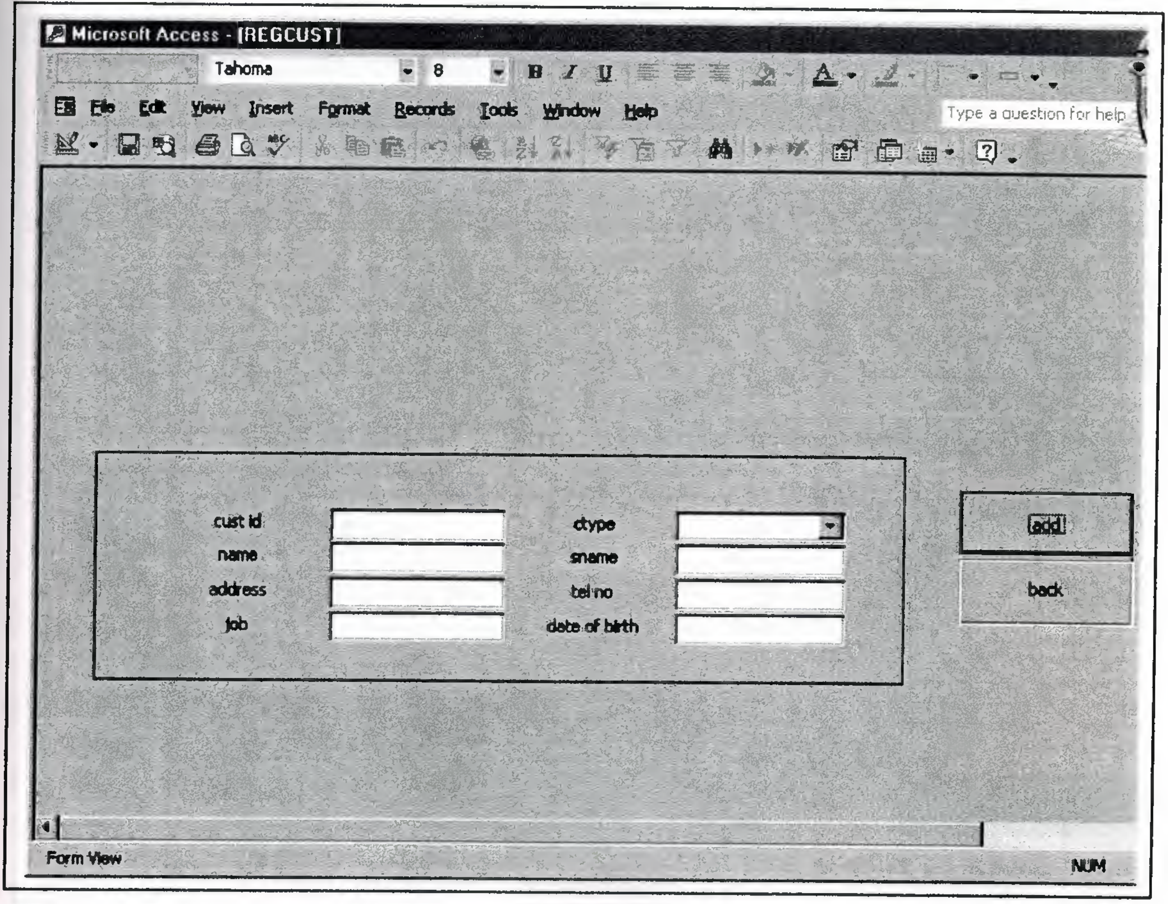
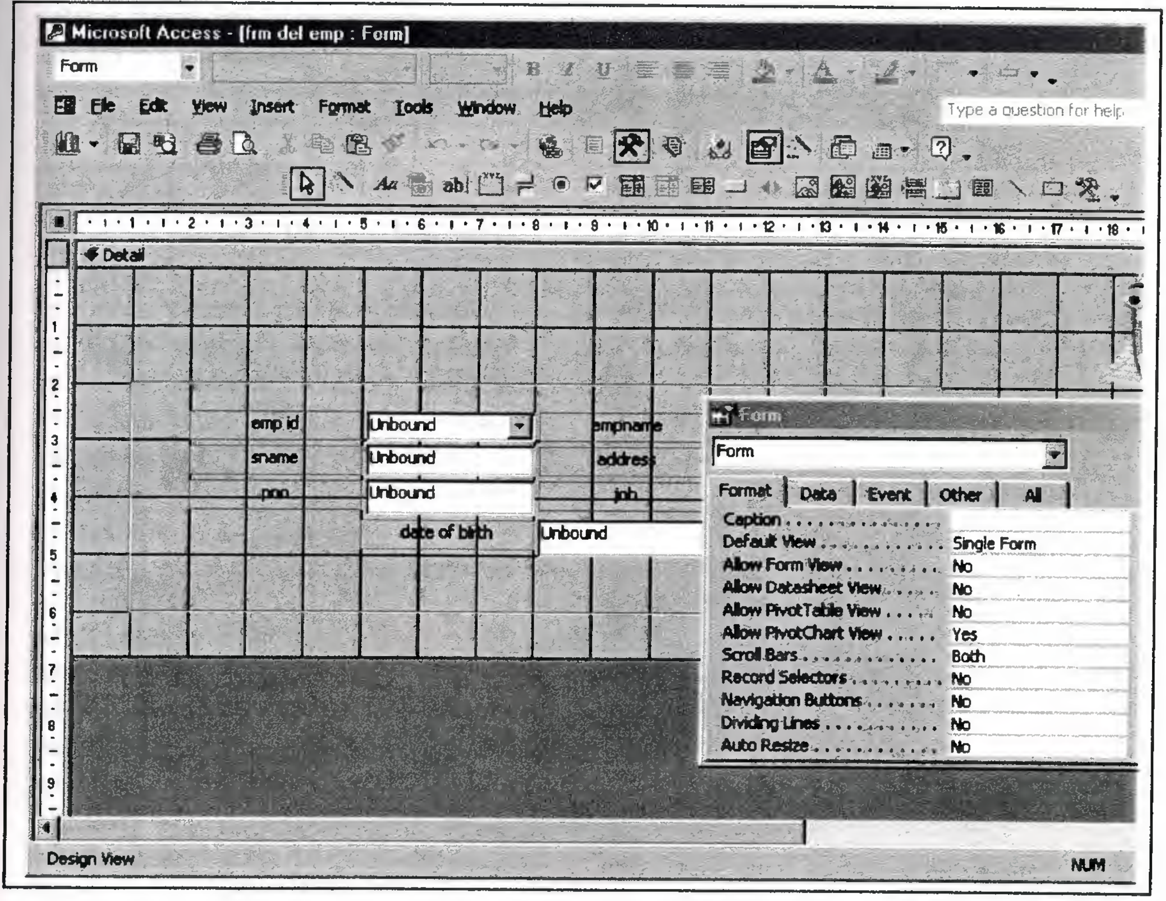


Figure 4.10. form “register new customer “

4.10.1 Design Register New Customer Form

In this form you can see the design register new customer


Microsoft Access - [frm del emp : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
Design View
NUM
Format Data Event Other All

emp id Unbound
sname Unbound
pnn Unbound
date of birth Unbound
empname
address
job

Form

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.10.1 design "register new customer"

4.10.2 Code Register New Customer

```
Option Compare Database

Private Sub Command18_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select * from tblcust where custid='" & Me.txtcid & " '"
Set rs = db.OpenRecordset(s)
If rs.EOF And rs.BOF Then
rs.AddNew
rs.Fields("custid").Value = Me.txtcid
rs.Fields("ctype").Value = Me.Cmbccty
rs.Fields("name").Value = Me.txtcn
rs.Fields("sname").Value = Me.txtcs
rs.Fields("address").Value = Me.txtca
rs.Fields("telno").Value = Me.txtcte
rs.Fields("job").Value = Me.txtcj
rs.Fields("date of birth") = Me.txtcd
rs.Update
MsgBox "new customer has been added", vbInformation
Me.txtcid = ""
Me.Cmbccty = ""
Me.txtcn = ""
Me.txtcs = ""
Me.txtca = ""
Me.txtcte = ""
Me.txtcj = ""
Me.txtcd = ""
Else
MsgBox "this record is already present "
End If
End Sub

Private Sub Command40_Click()
DoCmd.OpenForm "frmsubcust "
DoCmd.Close acForm, Me.Name
End Sub
```

4.11. Delete Customer Information Form

In this form you can see the delete customer information

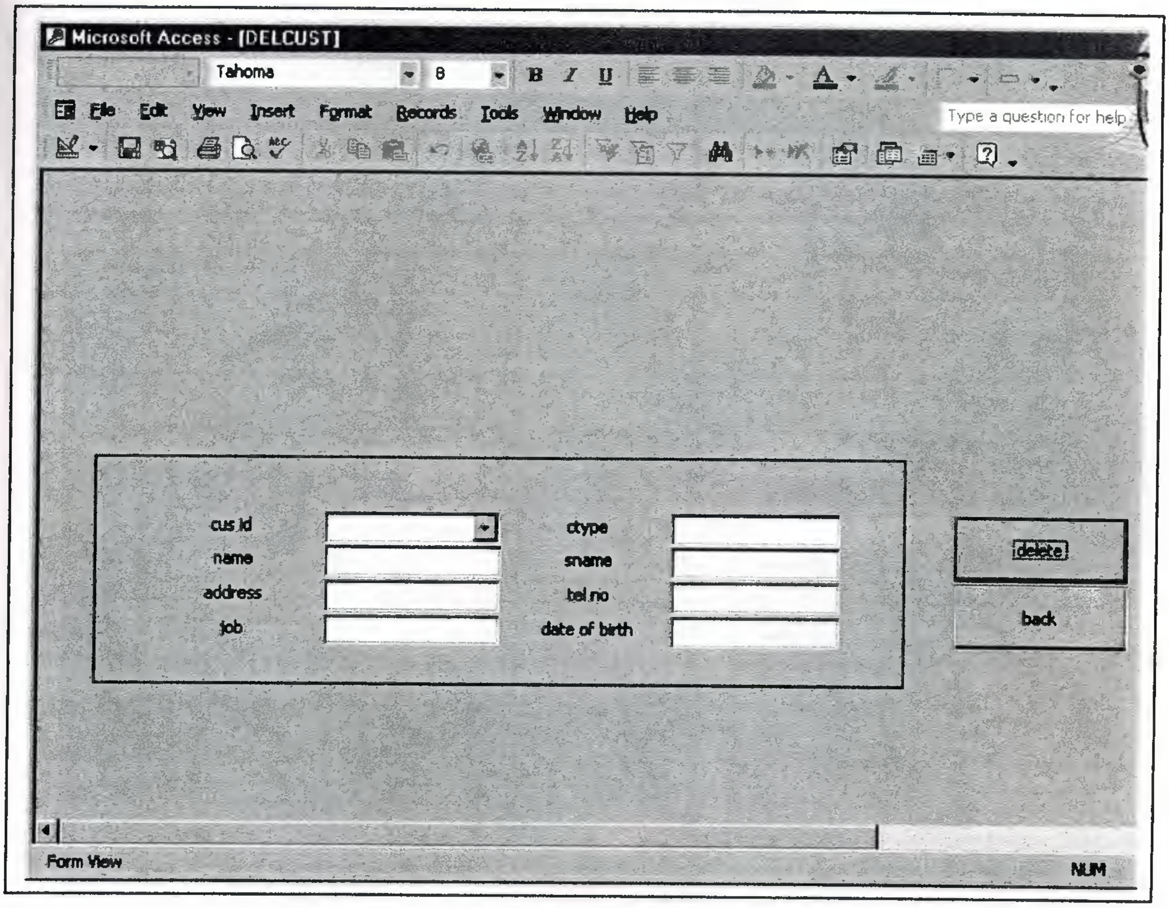
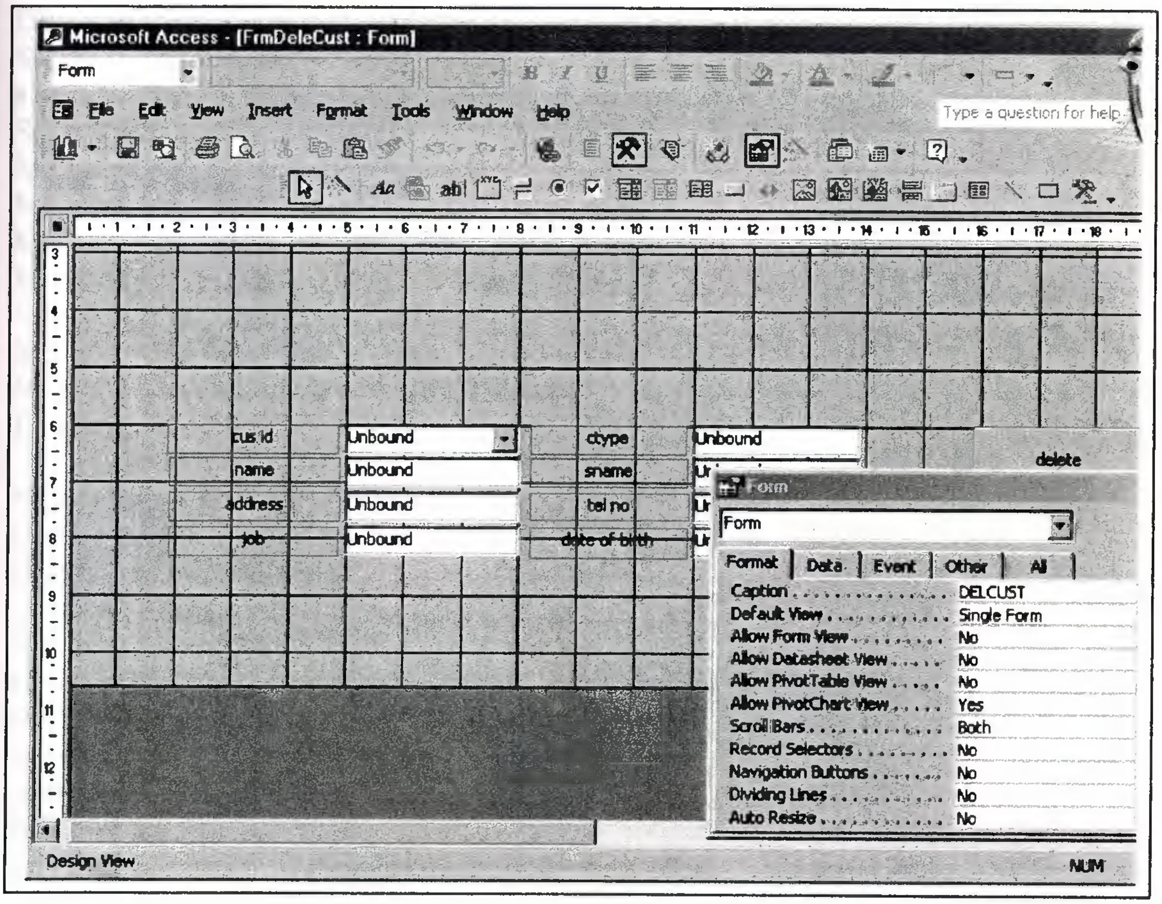


Figure 4.11 form “delete customer information”

4.11.1 Design Delete Customer Information Form

In this form you can see the design delete customer information


Microsoft Access - [FrmDeleCust : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Design View
NUM

cus id Unbound
name Unbound
address Unbound
job Unbound
ctype Unbound
sname Un
tel no Un
date of birth Un
delete

Form
Format | Data | Event | Other | All

Caption	DELCUST
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.11.1 design “delete customer information”

4.11.2 Code Delete Customer Information

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblcust where custid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcty = rs.Fields("ctype").Value
Me.txtcn = rs.Fields("name").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcte = rs.Fields("telno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth").Value
End Sub

Private Sub Command18_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblcust where custid='" & Me.cmbcid & "' "
Set rs = db.OpenRecordset(s)
rs.MoveLast
rs.Delete
rs.MoveFirst
End Sub

Private Sub Command21_Click()
DoCmd.OpenForm "frmsubcust "
DoCmd.Close acForm, Me.Name
End Sub
```

4.12. Edit Customer Information Form

In this form you can see the edit customer information

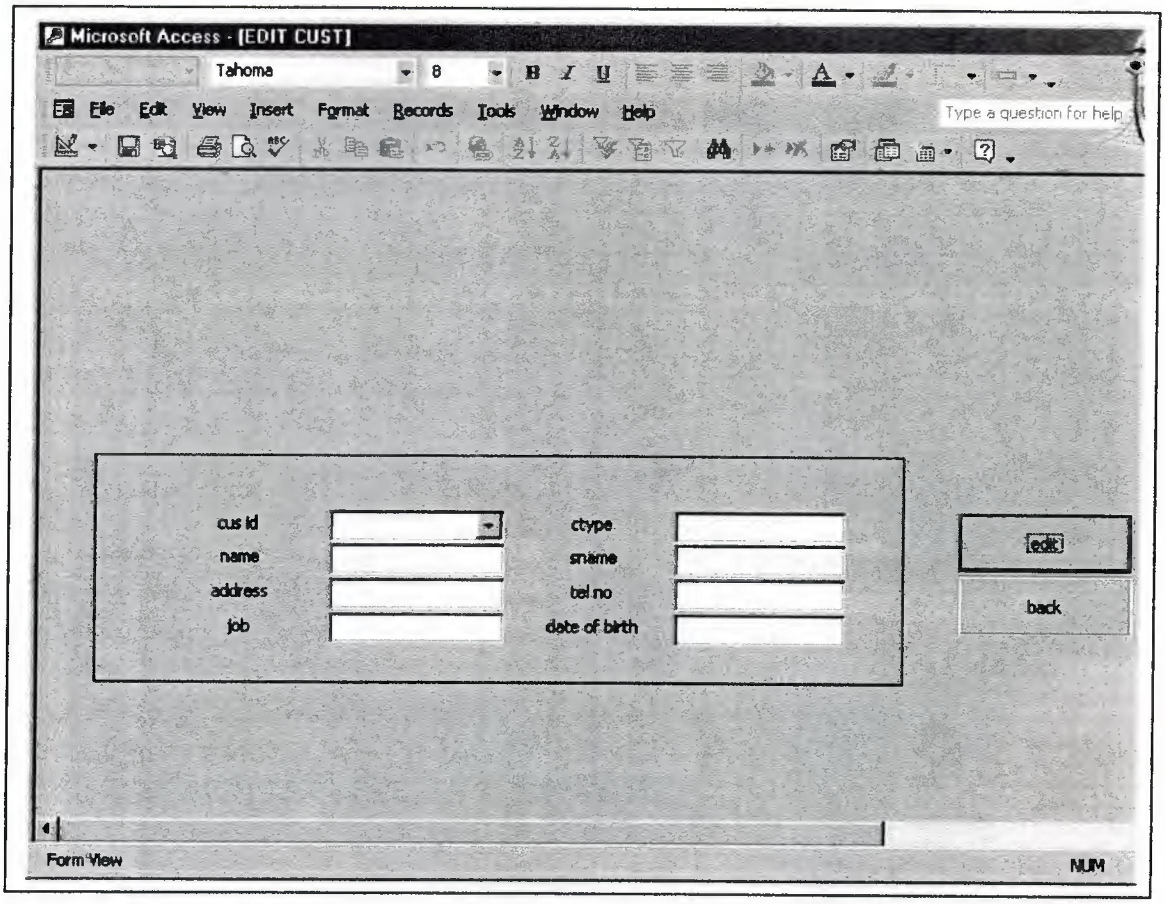
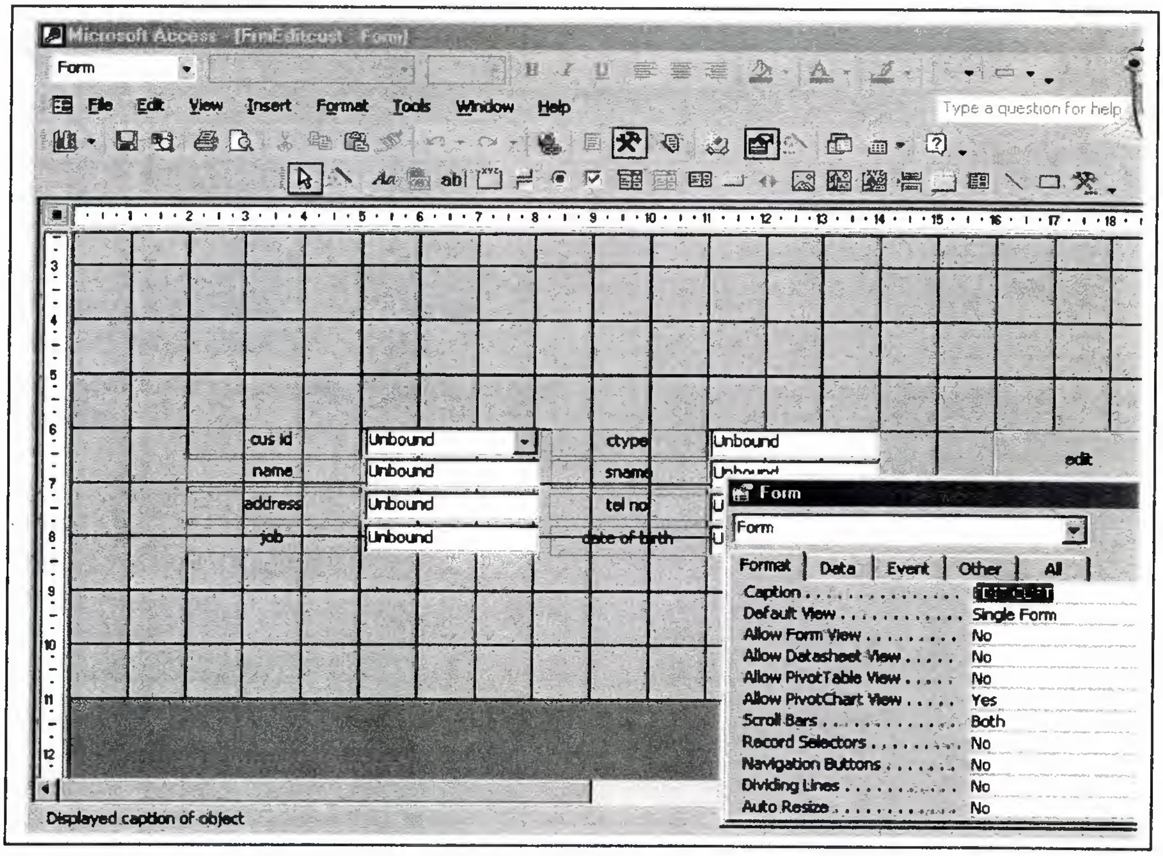


Figure 4.12 form "edit customer information"

4.12.1 Design Edit Customer Information Form

In this form you can see the design delete customer information



cus id	Unbound	ctype	Unbound
name	Unbound	sname	Unbound
address	Unbound	tel no	U
job	Unbound	date of birth	U

Form

Caption	EDIT CUST
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Displayed caption of object

Figure 4.12.1 design "edit customer information"

4.12.2 Code Edit Customer Information

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim a As String
Set db = CurrentDb()
s = "select*from tblcust where custid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcty = rs.Fields("ctype").Value
Me.txtcn = rs.Fields("name").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcte = rs.Fields("telno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth").Value
End Sub

Private Sub Command16_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
s = "tblcust"
Set db = CurrentDb()
Set rs = db.OpenRecordset(s)
rs.edit
rs.Fields("ctype").Value = Me.txtcty
rs.Fields("name").Value = Me.txtcn
rs.Fields("sname").Value = Me.txtcs
rs.Fields("address").Value = Me.txtca
rs.Fields("telno").Value = Me.txtcte
rs.Fields("job").Value = Me.txtcj
rs.Fields("date of birth").Value = Me.txtcd
rs.Update
End Sub

Private Sub Command20_Click()
DoCmd.OpenForm "frmsubcust "
DoCmd.Close acForm, Me.Name
End Sub
```

