Screen Based Trading

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

B.Sc (Computer Technology)

OF BHARATHIAR UNIVERSITY

450

Submitted By

V. DEEPA	**REG.No.9727Q0009**
G. DEVAPRABHA	**REG. No.9727Q0010**
K.S. SATHYA THARANI	**REG.NO.9727Q0036**

Guided by

Mr.BASKARAN. K.R

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Kumaraguru College of Technology

COIMBATORE - 641 006

MARCH 2000

CERTIFICATE

This is to certify that this project work entitled

" SCREEN BASED TRADING"

submitted to Kumaraguru College of Technology, Coimbatore (affiliated to Bharathiar University) in partial fulfillment of the requirements for the award of degree of B.Sc(Computer Technology) is record of original work done by

Ms. V. Deepa	Reg No. 9727Q0009
Ms. G. Devaprabha	Reg No. 9727Q0010
Ms. K.S. Sathya Tharani	Reg No. 9727Q0036

during their period of study in the Depatment of Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore under my supervision and guidance and this project work has not formed the basis for the award of any Degree / Diploma / Associateship / Fellowship to any similar title or to any candidate of any University.

S. Jhanjee 28/3/2000

Professor & Head

Staff-in-Charge

Submitted for the University Examination held on 28-03-2000.

Internal Examiner

External Examiner

DECLARATION

We here by declare that the project work submitted by me for the award of Degree of B.Sc(Computer Technology) has not formed the basis for the award of any other Degree, Diploma, Associateship, Fellowship of similar titles and this dissertion was by us under the guidance of Mr.K.Baskaran.

Name of the candidates	**Reg No**	**Signature**
Deepa . V	9727Q0009	
Devaprabha . G	9727Q0010	
Sathya Tharani . K. S	9727Q0036	

Date : 28-03-2000

Place : Coimbatore

DEDICATED

TO

OUR

PARENTS

SYNOPSIS

The main objective of this project ***"screen-based trading"*** is to develop modules which are more interactive and user-friendly.

The operations of the Coimbatore Stock Exchange ***(CSX)*** had to be enhanced from FoxBASE system to one of the GUI packages, to bring in easy transactions among the brokers. The package that was chosen for this purpose was ***VISUAL BASIC 6.0*** with ***MICROSOFT ACCESS 7.0*** as the backend. Among the various operations that are carried out in the CSX, the main operation – ***" BIDS MATCHING "*** was developed for this project undertaken by us.

The registered brokers of the CSX submit their bids entries for buying or selling through the separate terminal provided for them. Thus the trading takes place by matching the bids offered. This helps in enhancing the trading speed.

CONTENTS

CONTENTS

INTRODUCTION

INTRODUCTION

PURPOSE:

The project is to automate the screen-based trading at Coimbatore Stock Exchange (CSX). Precisely, the CSX is a place where the scripts of various registered organizations are brought and sold by the members of the CSX through online trading.

SCOPE:

The software is so designed that the buyer and seller members can enter their buying rates and selling rates (offer rate) and when those two rates match width each other then trading takes place.

HISTORY OF THE ORGANIZATION:

Realizing the need for a Stock Exchange in Coimbatore, the Indian Chamber of Commerce and Industry Coimbatore (ICCIC), an apex body of industrial houses and entrepreneurs, under the leadership of Sri K.G. Balakrishnan, made strenuous efforts to start a Stock Exchange in Coimbatore. As result of their efforts, the Government of India (GOI) permitted the establishment of a Stock Exchange in the year 1991.

INFRASTRUCTURE

CSX is located at "Stock Exchange Building", 683-686, Trichy Road, Coimbatore 641005 which is a prominent and easily accessible location. CSX has a Trading Hall with an area of 3500 sq.ft., office Area of 3500 sq.ft., and about 5500 sq.ft., for members and investigators' lounges. CSX has also computerized its post trading operations by installing the Pentium File Server with 14 nodes and 25 user NetWare.

MEMBERS

Members are brokers who are allowed to trade through the CSX. CSX at present has 193 Members who are registered with Securities Exchange Board of India (SEBI).

Most of the members are professionals (MBAs, Doctors, Engineers, postgraduates etc) with experience in banks, merchant banking and reputed companies.

TYPES OF MEMBERS:

There is three types of members. They are

1. Individual members
2. Corporate members
3. Partnership members

Individual members :

The members who individually own a concern and trade under that name are known as individual members.

Corporate members:

The members who can appoint substitute for them and trade through them are known as corporate members.

Partnership members:

The members who has the concern along with one or more partners.

Board of directors:

1 Chairman

3 SEBI Nominees

5 Public Representatives

Functioning of the CSX

Settlement period:

The settlement period is the time in which a transaction of the member should be completed.

In Coimbatore Stock Exchange the settlement period is from Wednesday to Tuesday. The trading hours is from 10 A.M to 3.30 P.M.

Days	**Functions**
Wednesday	Trading
Thursday	Trading & sell-in auction
Friday	Trading & sell-out auction
Saturday	Trading
Sunday	Holiday
Monday	No trading & Documents in
Tuesday	Trading &Documents out &payment out.

In the settlement period, Wednesday is considered to be the first day of the settlement and only trading is done.