4.13. Search Customer Information Form

In this form you can see the search customer information

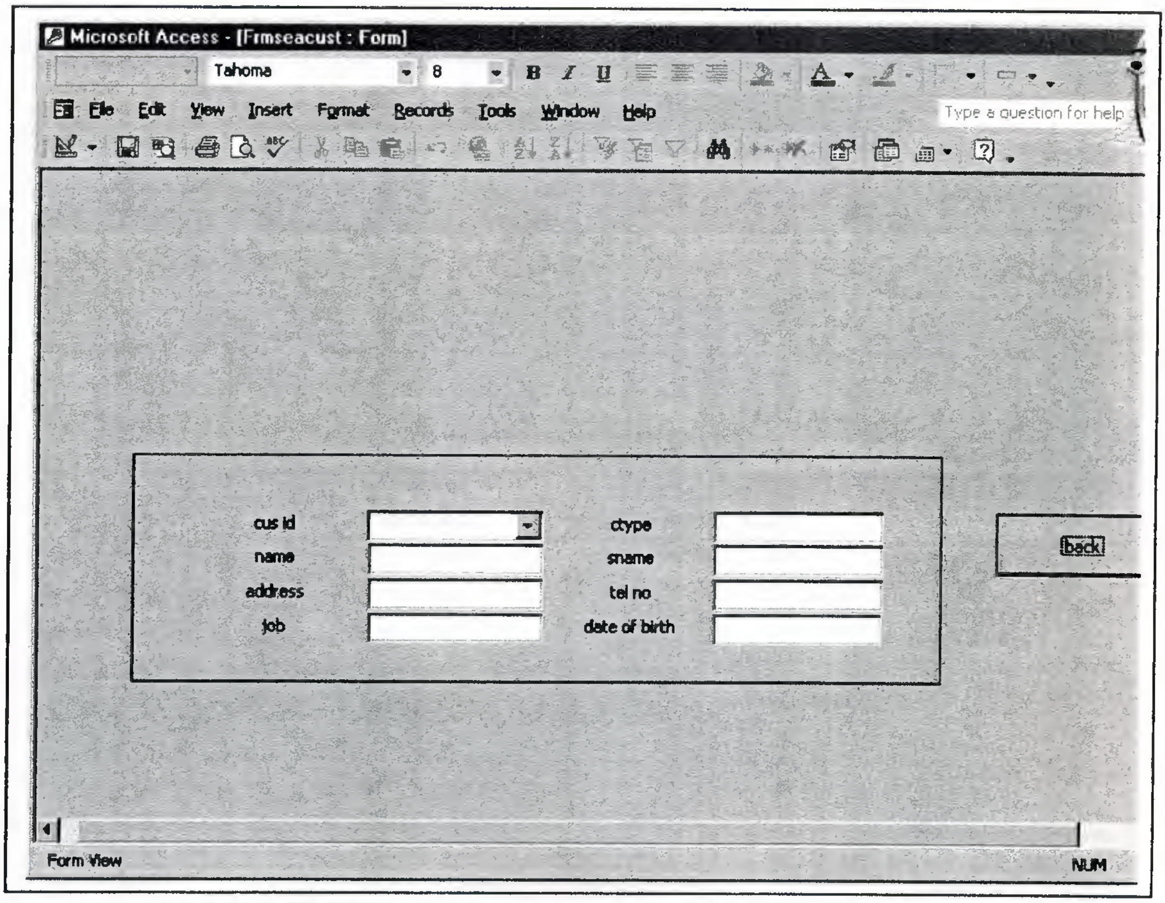
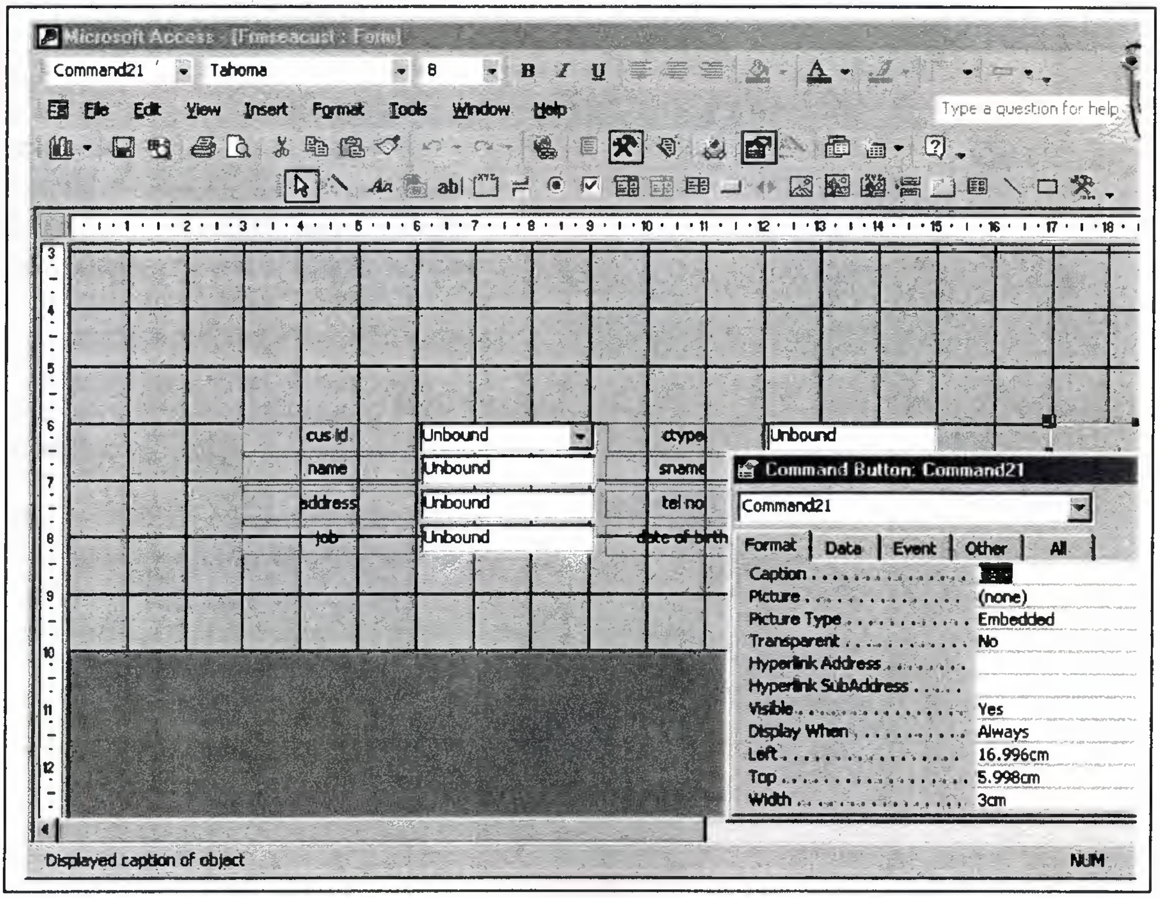


Figure 4.13 form "search customer information"

4.13.1 Design Search Customer Information Form

In this form you can see the design search customer information


Microsoft Access - [Frmseacust : Form]
Command21
Tahoma
8
File Edit View Insert Format Tools Window Help
Type a question for help
Command Button: Command21
Format Data Event Other All
NUM

cus id: Unbound
name: Unbound
address: Unbound
job: Unbound
ctype: Unbound
sname
tel no
date of birth

Command21

Caption	[illegible]
Picture	(none)
Picture Type	Embedded
Transparent	No
Hyperlink Address	
Hyperlink SubAddress	
Visible	Yes
Display When	Always
Left	16.996cm
Top	5.998cm
Width	3cm

Displayed caption of object

Figure 4.13.1 design “search customer information”

4.13.2 Code Search Customer Information

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim a As String
Set db = CurrentDb()
s = "select*from tblcust where custid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcty = rs.Fields("ctype").Value
Me.txtcn = rs.Fields("name").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcte = rs.Fields("telno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth").Value
End Sub

Private Sub Command20_Click()
DoCmd.OpenForm "frmSubcust"
DoCmd.Close acForm, Me.Name
End Sub
```

4.14. Call Tracking Menu Form

In this form you can see the call tracking menu

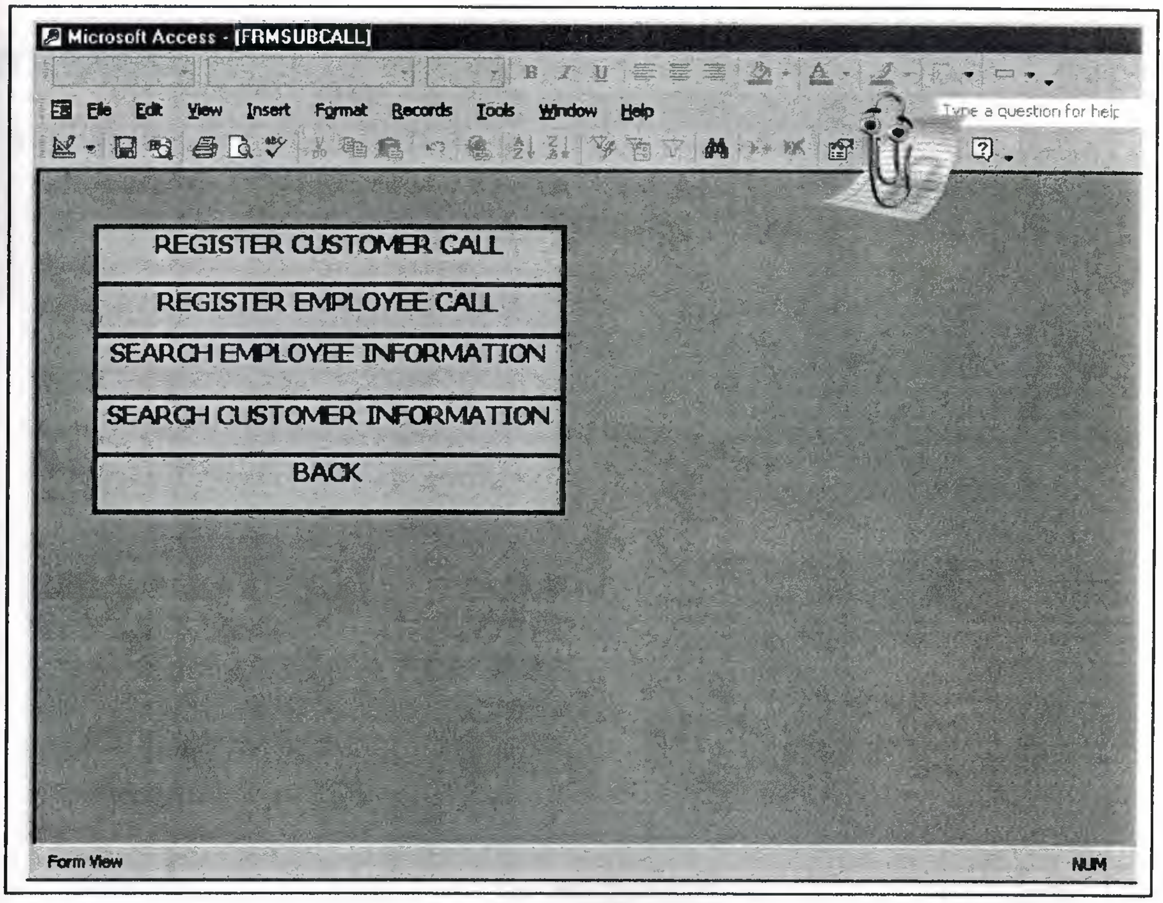
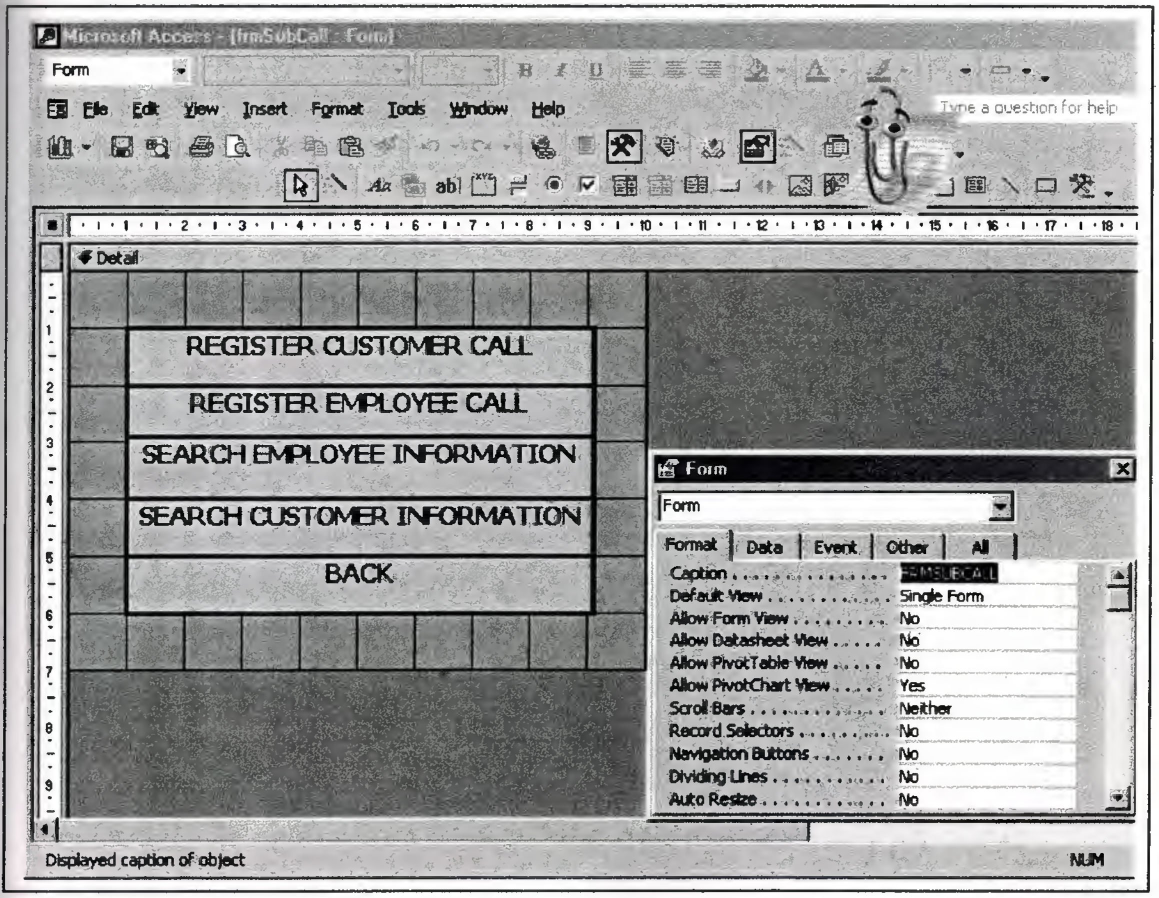


Figure 4.14. form "call tracking menu"

4.14.1. Design Call Tracking Menu Form

In this form you can see the design call tracking menu


Microsoft Access - [frmSubCall : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
REGISTER CUSTOMER CALL
REGISTER EMPLOYEE CALL
SEARCH EMPLOYEE INFORMATION
SEARCH CUSTOMER INFORMATION
BACK
Form
Form
Format Data Event Other All
Displayed caption of object
NUM

Caption	FRMSUBCALL
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Neither
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.14.1 design "call tracking menu"

4.14.2. Code Call Tracking Menu

```
Option Compare Database


Private Sub Label78_Click()
DoCmd.OpenForm "frmRegCustCall"
End Sub

Private Sub Label80_Click ()
DoCmd.OpenForm "frmRegempCall"
End Sub

Private Sub Label81_Click ()
DoCmd.OpenForm "frmseaempinf"
End Sub

Private Sub Label82_Click ()
DoCmd.OpenForm "frmseacustinf"
End Sub

Private Sub Label83_Click ()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, Me.Name
End Sub
```

4.15. Register Customer Call Form

In this form you can see the register customer call

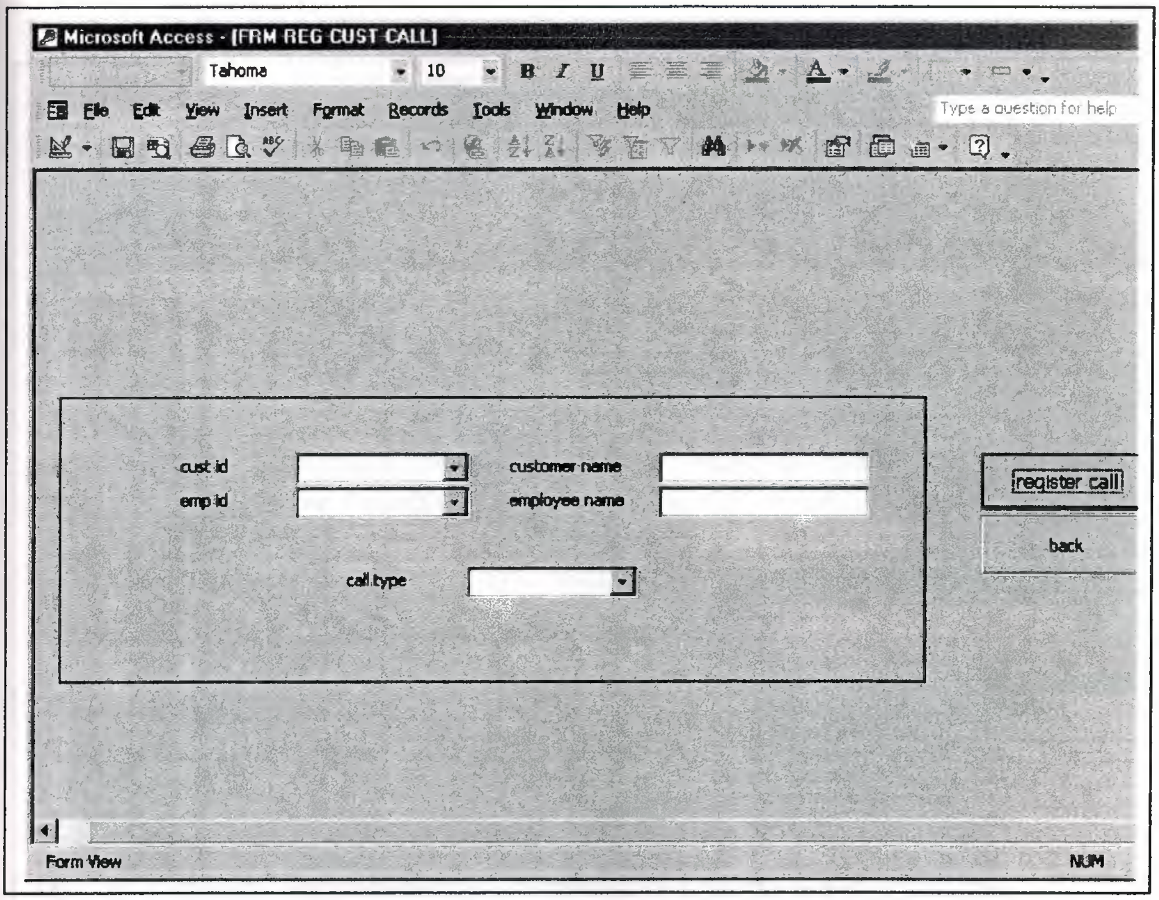
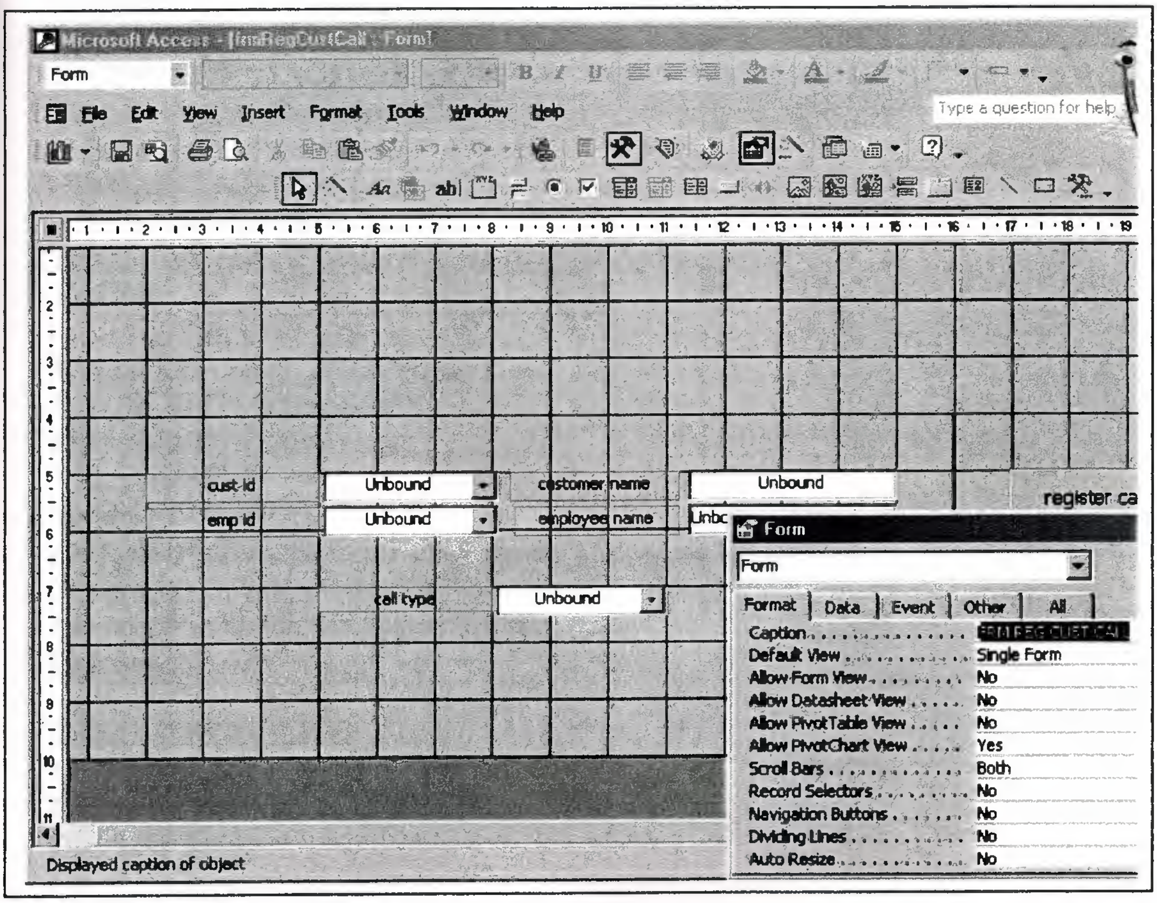


Figure 4.15 form "register customer call"

4.15.1 Design Register Customer Call

In this form you can see the design register customer call


Microsoft Access - [frmRegCustCall : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
register ca
Displayed caption of object

cust id: Unbound

emp id: Unbound

customer name: Unbound

employee name: Unbc

call type: Unbound

Form

Form

Format | Data | Event | Other | All

Caption	FRMREGCUSTCALL
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.15.1 design "register customer call"

4.15.2 Code Register Customer Call

```
Option Compare Database

Private Sub cmbcid_AfterUpdate ()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "SELECT * FROM tblCust WHERE custid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
 Me.txtcn = rs.Fields("Name").Value
 End Sub

Private Sub cmbcid_BeforeUpdate(Cancel As Integer)
End Sub

Private Sub cmbid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblemp where empid='" & Me.cmbid & "'"
Set rs = db.OpenRecordset(s)
Me.txtn = rs.Fields("empname").Value
End Sub

Private Sub Command0_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
Set rs = db.OpenRecordset("tblCustEmp")
rs.AddNew
rs.Fields("custid").Value = Me.cmbcid
rs.Fields("empid").Value = Me.cmbid
rs.Fields("cdate").Value = Date
rs.Fields("ctime").Value = Time
rs.Update
End Sub

Private Sub Command46_Click()
DoCmd.OpenForm "frmsubcall"
DoCmd.Close acForm, Me.Name
End Sub
```

4.16. Register Employee Call Form

In this form you can see the register customer call

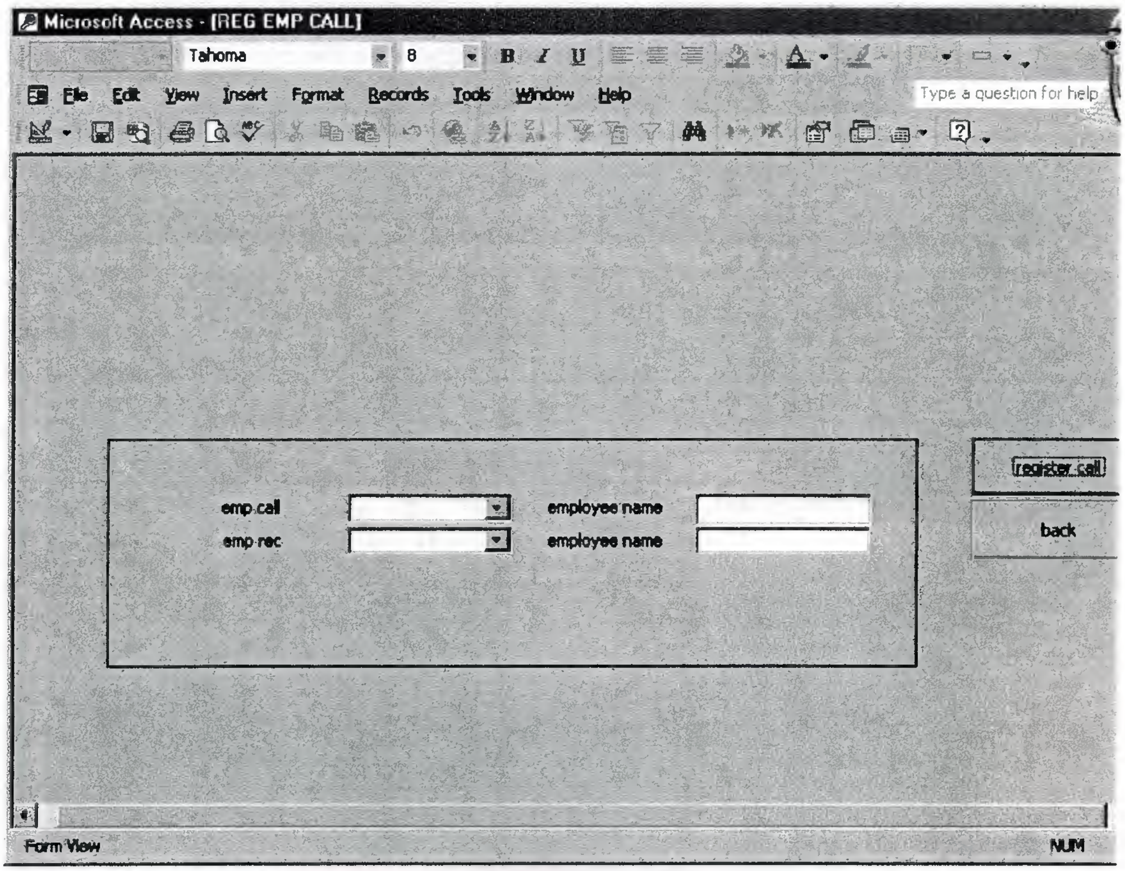
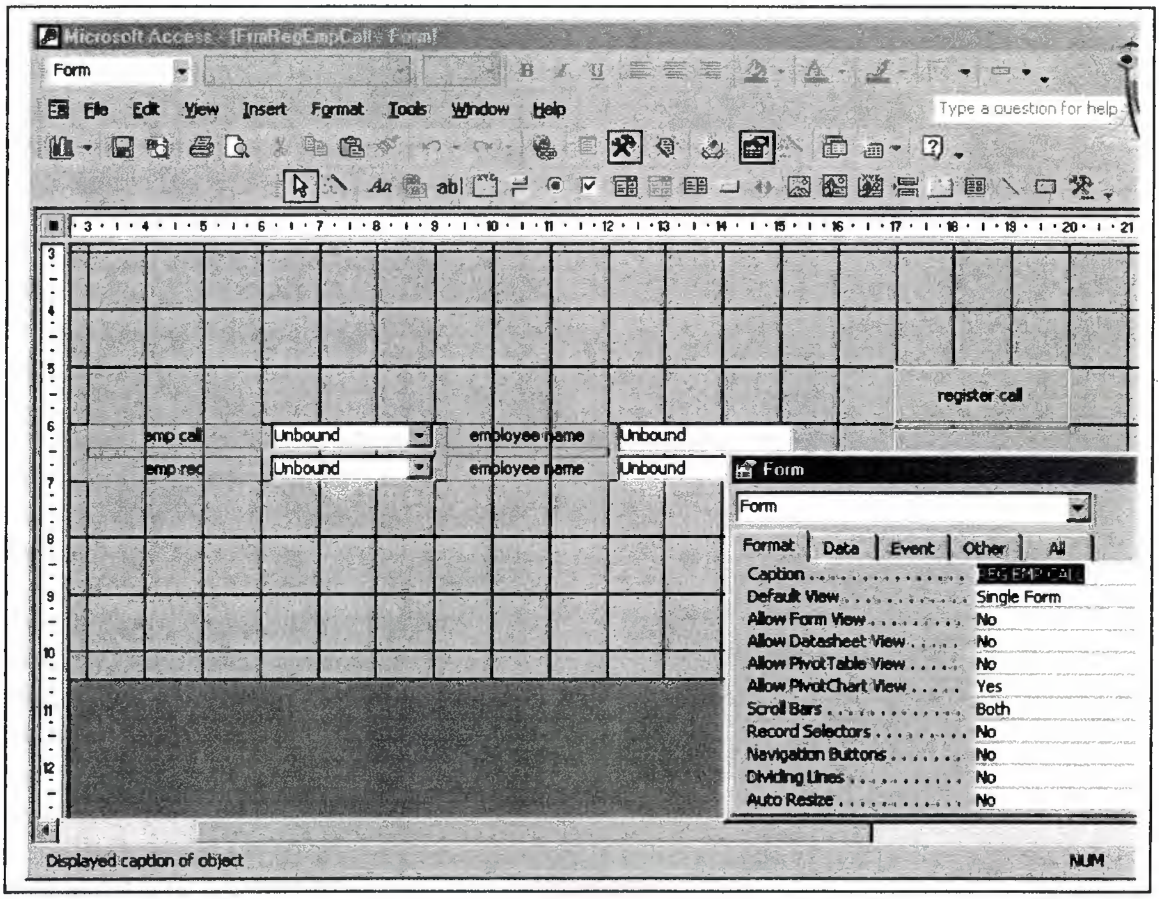


Figure 4.16 form "register employee call"

4.16.1. Design Register Employee Call

In this form you can see the design register employee call



Caption	REG EMP CALL
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.16.1 design "register employee call"

4.16.2 Code Register Employee Call

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblemp where empid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcn = rs.Fields("empname").Value
End Sub

Private Sub cmbid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblemp where empid='" & Me.cmbid & "'"
Set rs = db.OpenRecordset(s)
Me.txtn = rs.Fields("empname").Value
End Sub

Private Sub Command24_Click()
DoCmd.OpenForm "frmsubcall"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Command8_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
Set rs = db.OpenRecordset("tblemptemp")
rs.AddNew
rs.Fields("empidcall").Value = Me.cmbcid
rs.Fields("empidrec").Value = Me.cmbid
rs.Fields("date").Value = Date
rs.Fields("time").Value = Time
rs.Update
End Sub
```

4.17. Search Employee Information form

In this form you can see the search employee information

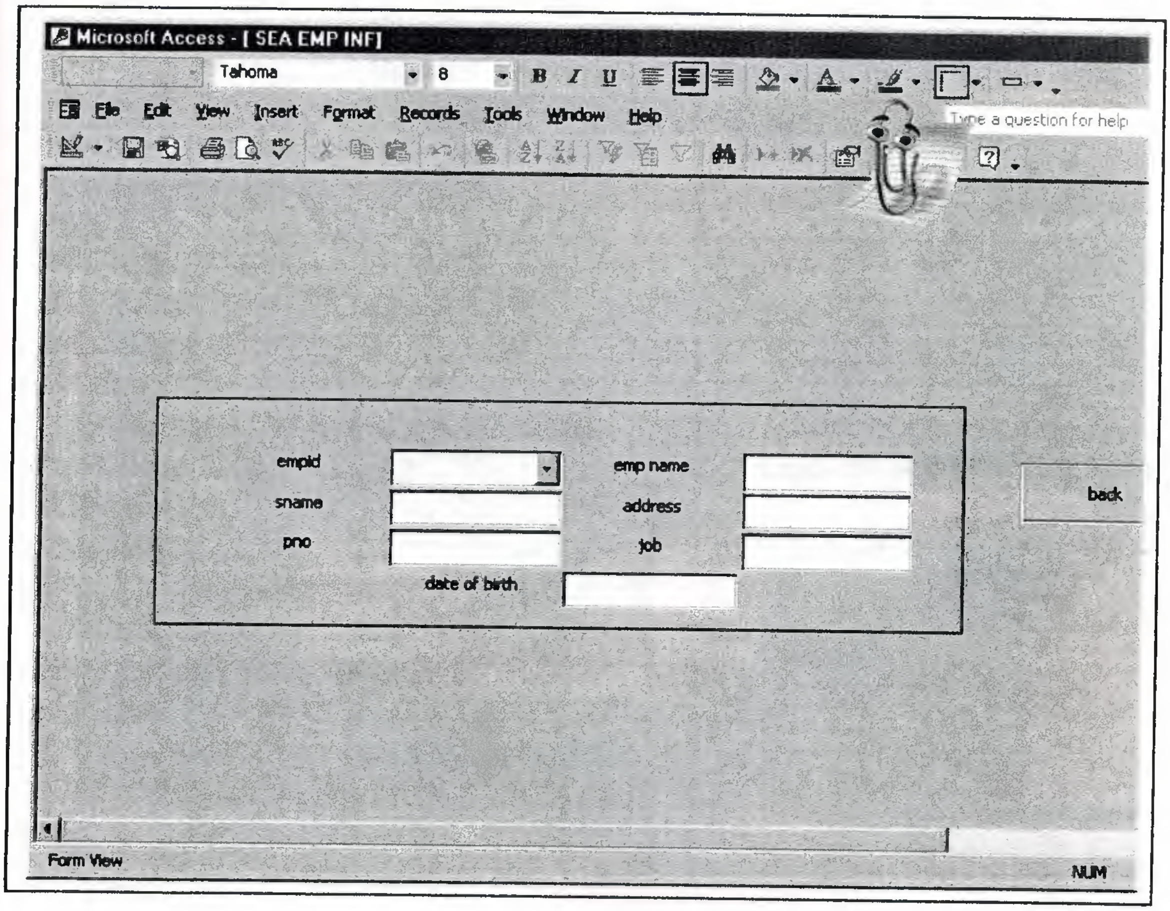
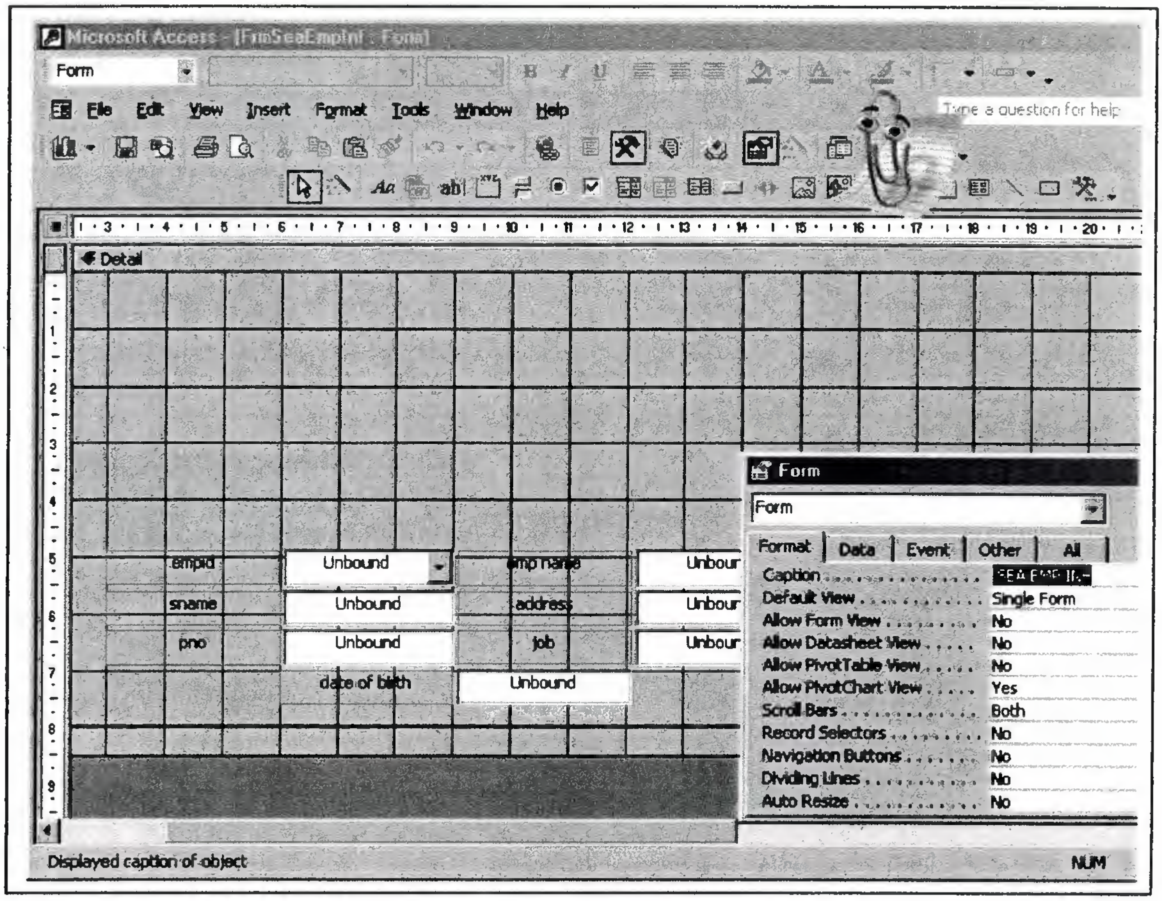