On Thursday, the trading will be done and the sell-in auction for the previous settlement will be done.

On Friday, trading will be done along with sell-out auction for the previous settlement will be conducted after 1.30 p.m. and the pre-delivery report will be prepared.

On Saturday, trading alone will be done and on Monday trading will not be done and documents of previous trading is submitted.

On Tuesday, trading is done and the documents are given out and payments are collected.

Sell-in auction:

If the seller of the Scripps is not able to submit the document on Saturday, then he can ask for a commitment period – a period of one settlement will be given as commitment period.

If the seller does not ask for commitment period then the Scripps are bought through the sell-in auction.

The auction rate will be fixed as (market price of the share + 20% of market price) and the offer with lowest

price will be selected. The default seller will pay the difference. If there is a gain in this transaction it will go to the CSX.

If the first seller did not settle the document even after the commitment period or the second seller (seller through the auction) then the respective person should pay the buyer (market price of share + 20% of market price) or (the price at which the share is bought + 20% of market price) whichever is higher. In that case the buyer will be able to enjoy only the privilege.

Sell-out auction:

If a member trades through the exchange and buys script and is not able to pay the amount to the CSX on Monday and get his document, he will be given time upto Thursday 11.30 a.m. with some fine. If he did not pay till 11.30 a.m. then his scrips will be auctioned on Thursday after 1.30 p.m.

The scrips are auctioned for the required amount and then the left documents will be given to the buyer. The scrips are auctioned by selecting the best bid offer.

SYSTEM CONFIGURATION

SYSTEM CONFIGURATION

HARDWARE SPECIFICATION

System:

Product : HCL

Processor : P-III

Operating system:

Windows version : WindowsNT Workstation 4.0

Windows mode : Enhanced

Memory:

Total memory : 64 MB (RAM)

Storage:

A: diskette drive 1.44MB

C: Hard drive 4.3GB

Input devices:

Keyboard:

Type : Enhanced

Number of keys : 107

Speed : 31ms

Delay : 0ms

Mouse:

Product : Microsoft

Speed : 2

Output devices:

Monitor :

Product : Samsung

Type : 14" digital color

Printer:

Product : Wipro EX 200+

Type : Laser

SOFTWARE SPECIFICATION

OPERATING SYSTEM

The project was developed in Windows NT, that supports both 32 bit and 16-bit application .The application can be run on any platform.

PLATFORM

The software used was Visual Basic 6.0 Enterprise Edition as Front-end and Microsoft Access 7.0 as Backend. Reports were generated using Crystal Reports version 5.0.

FRONTEND

INTRODUCTION TO VISUAL BASIC 6.0

Visual Basic was developed from the BASIC programming language. In the 1970's Microsoft started developing ROM-based interpreted basic for the early microprocessor-based computers. In 1982,Microsoft QUICK BASIC revolutionized BASIC and was legitimized as a serious development language for MS-DOS environment. Latter on, Microsoft Corporation created the enhanced version of BASIC called VISUAL BASIC for windows.

ABOUT VISUAL BASIC 6.0

Visual Basic 6.0 for Windows requires atleast Microsoft windows95 and windowsNT 3.51,486 processor and a minimum of 16 MB RAM.A complete installation of the most powerful version of Visual Basic 6.0,The Enterprise Edition requires more than 250 MB of the hard disk space.

Visual Basic 6.0 lets one to add menus, text boxes, command buttons, option buttons, check boxes, list boxes, scroll bars & file & directory boxes to blank Windows. One can also use grids to handle tabular data, communicate with Windows applications and access databases. It can have multiple Windows on a screen. These windows have full access to a clipboard.

One can use Visual Basic to communicate other applications running under Windows easily using the most modern version of Microsoft's OLE technology. One can create reusable objects, with their own properties and methods, and assemble them into an object model. It is also possible to use context or pop-up menus for forms, controls and codes, and attach add-ins for source code control and other features.

Data can be bound to more than one control like DBCombo, DBList and DBGrid controls. These provide automatic list management and instantaneous access to external data. Programs using object oriented technique can be developed. Third party toll can be added to the Visual Basic environment.

Conditional compilation for multi platform development is made easy.

Apart from all these, the Enterprise Edition of Visual Basic 6.0 includes new features designed to give developers the means to build client/server applications that manage business-critical data.

ADVANTAGES OF VISUAL BASIC 6.0

The main advantage of choosing a package like Visual Basic 6.0 is that it is a **GUI** tool, that is, Graphical User Interface that proves the proverb "**a picture is worth a thousand words**". The benefits of a Windows environment are an added advantage. Most of the operations are through the mouse. All these features make it easier to comprehend things in a quicker and an easier way.

Coding in GUIs environment is quite a transition to traditional, linear programming methods where the user is guided through a linear path of execution and is limited to a set of small operations. In a GUI environment, the number of options open to the user is much greater, allowing more freedom to the user and the developer. Features such as easier comprehension, user-friendliness, faster application development and many other aspects such as introduction to ActiveX technology and Internet features make Visual Basic tool to work with.

BACKEND

INTODUCTION TO MICROSOFT ACCESS 7.0

The backend that was used in this project was Microsoft Access 7.0.The backend contains the database in which the tables are stored. Tables are two-dimensional representation of data. Data are stored in the form of rows and columns called "tuples" and "domains" respectively.