Figure 4.17 form "search employee information"

4.17.1 Design Search Employee Information

In this form you can see the search employee information



Caption	[illegible]
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Figure 4.17.1 design "search employee information"

4.17.2 Code Search Employee Information

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select* from tblemp where empid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcn = rs.Fields("empname").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcp = rs.Fields("pno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth")
End Sub

Private Sub Command30_Click()
DoCmd.OpenForm "frmSubcall"
DoCmd.Close acForm, Me.Name
End Sub
```

4.18. Search Customer Information Form

In this form you can see the search customer information

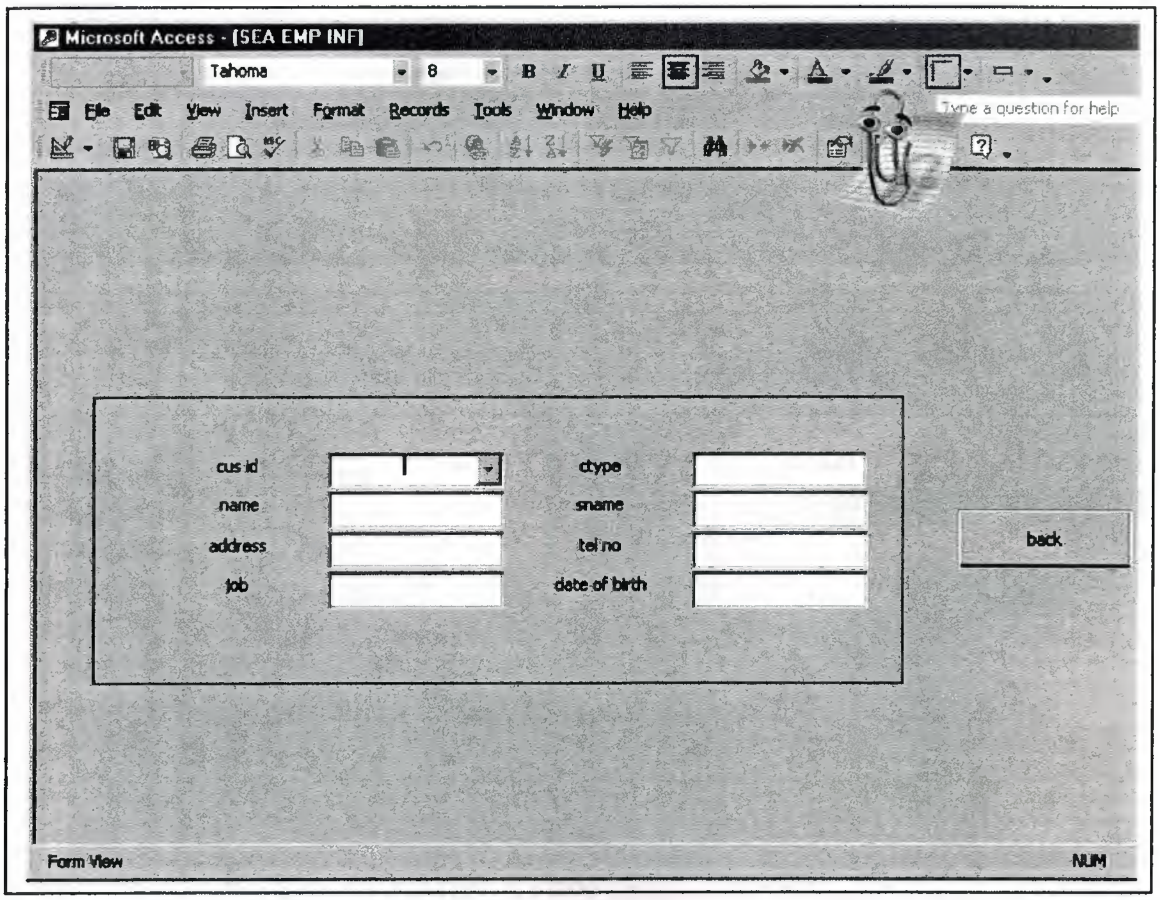
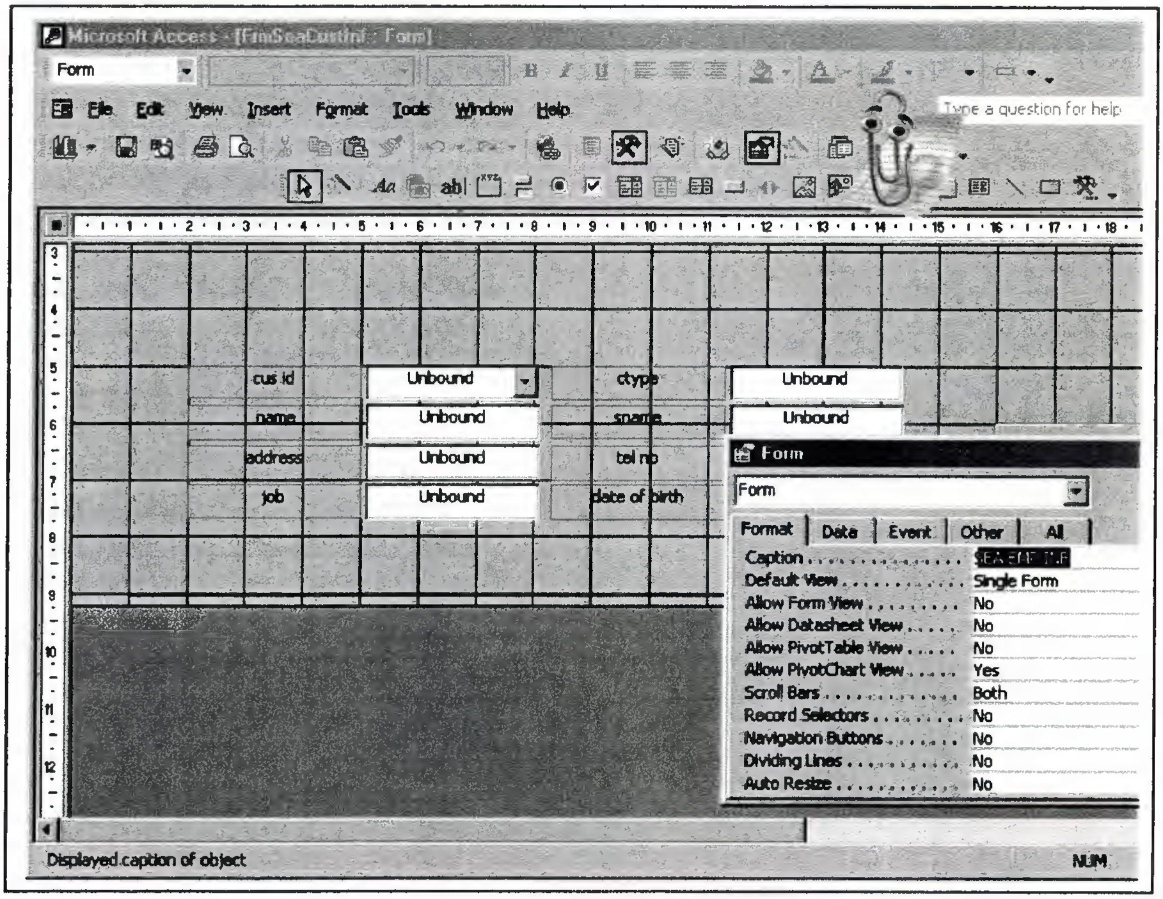


Figure 4.18 form "search customer information"

4.18.1 Design Search Customer Information

In this form you can see the design search customer information



Microsoft Access - [FrmSeaCustInf : Form]

cus id	Unbound	ctype	Unbound
name	Unbound	sname	Unbound
address	Unbound	tel no	
job	Unbound	date of birth	

Form

Form

Format	Data	Event	Other	All

Caption	SEA CUS INF
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Displayed caption of object

Figure 4.18.1 form “search customer information”

4.18.2 Code Search Customer Information

```
Option Compare Database

Private Sub cmbcid_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim a As String
Set db = CurrentDb()
s = "select*from tblcust where custid='" & Me.cmbcid & "'"
Set rs = db.OpenRecordset(s)
Me.txtcty = rs.Fields("ctype").Value
Me.txtcn = rs.Fields("name").Value
Me.txtcs = rs.Fields("sname").Value
Me.txtca = rs.Fields("address").Value
Me.txtcte = rs.Fields("telno").Value
Me.txtcj = rs.Fields("job").Value
Me.txtcd = rs.Fields("date of birth").Value
End Sub

Private Sub Command22_Click()
DoCmd.OpenForm "frmSubcall"
DoCmd.Close acForm, Me.Name
End Sub
```

4.19. Search form

In this form you can see the search form

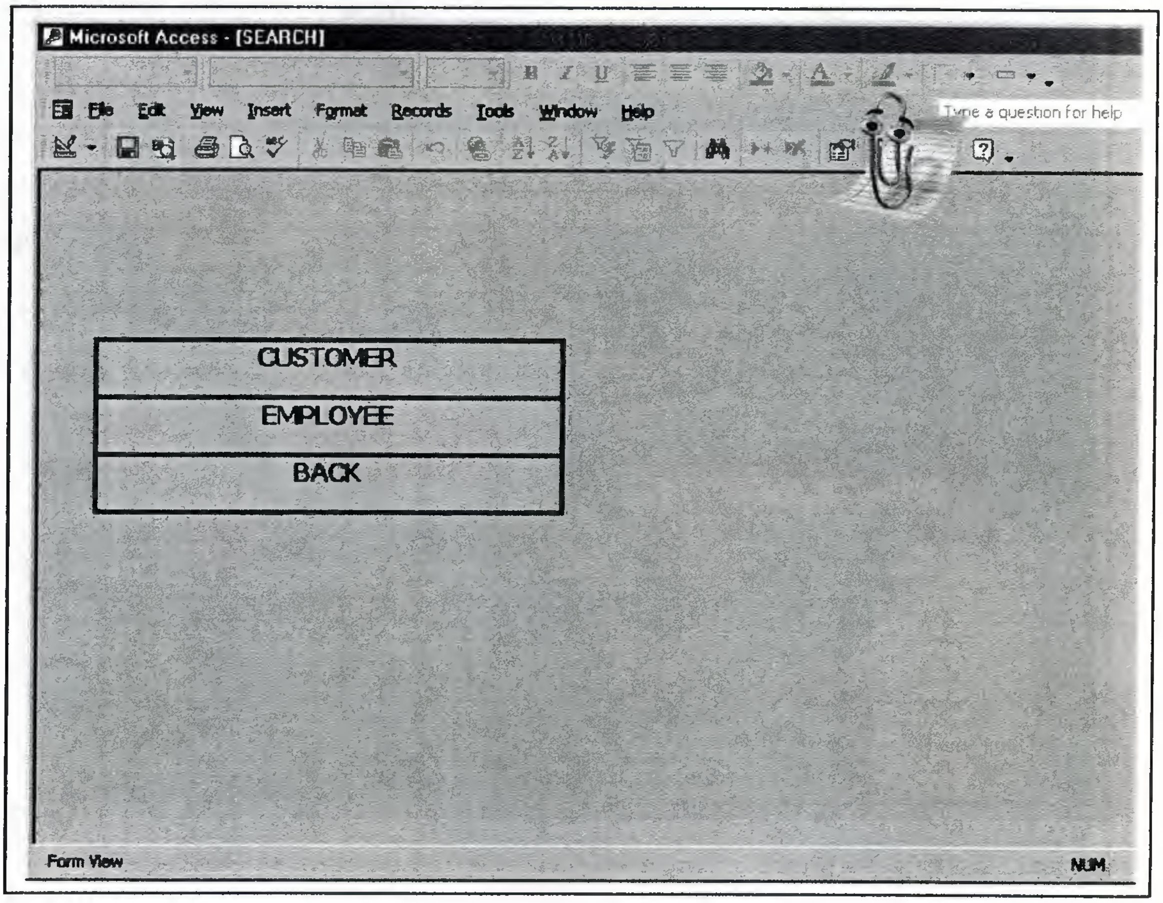
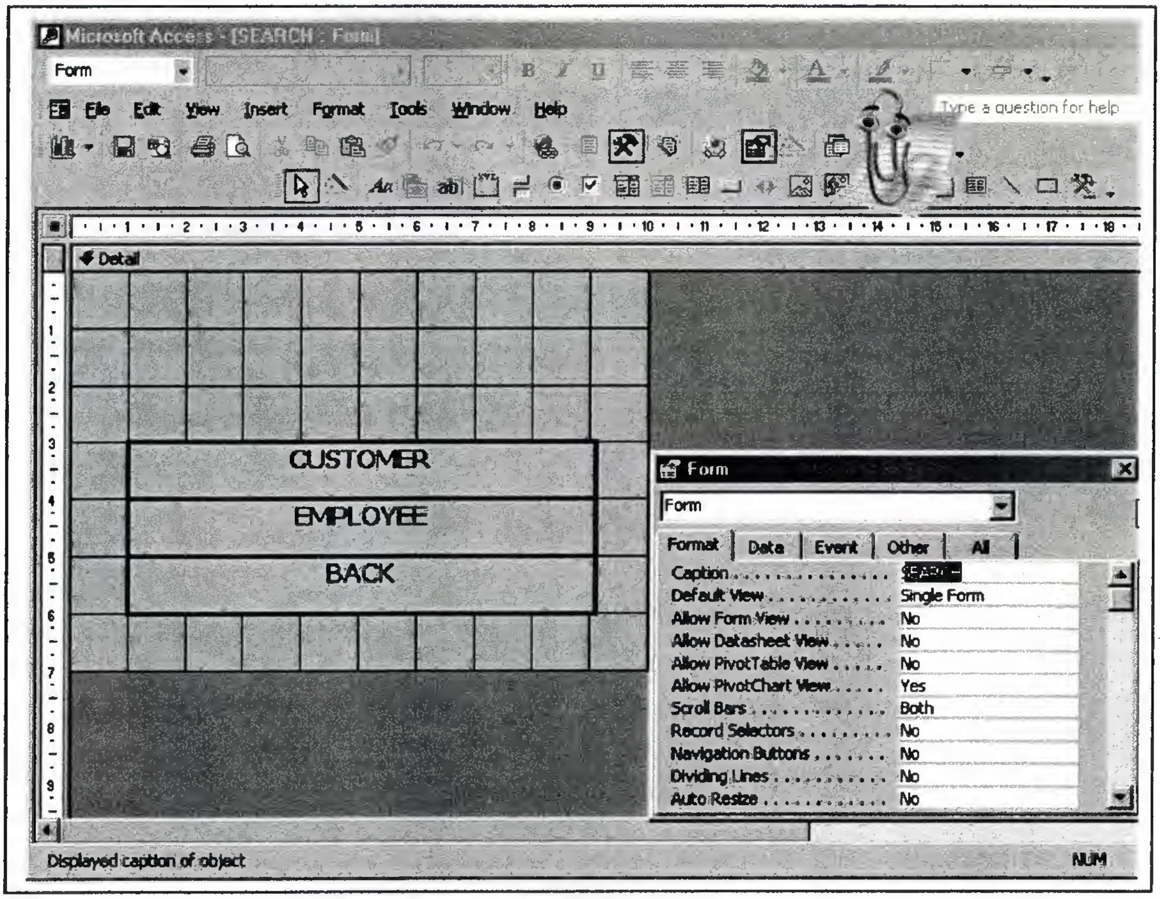

Microsoft Access - [SEARCH]
File Edit View Insert Format Records Tools Window Help
Type a question for help
CUSTOMER
EMPLOYEE
BACK
Form View
NUM

Figure 4.19 form “search”

4.19.1 Design Search

In this form you can see the design search


Microsoft Access - [SEARCH : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
CUSTOMER
EMPLOYEE
BACK
Form
Format Data Event Other All

Form

Caption . . . . SEARCH
Default View . . . . Single Form
Allow Form View . . . . No
Allow Datasheet View . . . . . No
Allow PivotTable View . . . . . No
Allow PivotChart View . . . . . Yes
Scroll Bars . . . . Both
Record Selectors . . . . No
Navigation Buttons . . . . No
Dividing Lines . . . . No
Auto Resize . . . . No

Displayed caption of object

NUM

Figure 4.19.1 design “search”

4.19.2 Code Search

```
Option Compare Database

Private Sub Label78_Click ()
DoCmd.OpenForm "frmcustsearch"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Label80_Click ()
DoCmd.OpenForm "frmempsearch"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Label83_Click ()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, Me.Name
End Sub
```

4.20. Search Customer Form

In this form you can see the search customer form

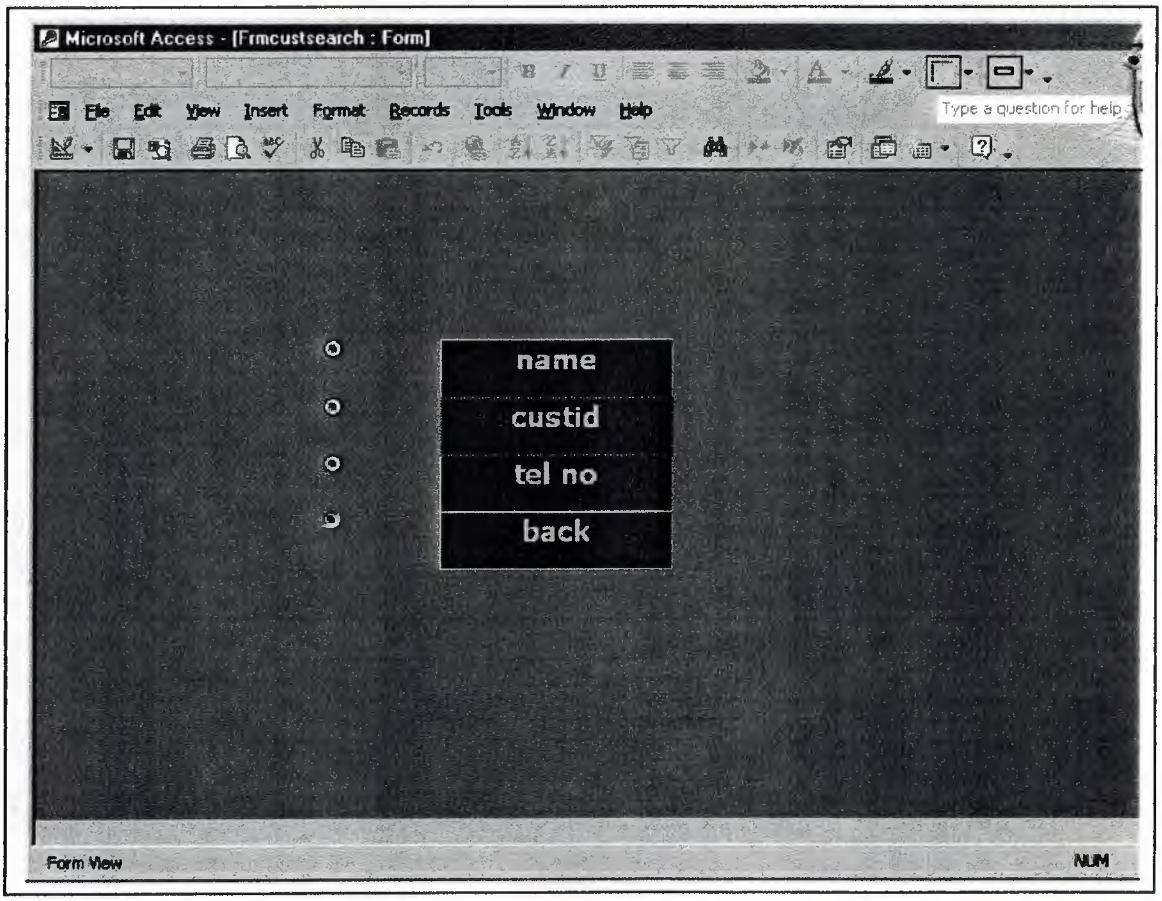
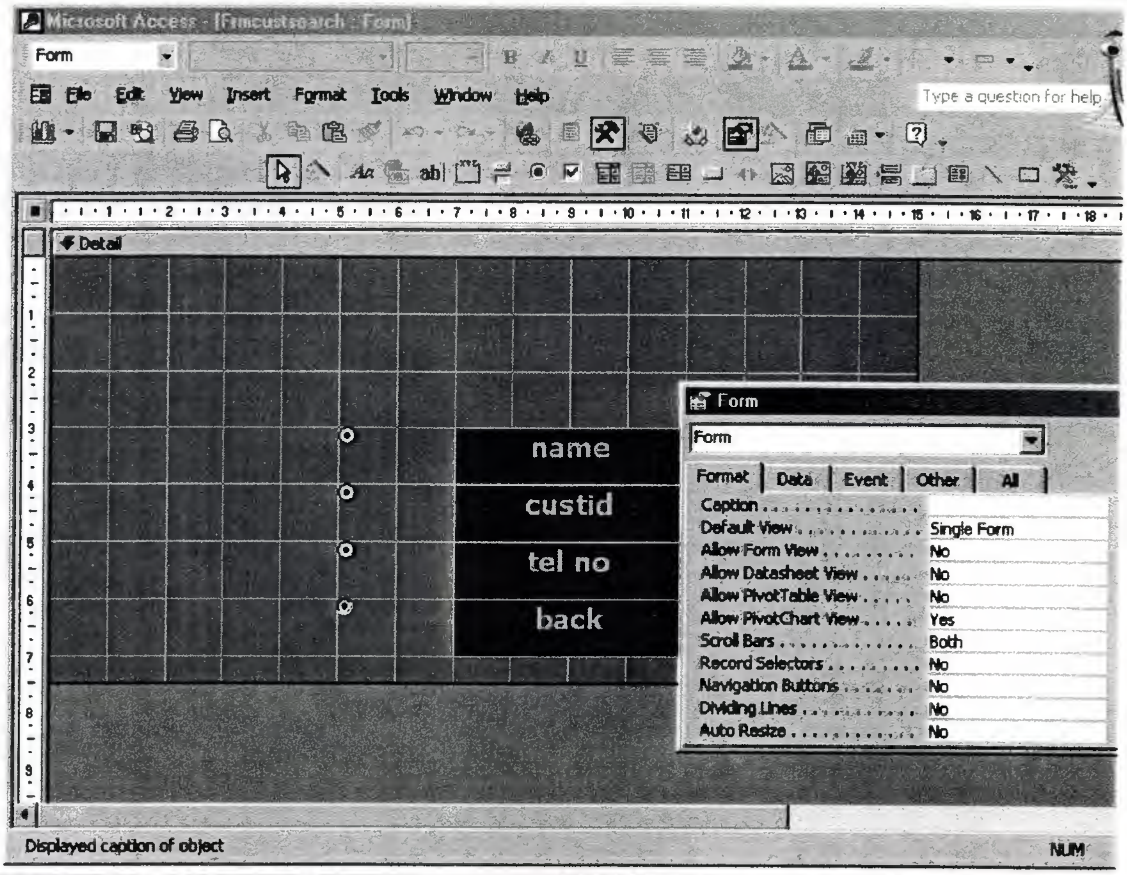


Figure 4.20 form "search customer"

4.20.1 Design Search Customer

In this form you can see the design search customer



Form

Form

Format | Data | Event | Other | All

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Displayed caption of object

Figure 4.20.1 design “search customer”

4.20.2 Code Search Customer

```
Option Compare Database

Private Sub Option0_Click()
DoCmd.OpenForm "frmseacust2"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Option20_Click ()
DoCmd.OpenForm "search"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Option4_Click ()
DoCmd.OpenForm "frmseacust1"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Option6_Click ()
DoCmd.OpenForm "frmseacust3"
DoCmd.Close acForm, Me.Name
End Sub
```

4.20.3 Search Name Customer Form

In this form you can see the search name customer form

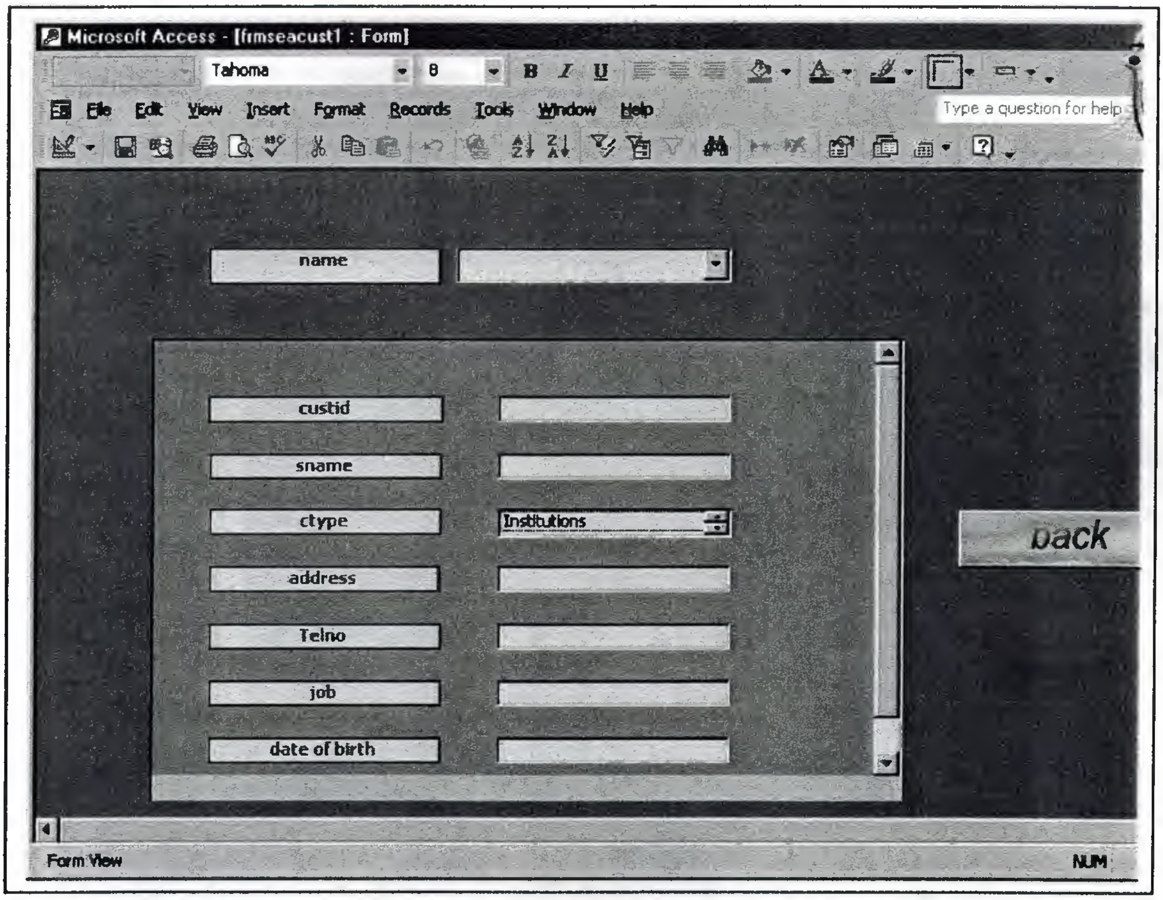
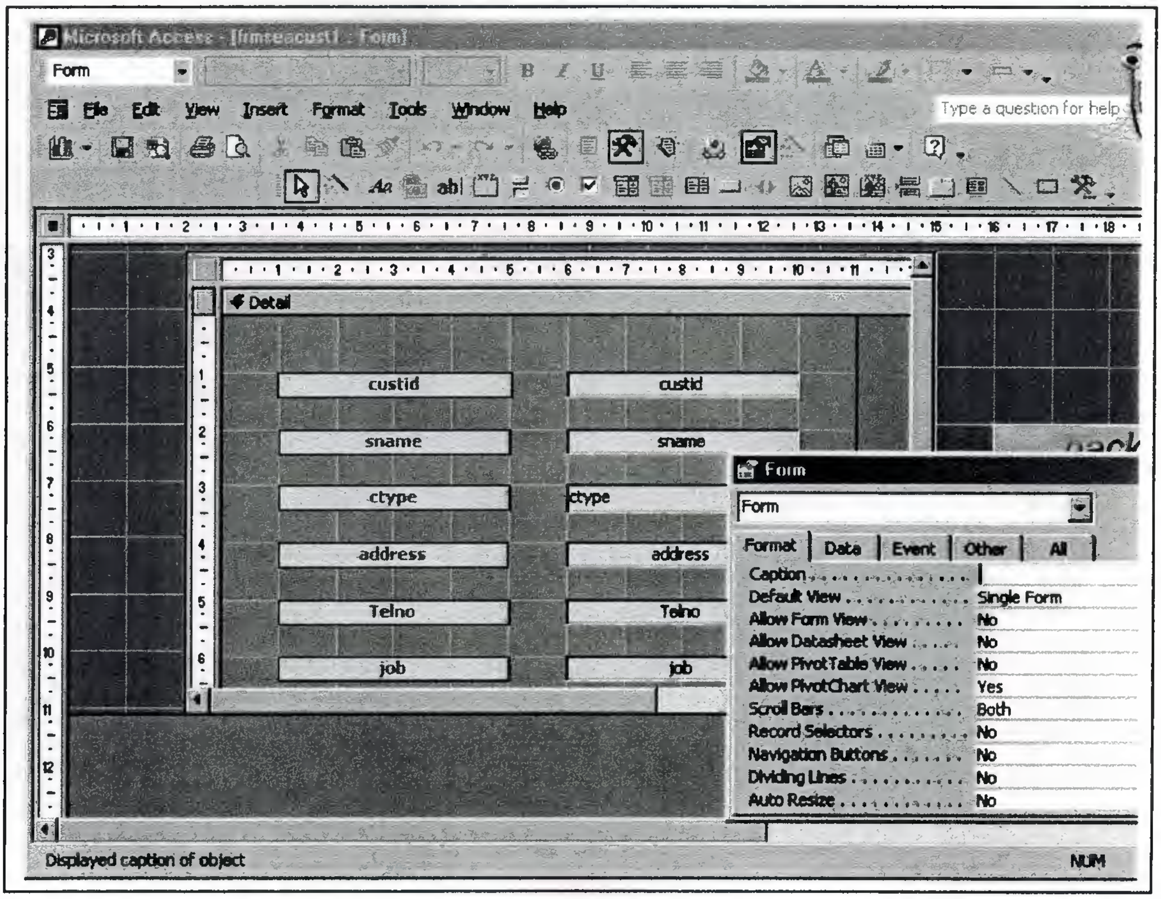


Figure 4.20.3 form “search name”

4.20.3.1 Design Search Name Customer

In this form you can see the design name customer



Microsoft Access - [frmseacust1 : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
custid
sname
ctype
address
Telno
job
Form
Format Data Event Other All
Displayed caption of object
NUM

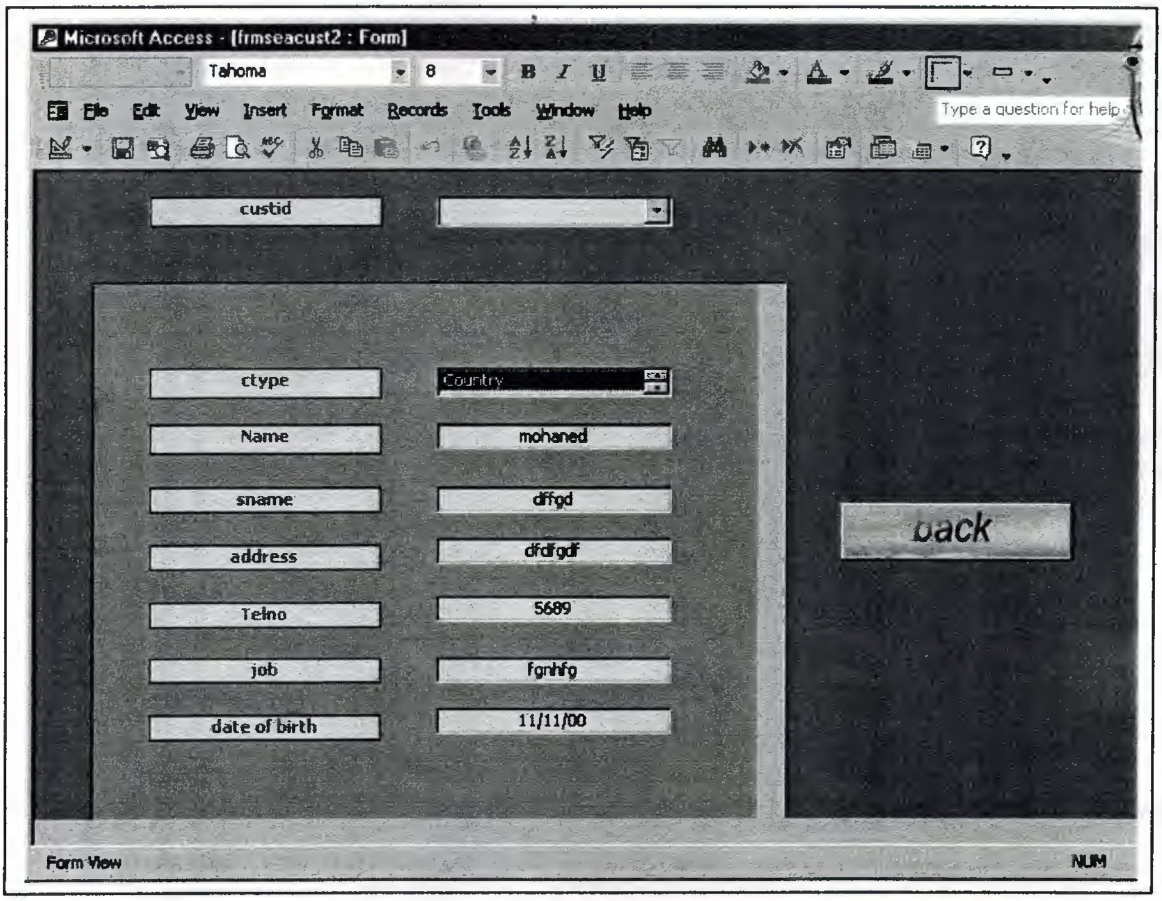
Form

Caption . . . .
Default View . . . . Single Form
Allow Form View . . . . No
Allow Datasheet View . . . . . . No
Allow PivotTable View . . . . . No
Allow PivotChart View . . . . . Yes
Scroll Bars . . . . Both
Record Selectors . . . . No
Navigation Buttons . . . . No
Dividing Lines . . . . No
Auto Resize . . . . No

Figure 4.20.3.1 design "name customer"

4.20.4 Search Custid Customer Form

In this form you can see the search custid customer form