ABOUT MICROSOFT ACCESS 7.0

The menus provided offer a number of functions. The records in the table can be arranged in ascending or descending order depending upon the user's requirement. Input masks available assure that data can be entered in a particular format. Properties such as Data Required, Allow Zero Length, Index with various options are also available. Relationships that are maintained between tables are extended to data also with the help of cascading updates and deletes.

ADVANTAGES OF MICROSOFT ACCESS 7.0

The advantages of using RDBMS [Access] are data security, data integrity, data independence and normalization. Security can be provided to each of the tables and hence duplication of tables can be restricted.

SYSTEM ANALYSIS

SYSTEM ANALYSIS

Objectives:

The operations of the Coimbatore Stock Exchange ***(CSX)*** had to be enhanced from FoxBASE system to one of the GUI packages, to bring in easy transactions among the brokers. The package that was chosen for this purpose was ***VISUAL BASIC 6.0*** with ***MICROSOFT ACCESS 7.0*** as the backend. Among the various operations that are carried out in the CSX, the main operation – ***" BIDS MATCHING "*** was developed for this project undertaken by us.

Existing System

The trading is done through trading using the software VERSATILE ENGINE FOR CENTRALISED TRADING REPORTING, provided by computer Maintenance Corporation(CMC) with effect from 9th October 1996. The system is capable of handling only 25,000 trades per day. All the members are provided with individual system and they are all connected in Local Area Network(LAN).

PROPOSED SYSTEM:

PROBLEM DEFINITION:

The major objectives of this Project Work entitled "SCREEN-BASEDTRADING" developed for the COIMBATORE STOCK EXCHANGE LIMITED are

- To help the Brokers to do the share trading business effectively.
- To find daily bargain rate for each company. This report will be very useful to analyse the trend in share price for each company over some period of time.
- To produce daily brokers performance reports. Based on these reports, the Stock Exchange can claim commission from Brokers.
- To increase transparency in the securities markets by allowing open access to all the investors.
- To promote investor protection, encourage market liquidity, and foster the efficiency of securities markets by facilitating price discovery and open competition, thus reducing the effects of fragmentation.

PROPOSED SYSTEM – OUTLINE

The computerized buying rates(bids) and selling rates(offers) matching system has been developed to implement on LAN. In this system, the brokers can come to the trading hall at stock exchange and enter their bids and offers. They can take the matching report for their bids and offers on the spot itself within some fraction of seconds.

From this system, the stock exchange can take daily bargain statement and daily Brokers performance report at any point of time.

ADVANTAGES OF PROPOSED SYSTEM

Growth of the trading volumes on Coimbatore Stock Exchange and the emergence of institutional investors, domestic as well as foreign are generating great pressure for establishment of screen based trading system.

Technological developments have permitted rapid change in the structure of securities trading markets. Because of the open access by all investors to all markets that technology permits, regulators need to examine the level of oversight necessary to ensure the protection of investors.

The high speed with which trades can be executed, and the large number of participants who can trade simultaneously allow faster incorporation of price sensitive information into the prevailing prices thus increasing the informational efficiency of the markets.

With the screen based trading system, it becomes possible for the market participants to see the full market , which helps to make markets more transparent and hence leads to increased investor confidence in them. Screen based trading systems also help to establish transparent audit trials which are extremely important given the volume and speed of trades that are common on such systems.

ORGANISATION OF PROPOSED SYSTEM

Automation facilitates increase transparency. In turn, transparency promotes investor protection, encourages market liquidity, and fosters, the efficiency of securities markets by facilitating price discovery and open competition, thus reducing the effects of fragmentation.

Internationally screen based trading systems which electronically match the buyer and the seller in order driven systems, or which find for down on time and cost and on risk of error, as well as on fraud. They have enabled distant participants to trade with each other improving the liquidity of markets.

SYSTEM DESIGN

SYSTEM DESIGN

PRODUCT OVERVIEW

This product consists of some modules namely master, Transactions, Reports etc. The modules, Masters and Transactions have the facilities to add the data, modify the existing data, Delete the existing data, search the data and view – print – save the checklist for existing data.

When search option is chosen, a help screen will be displayed, from that the user can select the data. Mostly, whenever help screen is required, the help screen will be displayed
The master module consists of two masters namely, Brokers and Scrip. The details of the brokers are maintained in Brokers master. New Brokers can be added into and amendments can be done on it. The details of public limited companies are maintained in Scrip master. Additions and amendments can be done on it.

The transaction module consists of three transactions namely, Buying entry, Selling entry and Memo of confirmation entry. Buying offers are entered in buying entry. Selling offers are entered in offer entry. Before exchanging the shares, the seller and the buyer should sign in an agreement called memo o confirmation. The Memo of confirmations is entered in Memo of confirmation entry.

The report module allows taking the reports like Matching reports for buyer rates to each Broker, Matching reports for offer to each Broker, Daily Bargain Statement and Daily Brokers Performance report. The Matching reports for bids and offers to each Broker will be saved in a specific name.

TABLE DESCRIPTION

Broker Master:

Field name	Type	Size
Code	Text	20
Name	Text	50
Member	Text	30
Date	Date/time	

Scrip Master:

Field name	Type	Size
Code	Text	20
Name	Text	50
Face Value	Double	
Lot	Integer	2
Listed	Text	1
List date	Date/time	

Buyer:

Field name	Type	Size
Scrip name	Text	50
Quantity	Double	
Rate	Double	
Scrip code	Text	20
Buy code	Text	20
Bdate	Date/time	
Btime	Date/time	
UTL	Double	
Status	Text	50
Contract	Text	50

Seller:

Field name	Type	Size
Scrip name	Text	50
Quantity	Double	
Rate	Double	
Scrip code	Text	20
Sell code	Text	20
Sdate	Date/time	
Stime	Date/time	
UTL	Double	
Status	Text	50
Contract	Text	50

Contract:

Field name	Type	Size
Scrip code	Text	20
Buyer	Text	50
Seller	Text	50
Quality	Double	
Rate	Double	
Date	Date/time	
Time	Date/time	
Reference number	Long	

Graph:

Field name	Type	Size
Open price	Double	
High	Double	
Close price	Double	
List date	double	

DATA DICTIONARY

Structured repository of data about data.

The data dictionary was built to maintain some structured space for the data flows, processes, and data stores. It promotes

- Documentation.
- Improved Analyst/User communication.
- Serves as common base during implementation.

DATA FLOW DIAGRAM

DATA FLOW DIAGRAM

Dataflow diagram is a graphical representation of a system that show data flows to from and within the system, processing function that change the data in some manner and the storage of data. These diagrams are nothing more than a network of related system functions that include from where information is received and to where it is sent. These diagrams are constructed using four basic symbols-*SQUARE,ARROW,CIRCLE,OPEN RECTANGLE.*

- A square defines a source (originator) or destination of system.
- An arrow identifies data flow – data in motion. It is a pipeline through which information flows.
- A circle or a "bubble" (or an oval bubble) represents a process that transforms incoming data flow(S) into outgoing data flow(s).
- An open rectangle is a data store – data at rest, or a temporary repository of data.

A data store symbol portrays a file or database in which data resides. Actually, it is a temporary storage position of data within the system itself. A data store

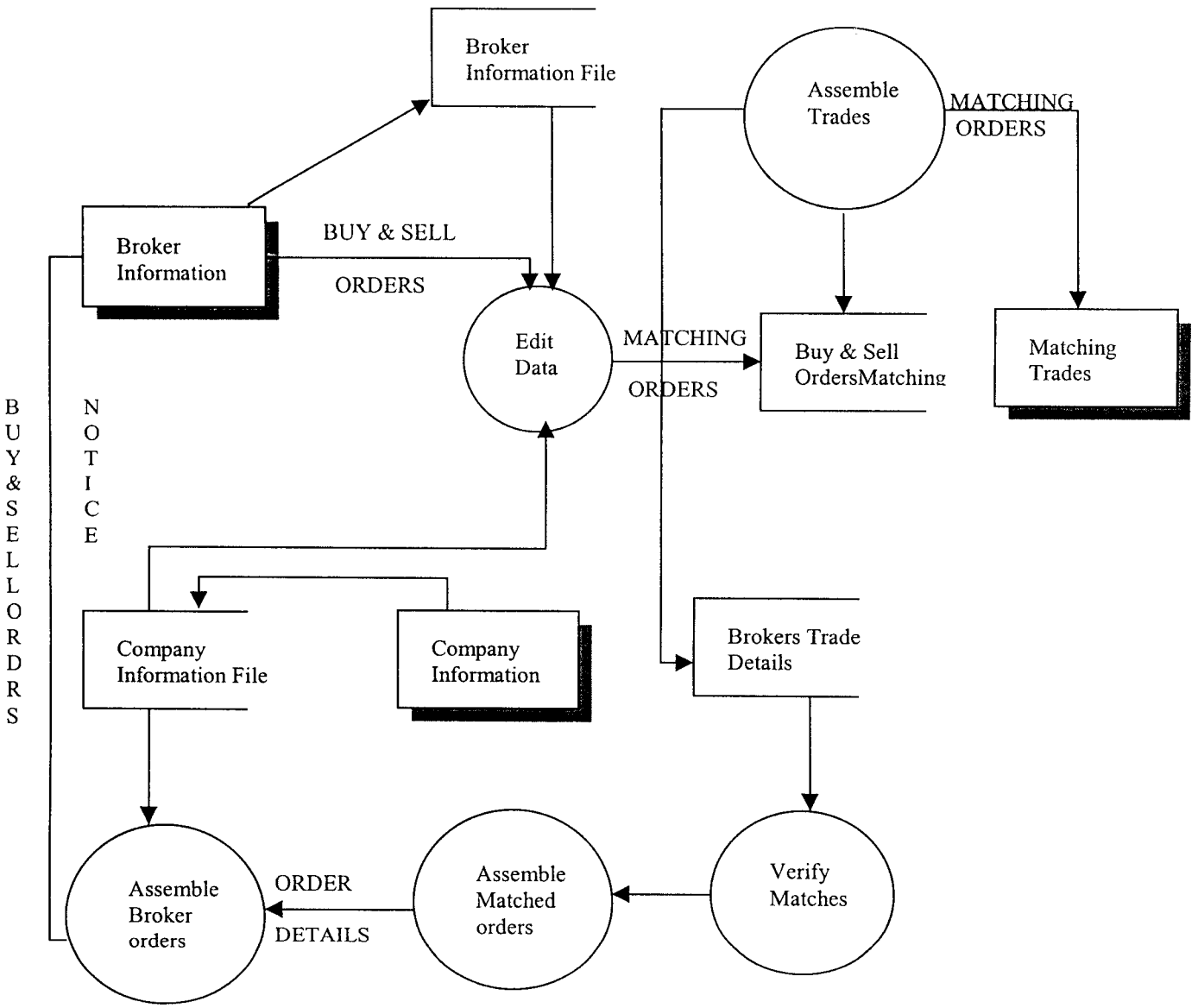
identifies a time delay for the contents of the data stored within.