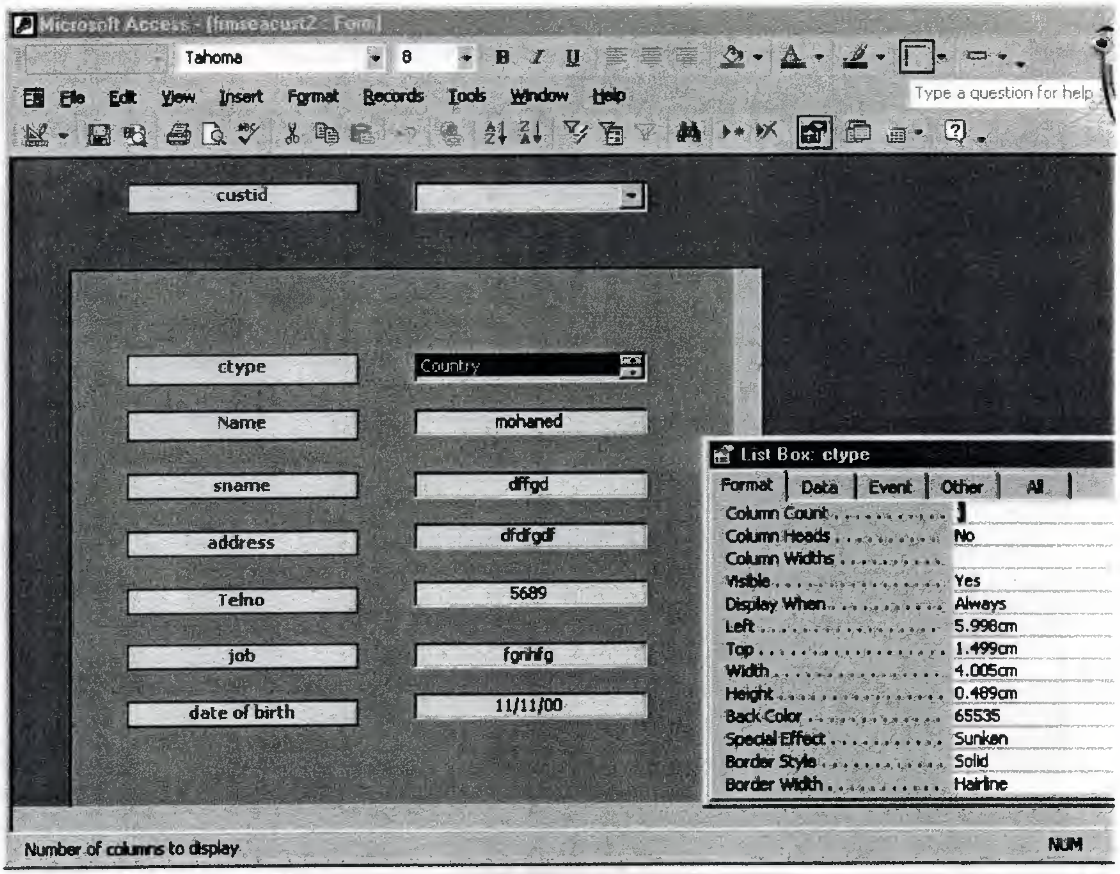
Microsoft Access - [frmseacust2 : Form]

custid	
ctype	Country
Name	mohaned
sname	dffgd
address	dfdfgdf
Telno	5689
job	fgnhfg
date of birth	11/11/00

Figure 4.20.4 form "search custid"

4.20.4.1 Design Search Custid Customer

In this form you can see the design search custid



custid	
ctype	Country
Name	mohaned
sname	dffgd
address	dfdfgdf
Telno	5689
job	fgnhfg
date of birth	11/11/00

List Box: ctype

Format | Data | Event | Other | All

Column Count	1
Column Heads	No
Column Widths	
Visible	Yes
Display When	Always
Left	5.998cm
Top	1.499cm
Width	4.005cm
Height	0.489cm
Back Color	65535
Special Effect	Sunken
Border Style	Solid
Border Width	Hairline

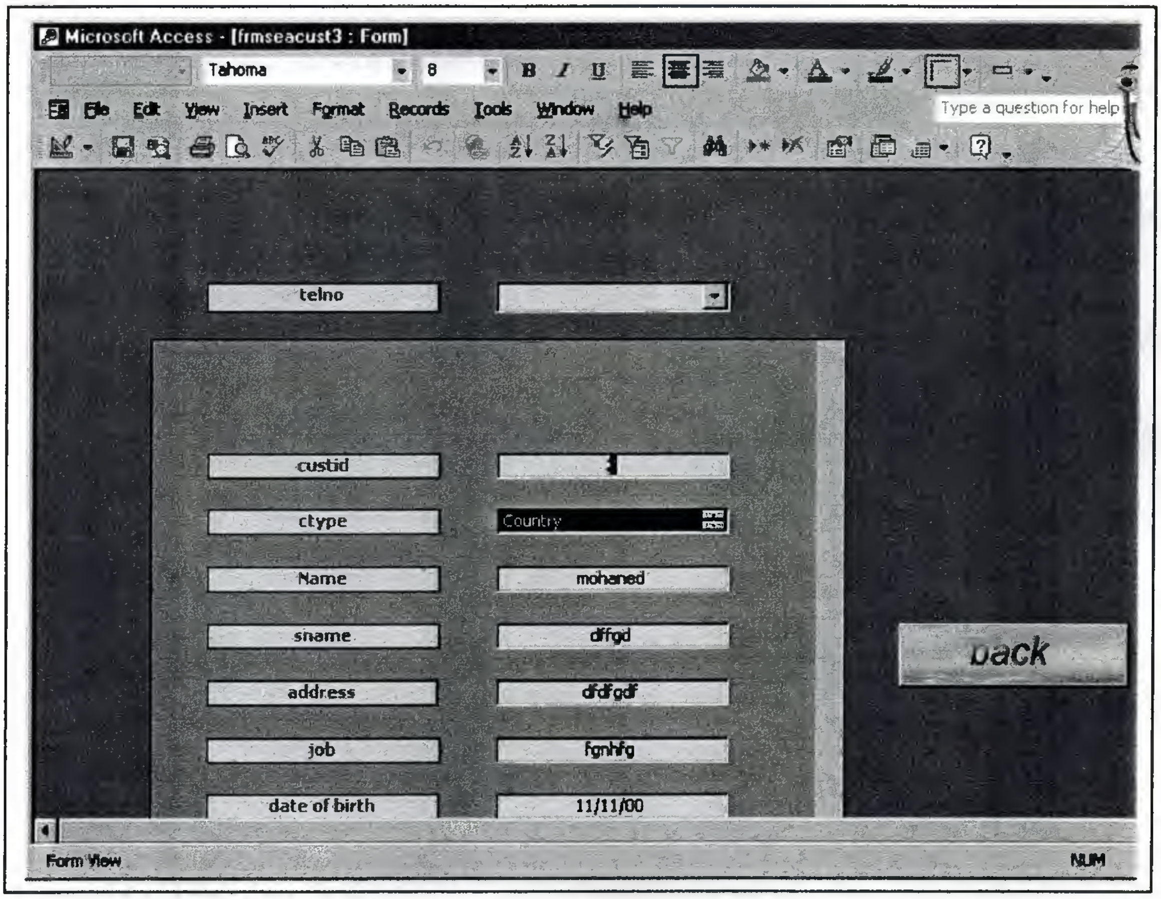
Number of columns to display

NUM

Figure 4.20.4.1 design "search custid"

4.20.5. Search Tel no Form

In this form you can see the search tel no form

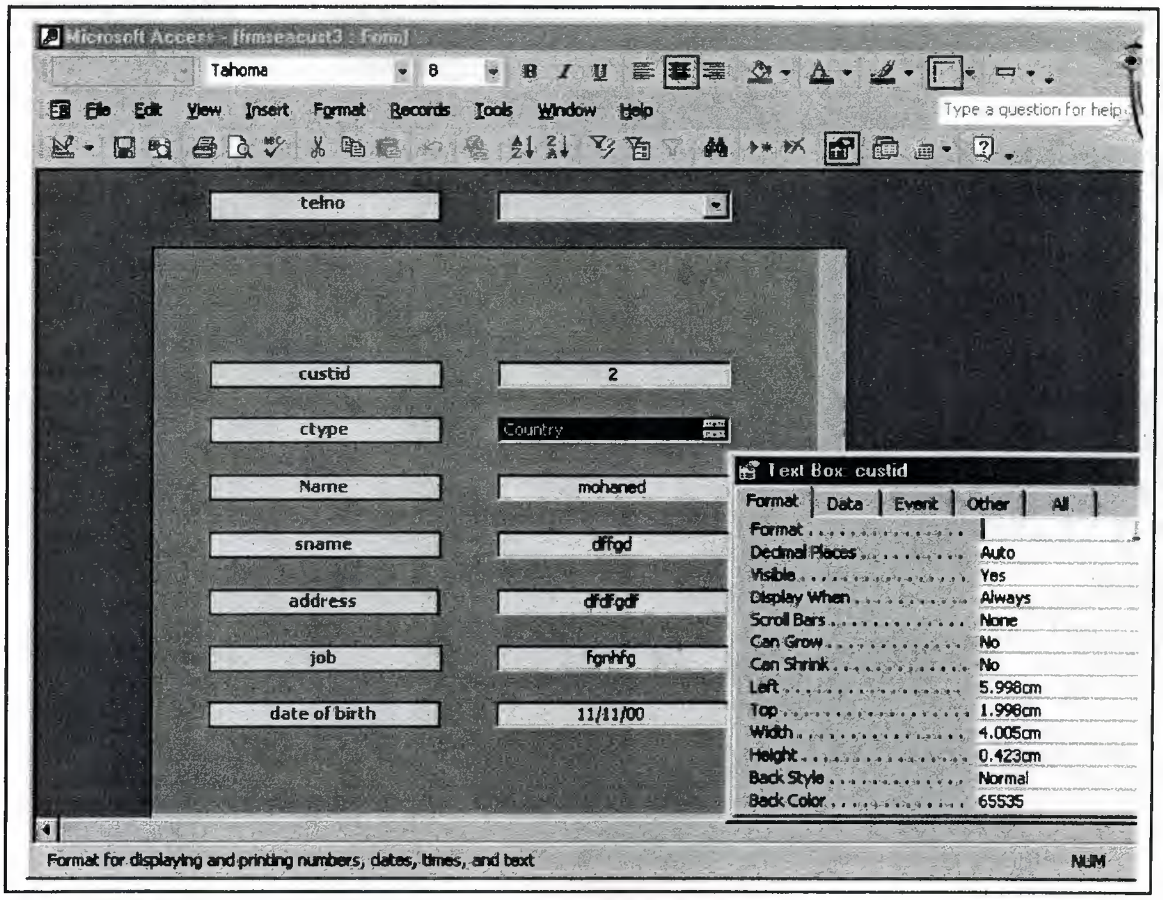


telno	
custid	
ctype	Country
Name	mohaned
sname	dffgd
address	dfdfgdf
job	fgnhfg
date of birth	11/11/00

Figure 4.20.5. form "search tel no"

4.20.5.1 Design Search Tel no Customer Form

In this form you can see the design tel no customer form



Microsoft Access - [frmseacust3 : Form]

telno	
custid	2
ctype	Country
Name	mohaned
sname	dffgd
address	dfdfgdf
job	fgnhfg
date of birth	11/11/00

Text Box: custid

Format	
Decimal Places	Auto
Visible	Yes
Display When	Always
Scroll Bars	None
Can Grow	No
Can Shrink	No
Left	5.998cm
Top	1.998cm
Width	4.005cm
Height	0.423cm
Back Style	Normal
Back Color	65535

Format for displaying and printing numbers, dates, times, and text

Figure 4.20.5.1 design “search tel no”

4.21. Search Employee Form

In this form you can see the search employee form

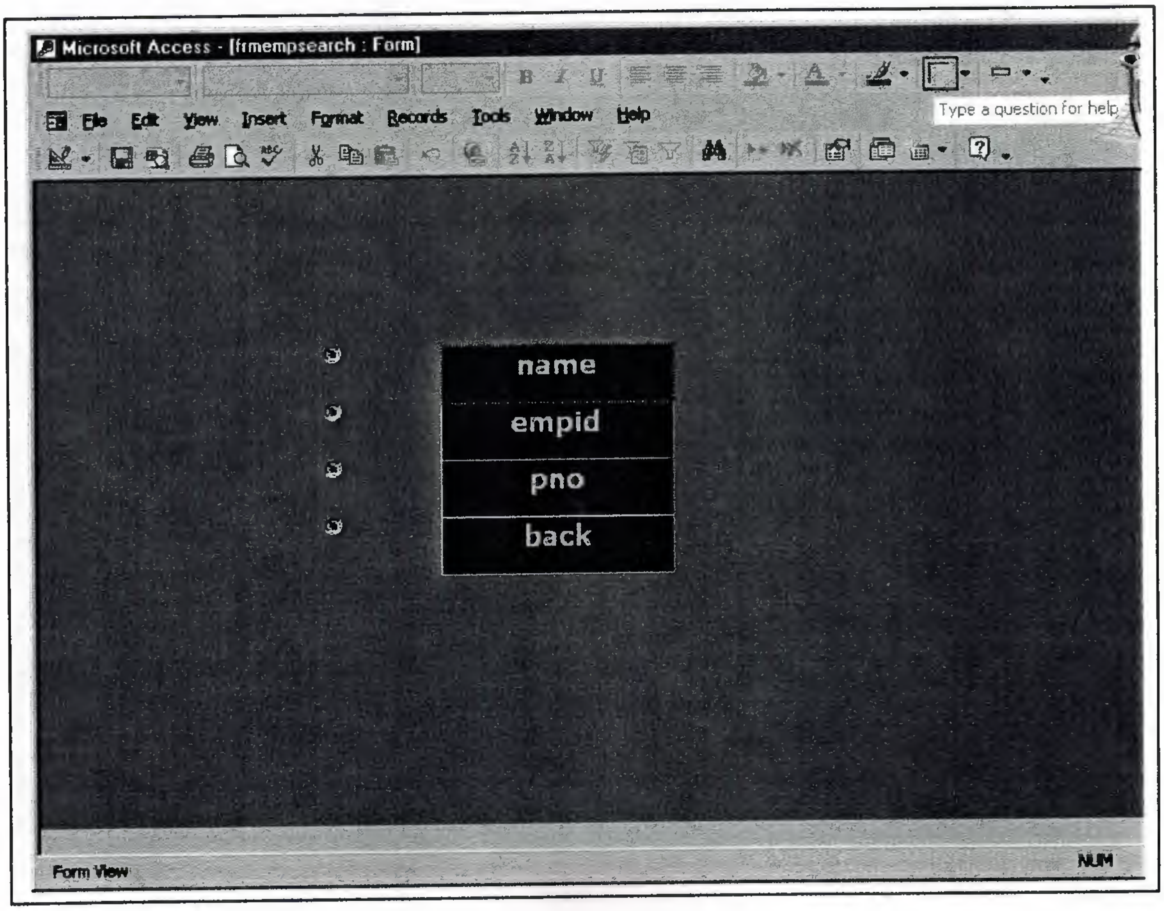
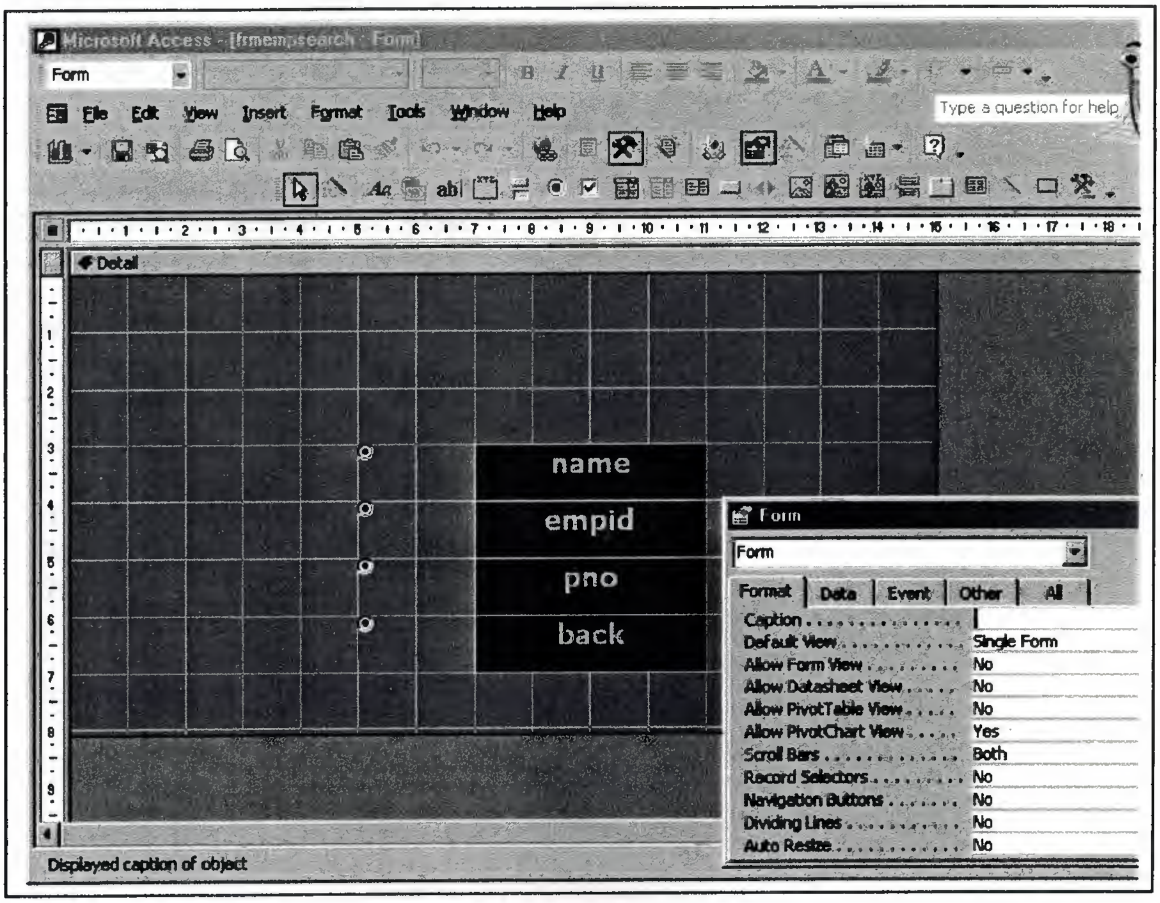


Figure 4.21 form "search employee"

4.21.1 Design Search Employee

In this form you can see the design search employee


Microsoft Access - [frmempsearch : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
name
empid
pno
back
Form
Format Data Event Other All

Form

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

Displayed caption of object

Figure 4.21.1 design "search employee"

4.21.2 Code Employee Customer