The data flow symbol is used to show the flow or movement of data between process bubbles and other process, external entities, on data stores.

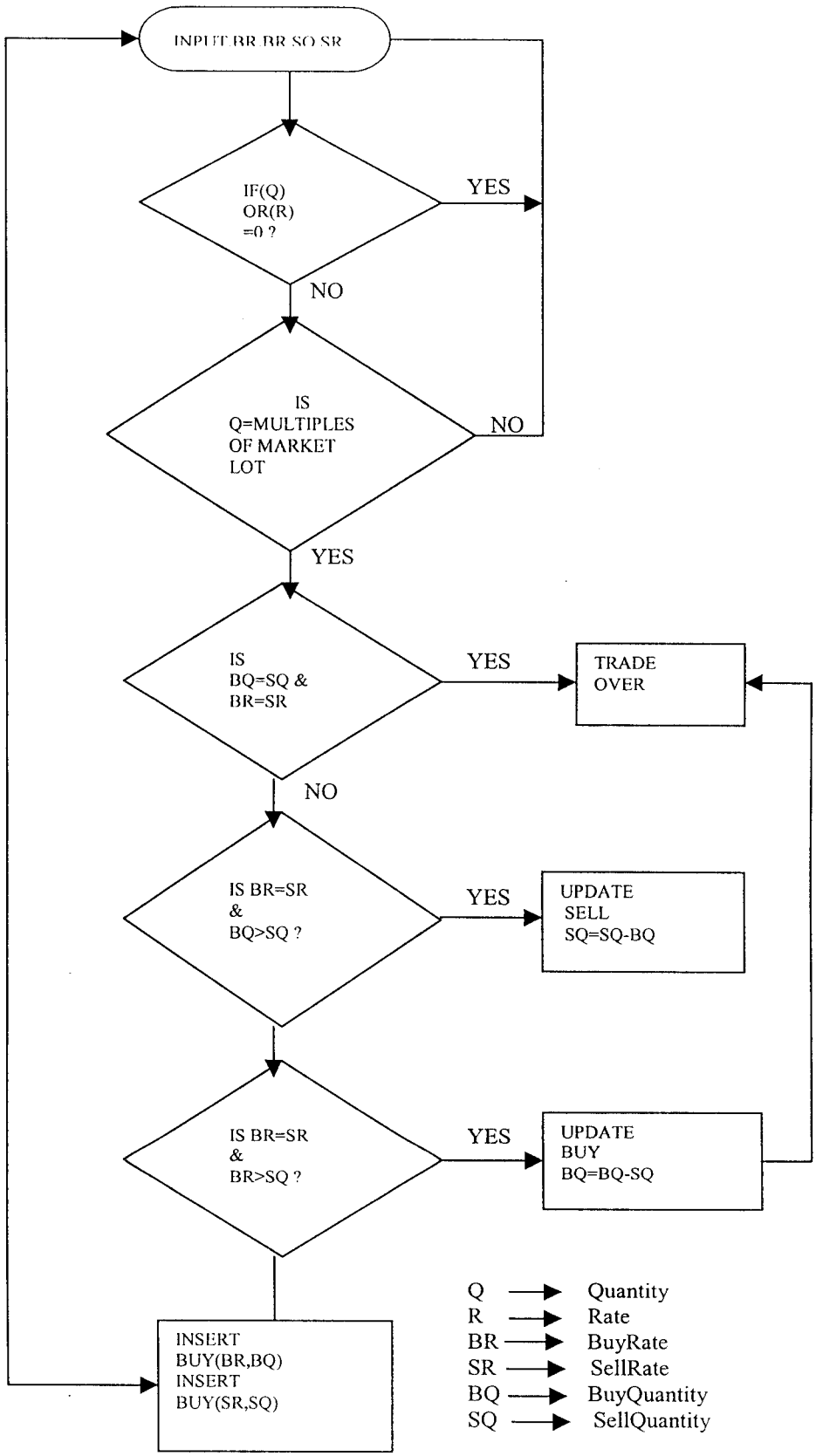
A data flow always must flow to or from a process. These diagrams are constructed to show various levels of details. These diagram are more detailed than context diagrams, data flow diagram are used to depict specific data flow from both the physical point view and logical point view.

An external entity is the originator or receiver of data or information. The originator or receiver of data is identified by writing the name of the external entity within the rectangular bit. External entities are also called interfaces, external interfaces, sinks sources or originators.

DATA FLOW DIAGRAM



Structure Chart



PROJECT DESCRIPTION

PROJECT DESCRIPTION

The CSX allots each broker a separate terminal through which they feed the bids offer. Each broker should pay a deposit at CSX while registering and the brokers will be able to trade only within that limit.

If the brokers opt to buy a script they have to feed the scrip name, quantity, and rate at which he like to buy. If any other broker has entered to sell that particular scrip at that particular rate then the bids are matched and transaction is completed.

If the brokers opt to sell a script they have to feed the scrip name, quantity, and rate at which he like to sell. If any other broker has entered to buy that particular scrip at that particular rate then the bids are matched and transaction is completed.

If more than one offer comes for the same rate then priority is fixed with respect to date and time at which the bids are entered.

The matched bids will be stored in the contract database. Sometimes the offer to buy or sell will be more than the available quantity then the available quantity will be matched and sent to the contract database and the balance will be sent to the pending or temporary database and wait until other offers come. The

unmatched bids entry can be cancelled by the brokers while the matched bids cannot be cancelled.

The reports generated are brokers turn over report, scrip turn over report, transaction report with respect to date and so on.

All the coding for the project was done using the GUI package Visual Basic 6.0. The special features offered by the package were made full use of. To start with, the PowerPoint presentation was included giving a brief idea of the operations carried out.

CODING

CHANGE PASSWORD

```
Private Sub Command1_Click()
Dim RstPass As Recordset
Set DB = OpenDatabase(App.Path & "\broker.mdb")
Set RstPass = DB.OpenRecordset("select * from Pass where code='" &
PASS.Text1 & "'")
If Text2.Text <> Text3.Text Then
MsgBox "ReEnter Password"
Text3.SetFocus
End If
RstPass.Edit
RstPass!PASS = Text2.Text
RstPass.Update
PASS.Show
Unload CHG
End Sub
Private Sub Command2_Click()
PASS.Show
End Sub
Private Sub Text1_LostFocus()
Dim RstPass As Recordset
Set DB = OpenDatabase(App.Path & "\broker.mdb")
Set RstPass = DB.OpenRecordset("select * from Pass where code='" &
PASS.Text1 & "'")
If Text1.Text <> RstPass!PASS Then
MsgBox "Enter Correct Password"
Text1.SetFocus
'Unload CHG
Else
Text2.SetFocus
End If
End Sub


Private Sub Text3_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = 13 Then
   Command1.SetFocus
End If
End Sub

Private Sub Text3_LostFocus()
```

```
If Text2.Text <> Text3.Text Then
MsgBox "ReEnter Password"
Text3.SetFocus
End If
End Sub

Public SA, SA1 As String

Private Sub Command1_Click()
Dim RstPass As Recordset
Set DB = OpenDatabase(App.Path & "\broker.mdb")
Set RstPass = DB.OpenRecordset("select * from Pass where Code='" &
Text1 & "'")
Dim RSTNAME As Recordset

If Trim(Text2.Text) = RstPass!PASS Then
gCODE = Text1
Set RSTNAME = DB.OpenRecordset("SELECT NAME FROM
BROKERMASTER WHERE CODE='" & gCODE & "'")
gNAME = RSTNAME.Fields(0)
Unload PASS
MdiCtx.Caption = gNAME
MdiCtx.Show
Else
MsgBox "REENTER PASSWORD"
End If
Text2.Text = ""

End Sub

Private Sub Command2_Click()
Text1.Text = ""
Text2.Text = ""
Text1.SetFocus
End Sub

Private Sub Command3_Click()
CHG.Show
End Sub

Private Sub Form_Load()
Text2.Text = ""
End Sub
```

```
Set BRO(5) = brotb.CreateField("LISTDATE", dbDate)

'APPEND FIELDS
brotb.Fields.Append BRO(0)
brotb.Fields.Append BRO(1)
brotb.Fields.Append BRO(2)
brotb.Fields.Append BRO(3)
brotb.Fields.Append BRO(4)
brotb.Fields.Append BRO(5)

'APPEND TABLE TO DATABASE
mydb.TableDefs.Append brotb

MsgBox "WORKSPACE CREATED", vbOKOnly + vbInformation

Set RSSCRMASTER = mydb.OpenRecordset("SELECT * FROM
SCRIPTMASTER")
Set RSBROKER = mydb.OpenRecordset("SELECT * FROM " &
TXTCODE)

'FOR DETAIL TABLE
RSBROKER.AddNew
RSBROKER.Fields(0) = TXTCODE
RSBROKER.Fields(1) = TXTNAME
RSBROKER.Fields(2) = TXTFACEVALUE
RSBROKER.Fields(3) = TXTLOT
RSBROKER.Fields(4) = LSTOPT.Text
RSBROKER.Fields(5) = TXTLASTDATE
RSBROKER.Update

'FOR MASTERTABLE
RSSCRMASTER.AddNew
RSSCRMASTER.Fields(0) = TXTCODE
RSSCRMASTER.Fields(1) = TXTNAME
RSSCRMASTER.Fields(2) = TXTFACEVALUE
RSSCRMASTER.Fields(3) = TXTLOT
RSSCRMASTER.Fields(4) = LSTOPT.Text
RSSCRMASTER.Fields(5) = TXTLASTDATE
RSSCRMASTER.Update

MsgBox "SUBMITTED", vbOKOnly + vbInformation

CLEAR_BROKER
```