```
Option Compare Database

Private Sub Option0_Click()
DoCmd.OpenForm "frmseacust2"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Option20_Click ()
DoCmd.OpenForm "search"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Option4_Click ()
DoCmd.OpenForm "frmseacust1"
DoCmd.Close acForm, Me.Name
End Sub

Private Sub Option6_Click ()
DoCmd.OpenForm "frmseacust3"
DoCmd.Close acForm, Me.Name
End Sub
```

4.21.3 Search Name Employee Form

In this form you can see the search name employee form

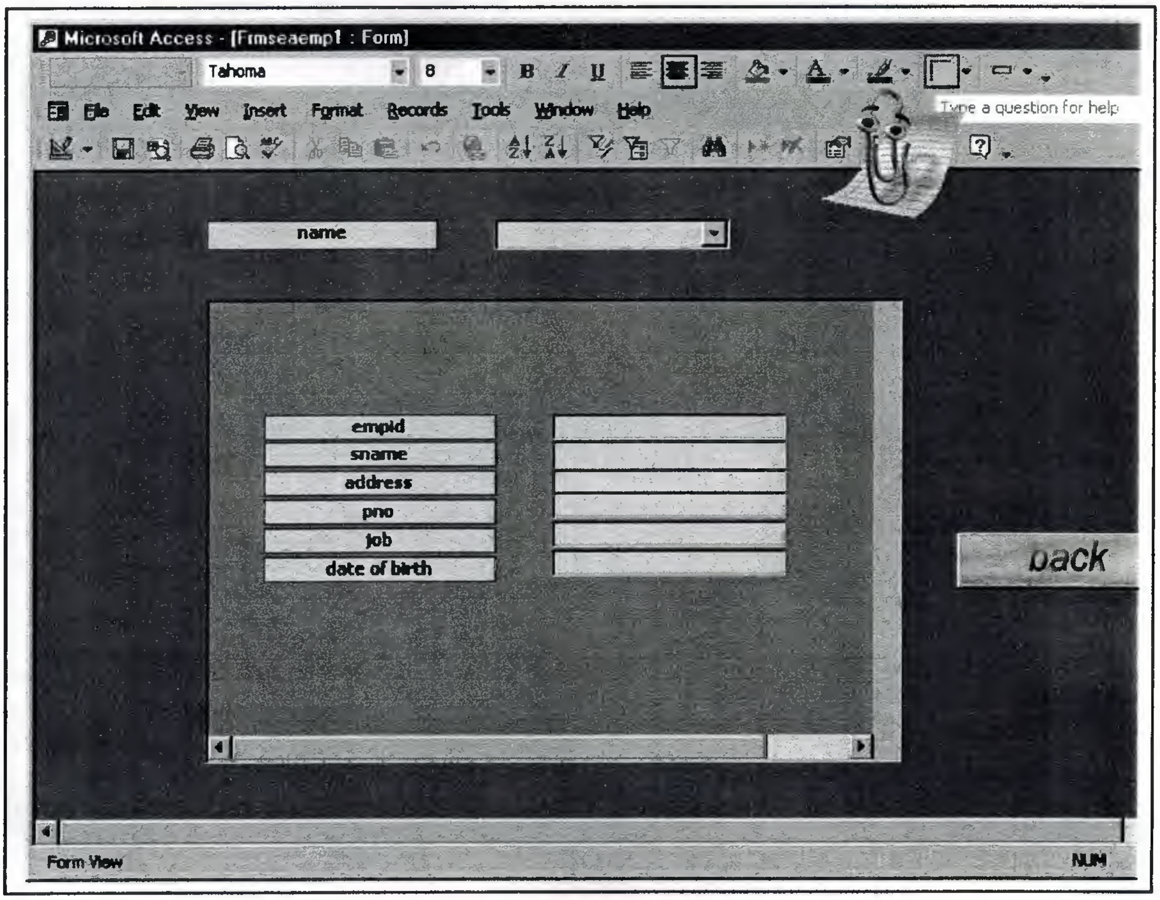
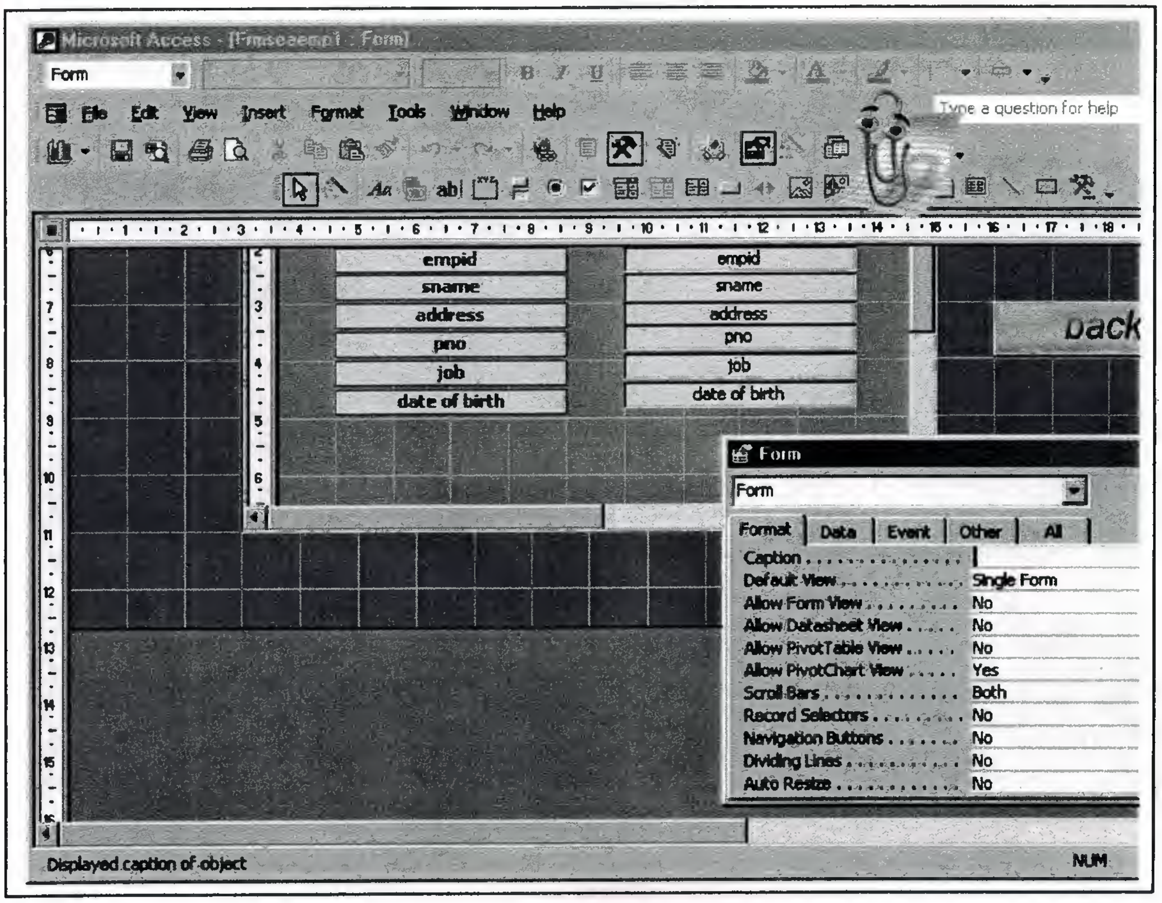

Microsoft Access - [Frmseaemp1 : Form]
Tahoma
8
File Edit View Insert Format Records Tools Window Help
Type a question for help
name
empid
sname
address
pno
job
date of birth
back
Form View
NUM

Figure 4.21.3 form "search name"

4.21.3.1 Design Search Name Employee

In this form you can see the design name employee

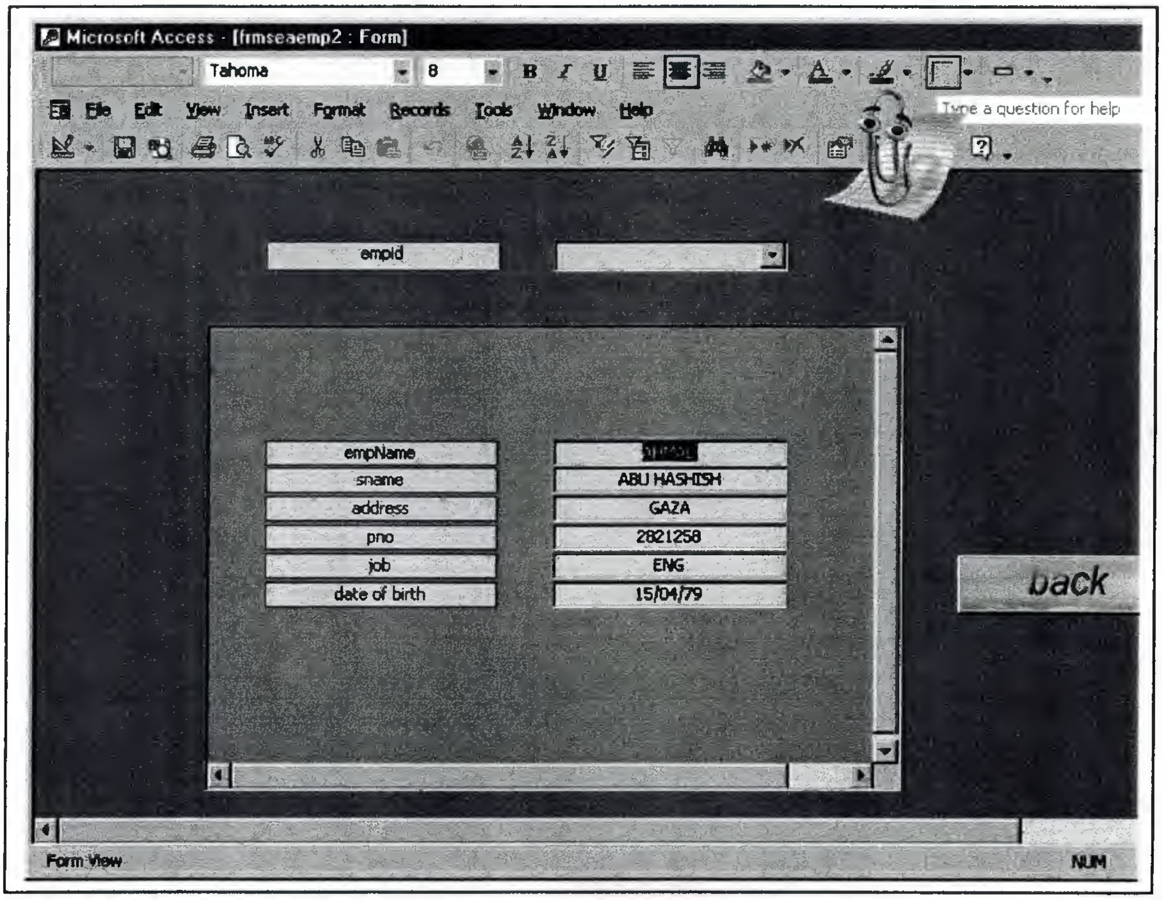


Caption . . . .
Default View . . . . Single Form
Allow Form View . . . . No
Allow Datasheet View . . . . . . No
Allow PivotTable View . . . . . No
Allow PivotChart View . . . . . Yes
Scroll Bars . . . . Both
Record Selectors . . . . No
Navigation Buttons . . . . No
Dividing Lines . . . . No
Auto Resize . . . . No

Figure 4.21.3.1 design “name employee”

4.21.4 Search Emp id Employee Form

In this form you can see the search emp id employee form

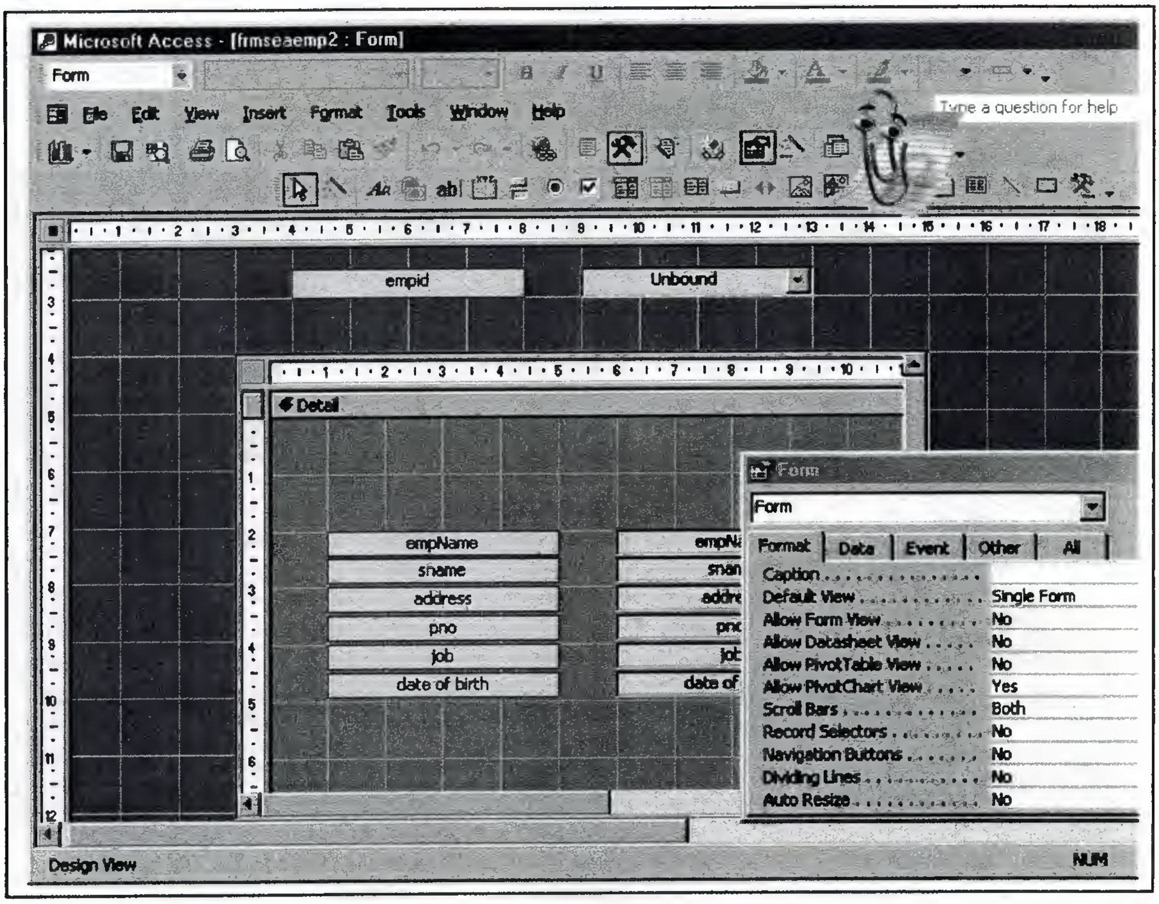


empid	
empName	[illegible]
sname	ABU HASHISH
address	GAZA
pno	2821258
job	ENG
date of birth	15/04/79

Figure 4.21.4 form "search emp id"

4.21.4.1 Design Search Emp id Employee

In this form you can see the design emp id employee

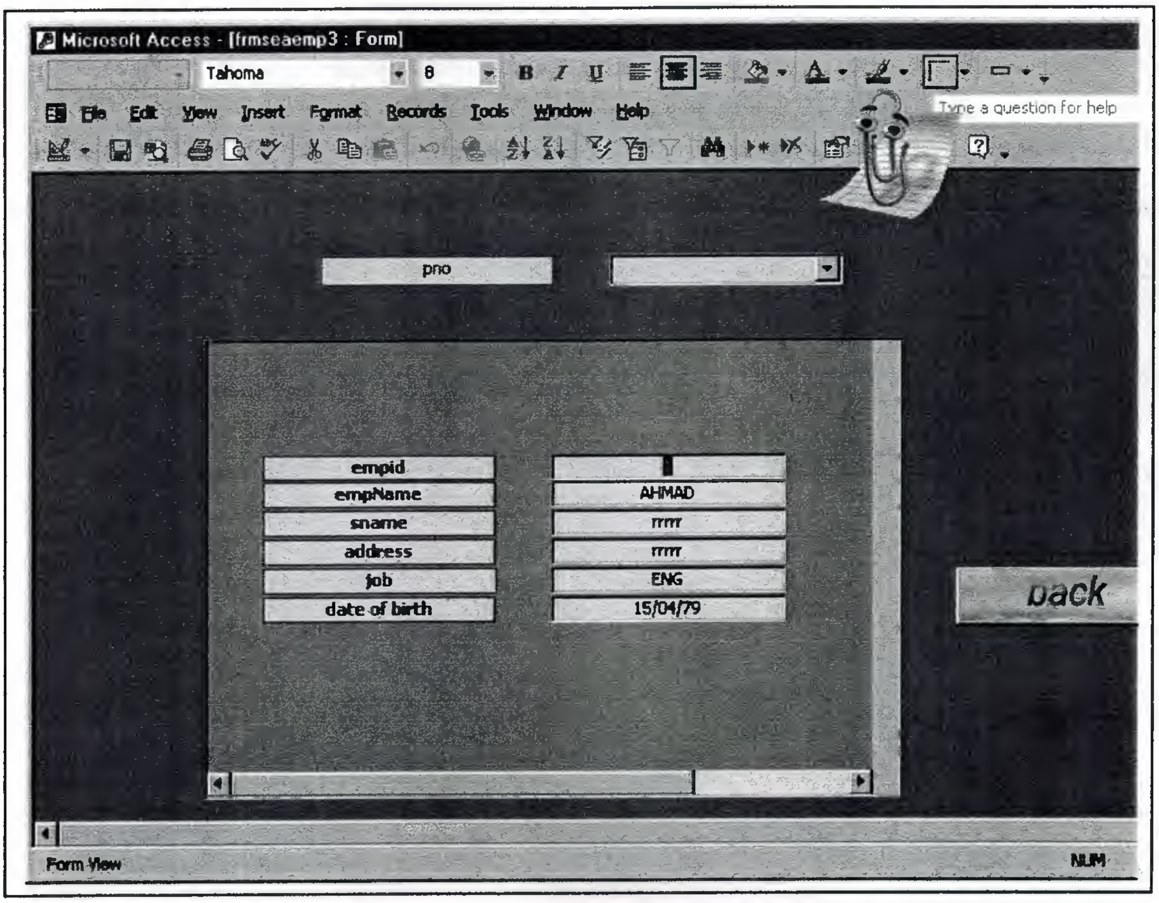


Caption . . . .
Default View . . . . Single Form
Allow Form View . . . . No
Allow Datasheet View . . . . . No
Allow PivotTable View . . . . . No
Allow PivotChart View . . . . Yes
Scroll Bars . . . . Both
Record Selectors . . . . No
Navigation Buttons . . . . No
Dividing Lines . . . . No
Auto Resize . . . . No

Figure 4.21.4.1 design "emp id employee"

4.21.5. Search Pno Employee Form

In this form you can see the pno employee form


Microsoft Access - [frmseaemp3 : Form]
Tahoma
8
File Edit View Insert Format Records Tools Window Help
Type a question for help
back
Form View
NUM

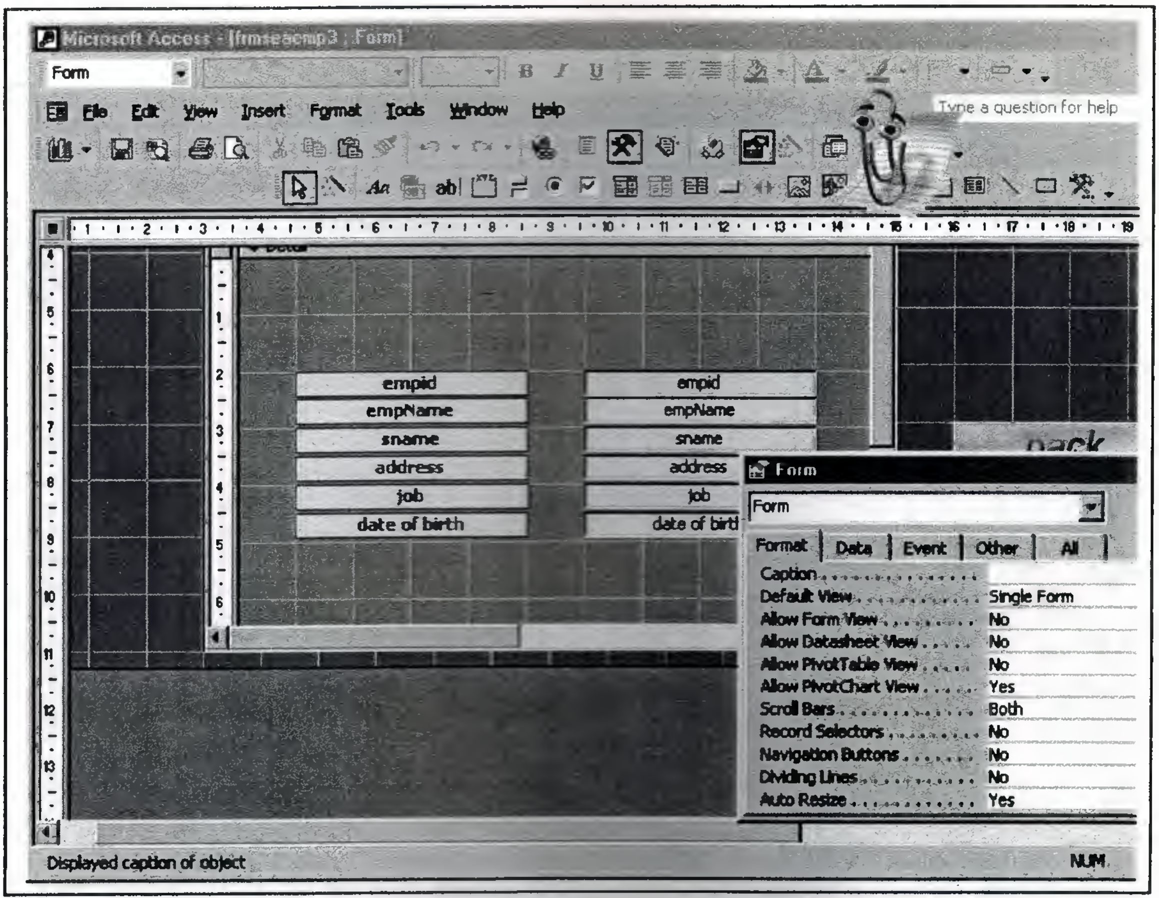
pno

empid	
empName	AHMAD
sname	rrrrr
address	rrrrr
job	ENG
date of birth	15/04/79

Figure 4.21.5.form design “pno”

4.21.5.1 Design Search Pno Employee

In this form you can see the design pno employee



Caption
Default View . . . . Single Form
Allow Form View . . . . No
Allow Datasheet View . . . . . No
Allow PivotTable View . . . . . No
Allow PivotChart View . . . . . Yes
Scroll Bars . . . . Both
Record Selectors . . . . No
Navigation Buttons . . . . No
Dividing Lines . . . . No
Auto Resize . . . . Yes

Displayed caption of object

Figure 4.21.5.1 design "pno"

4.22. Report Form

In this form you can see the report form

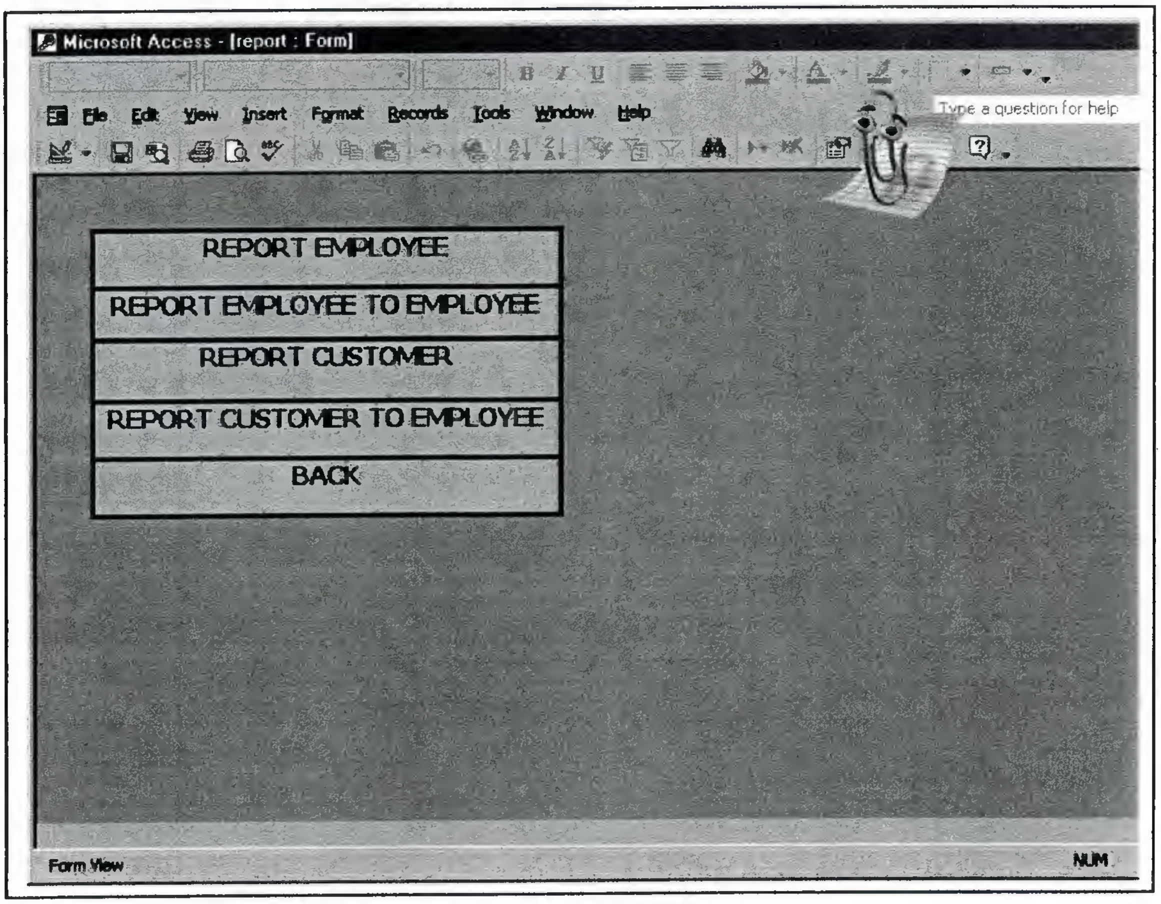
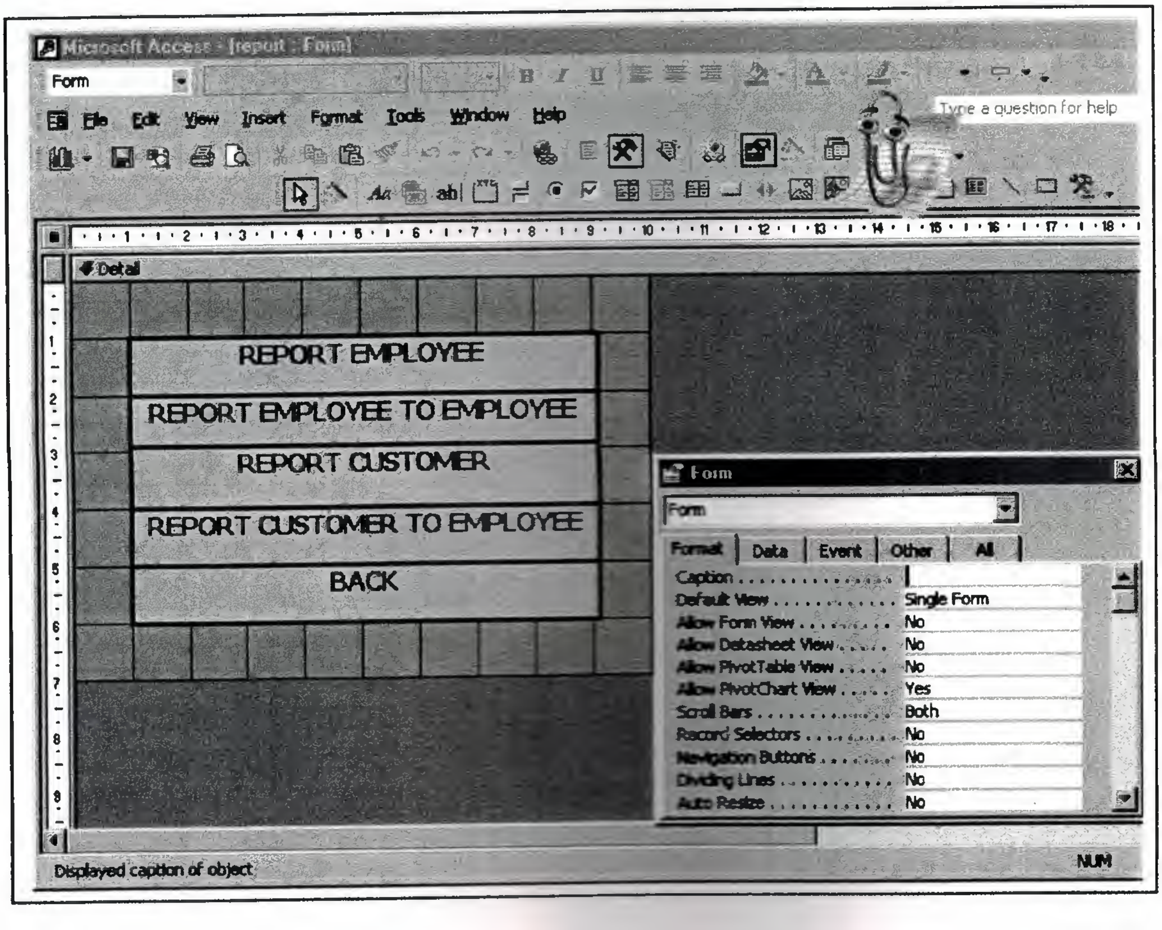


Figure 4.22 form “report”

4.22.1 Design Report

In this form you can see the design report


Microsoft Access - [report : Form]
Form
File Edit View Insert Format Tools Window Help
Type a question for help
Detail
REPORT EMPLOYEE
REPORT EMPLOYEE TO EMPLOYEE
REPORT CUSTOMER
REPORT CUSTOMER TO EMPLOYEE
BACK
Form
Form
Format Data Event Other All
Displayed caption of object
NUM

Caption	
Default View	Single Form
Allow Form View	No
Allow Datasheet View	No
Allow PivotTable View	No
Allow PivotChart View	Yes
Scroll Bars	Both
Record Selectors	No
Navigation Buttons	No
Dividing Lines	No
Auto Resize	No

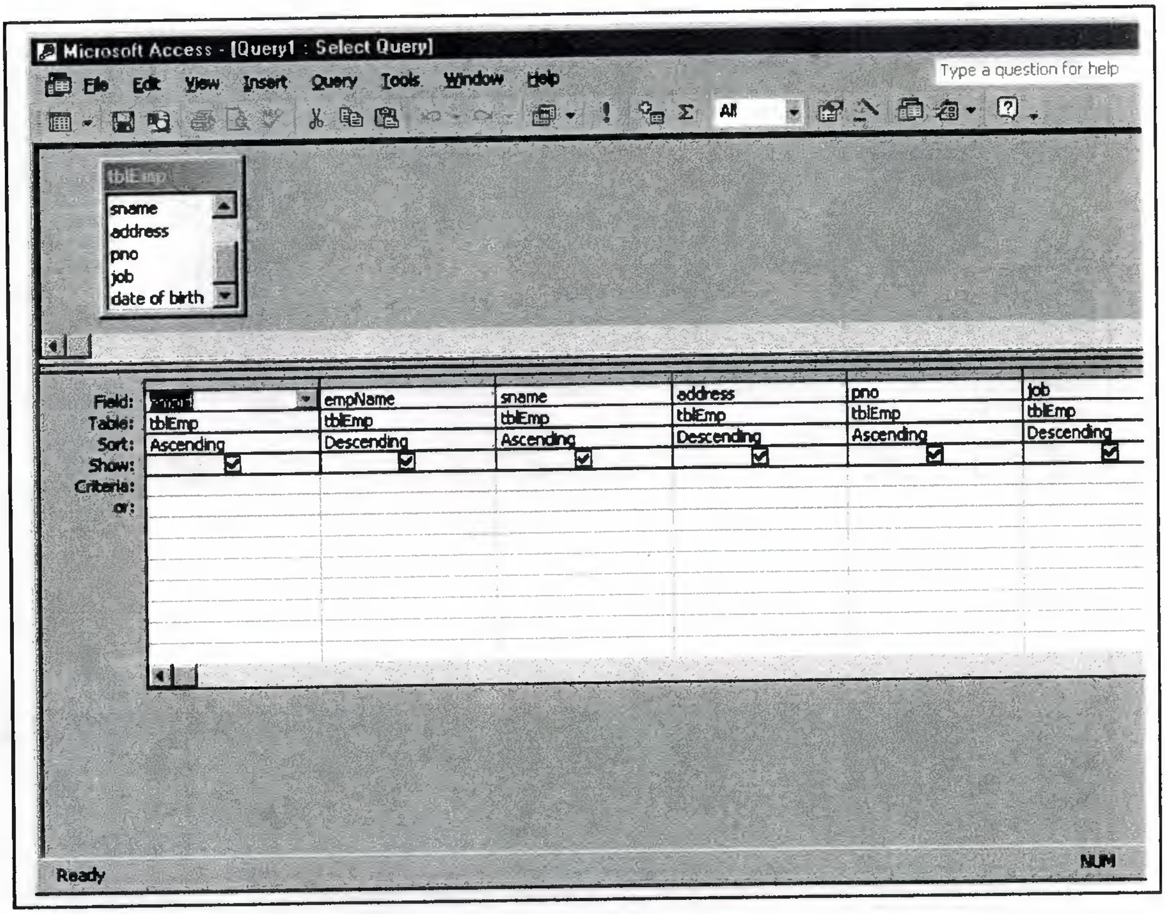
Figure 4.22.1 design "report"

4.22.2 Code Report