SCRIPT

```
Private Sub CMDCLOSE_Click()
Unload Me
End Sub

Private Sub CMDSUBMIT_Click()

'DECLARATIONS
Dim mydb As Database
Dim RSBROKER As Recordset
Dim BROKER As TableDef
Dim BRO(6) As Field
Dim RSSCRMASTER As Recordset

'EXECUTED WHEN ERROR ENCOUTERED
On Error GoTo AA

'CHECK OPTIONS SELECTED
If LSTOPT.SelCount > 1 Then
   MsgBox "SELECT ANY ONE", vbOKOnly + vbCritical
   CLEAR_BROKER
   Exit Sub
End If


'SETS DATABASENAME
Set mydb = OpenDatabase(App.Path & "\SCRIPT.MDB")

'SETS TABLENAME
Set brotb = mydb.CreateTableDef(TXTCODE)

'SETS FIELDS
Set BRO(0) = brotb.CreateField("CODE", dbText)
Set BRO(1) = brotb.CreateField("NAME", dbText, 50)
Set BRO(2) = brotb.CreateField("FACEVALUE", dbDouble)
Set BRO(3) = brotb.CreateField("LOT", dbInteger)
Set BRO(4) = brotb.CreateField("LISTED", dbText, 1)
```

```
RSBROKER.AddNew
RSBROKER.Fields(0) = TXTCODE
RSBROKER.Fields(1) = TXTNAME
RSBROKER.Fields(2) = TXTMEMBER
RSBROKER.Fields(3) = TXTDATE
RSBROKER.Update
RSBROMASTER.AddNew
RSBROMASTER.Fields(0) = TXTCODE
RSBROMASTER.Fields(1) = TXTNAME
RSBROMASTER.Fields(2) = TXTMEMBER
RSBROMASTER.Fields(3) = TXTDATE
RSBROMASTER.Update
MsgBox "SUBMITTED", vbOKOnly + vbInformation
CLEAR_BROKER
Exit Sub
AA:
  MsgBox "INVALID CODE", vbOKOnly + vbCritical
End Sub

Public Sub CLEAR_BROKER()
TXTCODE = ""
TXTNAME = ""
TXTMEMBER = ""
TXTDATE = ""
End Sub
```

BROKER

```
Option Explicit

Private Sub CMDCLOSE_Click()
Unload Me
End Sub

Private Sub CMDSUBMIT_Click()
Dim mydb As Database
Dim RSBROKER As Recordset
Dim RstPass As Recordset
Dim brotb As TableDef
Dim BRO(4) As Field
Dim RSBROMASTER As Recordset
On Error GoTo AA
Set mydb = OpenDatabase(App.Path & "\BROKER.MDB")
Set RstPass = mydb.OpenRecordset("SELECT * FROM PASS")
RstPass.AddNew
   RstPass!CODE = TXTCODE
   RstPass!PASS = "CTX" & TXTCODE
RstPass.Update
Set brotb = mydb.CreateTableDef(TXTCODE)
Set BRO(0) = brotb.CreateField("CODE", dbText)
Set BRO(1) = brotb.CreateField("NAME", dbText, 50)
Set BRO(2) = brotb.CreateField("MEMBER", dbText, 50)
Set BRO(3) = brotb.CreateField("DATE", dbDate)
brotb.Fields.Append BRO(0)
brotb.Fields.Append BRO(1)
brotb.Fields.Append BRO(2)
brotb.Fields.Append BRO(3)
mydb.TableDefs.Append brotb
MsgBox "WORKSPACE CREATED", vbOKOnly + vbInformation
Set RSBROMASTER = mydb.OpenRecordset("SELECT * FROM
BROKERMASTER")
Set RSBROKER = mydb.OpenRecordset("SELECT * FROM " &
TXTCODE)
```

```
Exit Sub
AA:
  MsgBox "INVALID CODE", vbOKOnly + vbCritical
End Sub

Public Sub CLEAR_BROKER()
TXTCODE = ""
TXTNAME = ""
TXTFACEVALUE = ""
LSTOPT.Selected(0) = False
LSTOPT.Selected(1) = False

TXTLOT = ""
TXTLASTDATE = ""
End Sub
```

CONTRACT

```
Option Explicit

Private Sub Form_Load()
datContaract.DatabaseName = App.Path & "\entry.mdb"
msContract.ColWidth(0) = 1200

msContract.ColWidth(1) = 3000
msContract.ColWidth(6) = 1200
msContract.ColWidth(7) = 1250

msContract.TextMatrix(0, 0) = "ScriptCode"
msContract.TextMatrix(0, 1) = "          ScriptName "
msContract.TextMatrix(0, 2) = "BuyCode"
msContract.TextMatrix(0, 3) = "SellCode"
msContract.TextMatrix(0, 4) = "Quantity"
msContract.TextMatrix(0, 5) = "  Rate"
msContract.TextMatrix(0, 6) = "  Date"
msContract.TextMatrix(0, 7) = "  Time"
msContract.TextMatrix(0, 8) = "RefNo"

End Sub
```