```
Option Compare Database

Private Sub Label78_Click ()
DoCmd.OpenReport "report1"
End Sub

Private Sub Label80_Click ()
DoCmd.OpenReport "report4"
End Sub

Private Sub Label81_Click ()
DoCmd.OpenReport "report2"
End Sub

Private Sub Label82_Click ()
DoCmd.OpenReport "report3"
End Sub

Private Sub Label83_Click ()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, Me.Name
End Sub
```

4.22.3 Employee Query

In this table you can see every thing about employee query

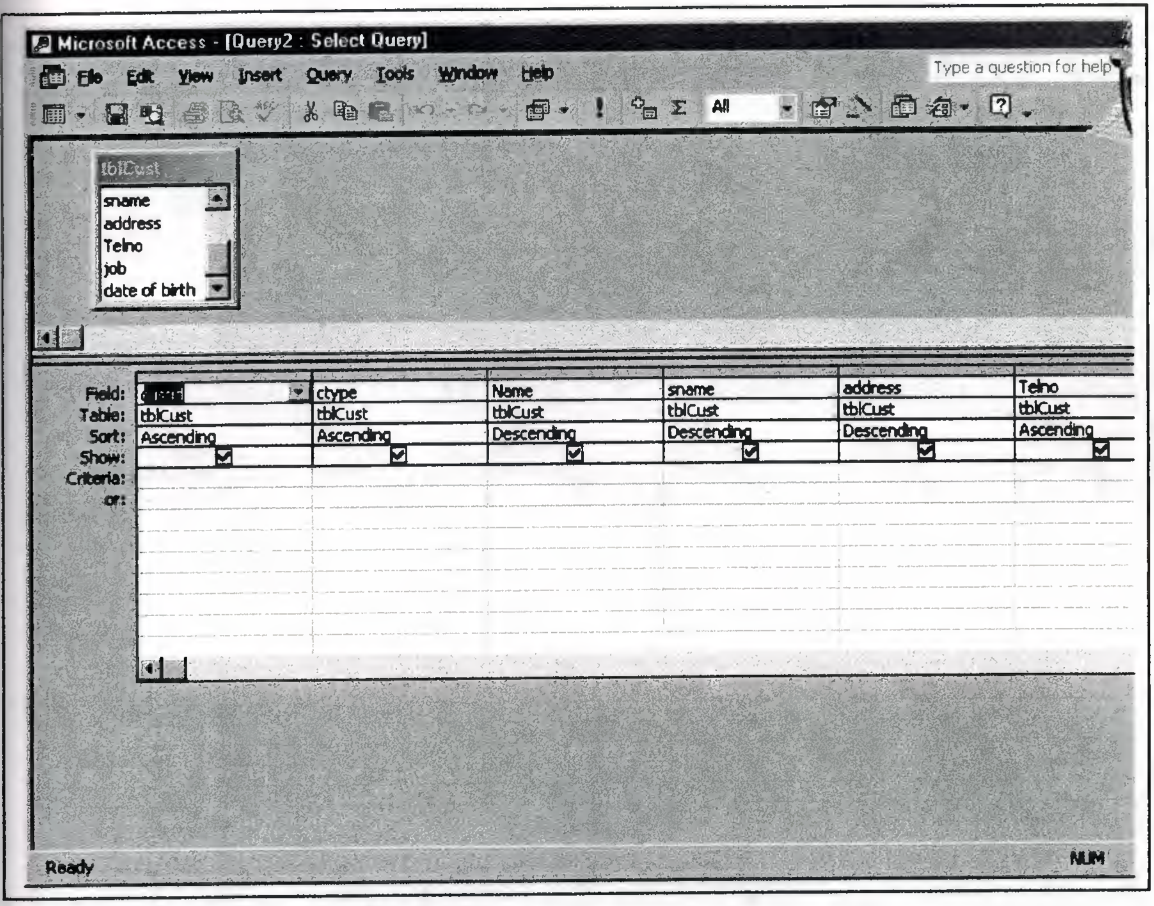


Field:	[illegible]	empName	sname	address	pno	job
Table:	tblEmp	tblEmp	tblEmp	tblEmp	tblEmp	tblEmp
Sort:	Ascending	Descending	Ascending	Descending	Ascending	Descending
Show:	☑	☑	☑	☑	☑	☑
Criteria:						
or:						

Figure 4.22.3 query "employee"

4.22.4 Customer Query

In this table you can see every thing about customer query

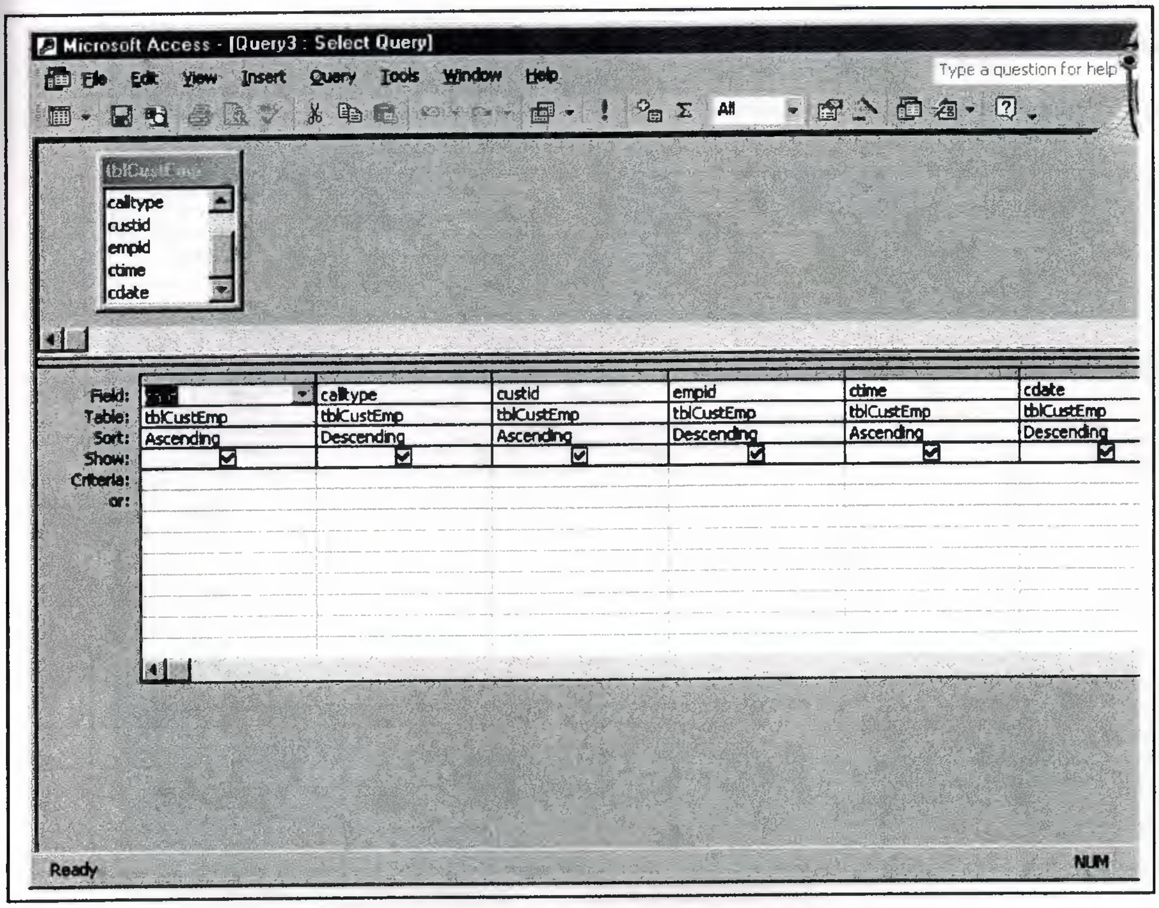


Field:	[illegible]	ctype	Name	sname	address	Telno
Table:	tblCust	tblCust	tblCust	tblCust	tblCust	tblCust
Sort:	Ascending	Ascending	Descending	Descending	Descending	Ascending
Show:	☑	☑	☑	☑	☑	☑
Criteria:						
or:						

Figure 4.22.4 query "customer"

4.22.5 Query Customer to Employee

In this table you can see every thing about customer to employee query



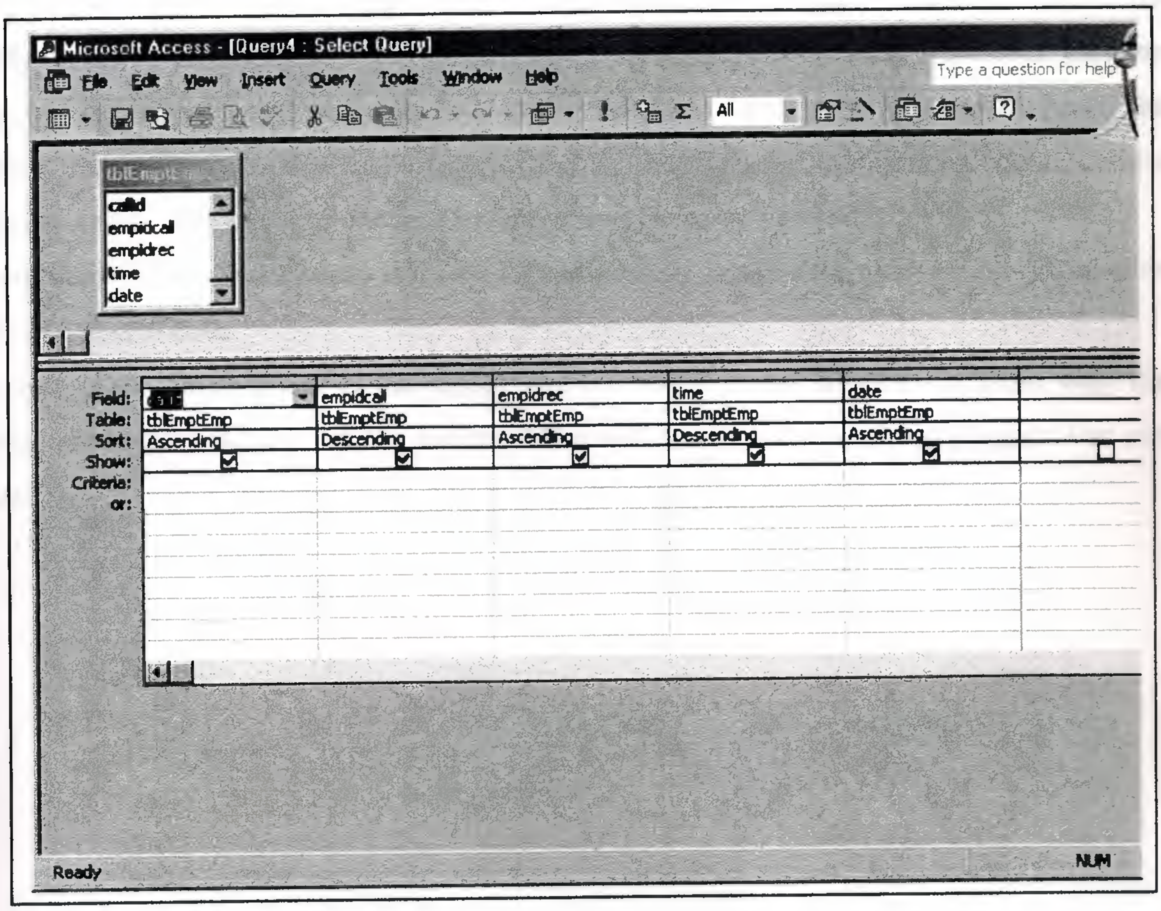
Microsoft Access - [Query3 : Select Query]
File Edit View Insert Query Tools Window Help
Type a question for help
All
tblCustEmp
calltype
custid
empid
ctime
cdate
Ready
NUM

Field:	[illegible]	calltype	custid	empid	ctime	cdate
Table:	tblCustEmp	tblCustEmp	tblCustEmp	tblCustEmp	tblCustEmp	tblCustEmp
Sort:	Ascending	Descending	Ascending	Descending	Ascending	Descending
Show:	☑	☑	☑	☑	☑	☑
Criteria:						
or:						

Figure 4.22.5 query "customer to employee"

4.22.6 Query Employee to Employee

In this table you can see every thing about employee to employee query



Field:	[illegible]	empidcall	empidrec	time	date	
Table:	tblEmptEmp	tblEmptEmp	tblEmptEmp	tblEmptEmp	tblEmptEmp	
Sort:	Ascending	Descending	Ascending	Descending	Ascending	
Show:	☑	☑	☑	☑	☑	□
Criteria:						
or:						

Figure 4.22.6 query “employee to employee”

CONCLUSION

When developing this project we see that **Microsoft access** is making the operation in database, and that's the reasons, why it's gaining interest. (Notice of most of the companies.)

With the information observed from the lecturer and our researchers for this topic **database**, is a convenient tool wit a wide range of useful ways to be used. Such examples can be mentioned several database cab be used at the same time, easy adjustments from the database code can be made within a few minutes by the keyboard, installed **Microsoft access** programs can be controlled or checked before within the once laboratory.

Microsoft access is very easy and safe for the workers which Microsoft access database or Microsoft access code.

I hope in the future found in the collage and university a new information about what the new in data base world, and we can see that Microsoft access is a very cheap program that can be used.