PENDING

```
Option Explicit

Private Sub Form_Load()
Dim i As Integer
MSbUY.ColWidth(0) = 1000
MSbUY.ColWidth(1) = 3000
MSbUY.ColWidth(2) = 1200
MSbUY.ColWidth(3) = 1300
MSbUY.ColWidth(4) = 1000
MSbUY.ColWidth(5) = 1000

For i = 0 To 5
MSbUY.ColAlignment(i) = flexAlignCenterCenter
Next


MsSell.ColWidth(0) = 1000
MsSell.ColWidth(1) = 3000
MsSell.ColWidth(2) = 1200
MsSell.ColWidth(3) = 1300
MsSell.ColWidth(4) = 1000
MsSell.ColWidth(5) = 1000


End Sub
```

REPORTS

```
Option Explicit

Private Sub Command1_Click()
Dim DB As Database
Dim RS, RS1 As Recordset
 If TXTTO = "" Or TXTFROM = "" Then
   MsgBox "ENTER DATE", vbCritical + vbOKOnly
   Exit Sub
 End If
Set DB = OpenDatabase(App.Path & "\ENTRY.MDB")
DB.Execute "DELETE * FROM TMPRPT"

DB.Execute " INSERT INTO TMPRPT " _
      & "SELECT BCODE AS CODE ,SUM(BUYEDQTY*BRATE) AS
BQTY FROM TMPTRANSACTION WHERE tDate >= #" &
Format(TXTFROM, "DD/MM/yyYY") & "# AND tDate <= #" &
Format(TXTTO, "DD/MM/yyYY") & "#  GROUP BY BCODE"

Set RS1 = DB.OpenRecordset("SELECT
SCODE,SUM(SOLDQTY*BRATE)FROM TMPTRANSACTION
WHERE tDate >= #" & Format(TXTFROM, "DD/MM/yyyy") & "#
AND tDate <= #" & Format(TXTTO, "DD/MM/yyyy") & "#  GROUP
BY SCODE")
Set RS = DB.OpenRecordset("SELECT * FROM TMPRPT")



If Not (RS1.EOF And RS1.BOF) Then
   RS1.MoveFirst
   Do While Not RS1.EOF
   RS.FindFirst "CODE='" & RS1!sCode & "'"
   If RS.NoMatch = True Then
      RS.AddNew
   Else
      RS.Edit
   End If
      RS.Fields(0) = RS1.Fields(0)
      RS!SQTY = RS1.Fields(1)
```

```
      RS.Update
   RS1.MoveNext
   Loop
End If
If Len(dbcbroker) > 0 Then
   CrystalReport1.SelectionFormula = "{TMPRPT.CODE}='" &
dbcbroker & "'"
Else
   CrystalReport1.SelectionFormula = ""
End If
CrystalReport1.ReportFileName = App.Path & "\bROWISE.RPT"
CrystalReport1.DiscardSavedData = True
CrystalReport1.Action = 1

End Sub

Private Sub Command2_Click()
Dim DB As Database
Dim RS, RS1 As Recordset

Set DB = OpenDatabase(App.Path & "\ENTRY.MDB")
DB.Execute "DELETE * FROM TMPRPT"

DB.Execute " INSERT INTO TMPRPT " _
      & "SELECT BCODE AS CODE ,SUM(BUYEDQTY*BRATE) AS
BQTY FROM TMPTRANSACTION GROUP BY BCODE"

Set RS1 = DB.OpenRecordset("SELECT
SCODE,SUM(SOLDQTY*BRATE)FROM TMPTRANSACTION
GROUP BY SCODE")
Set RS = DB.OpenRecordset("SELECT * FROM TMPRPT")



If Not (RS1.EOF And RS1.BOF) Then
   RS1.MoveFirst
   Do While Not RS1.EOF
   RS.FindFirst "CODE='" & RS1!sCode & "'"
   If RS.NoMatch = True Then
      RS.AddNew
   Else
      RS.Edit
   End If
```

```
      RS.Fields(0) = RS1.Fields(0)
      RS!SQTY = RS1.Fields(1)
      RS.Update
    RS1.MoveNext
    Loop
End If
If Len(dbcbroker) > 0 Then
    CrystalReport1.SelectionFormula = "{TMPRPT.CODE}='" &
dbcbroker & "'"
Else
    CrystalReport1.SelectionFormula = ""
End If
CrystalReport1.ReportFileName = App.Path &
"\bROWISEDET.RPT"
CrystalReport1.Action = 1
End Sub

Private Sub Command3_Click()
Dim DB As Database
Dim RS, RS1 As Recordset
 If TXTTO = "" Or TXTFROM = "" Then
    MsgBox "ENTER DATE", vbCritical + vbOKOnly
    Exit Sub
 End If
Set DB = OpenDatabase(App.Path & "\ENTRY.MDB")
DB.Execute "DELETE * FROM TMPRPT"

DB.Execute " INSERT INTO TMPRPT " _
      & "SELECT  SCRIPTCODE AS CODE
,SUM(BUYEDQTY*BRATE) AS BQTY,SUM(BUYEDQTY)AS TQTY
FROM TMPTRANSACTION WHERE tDate >= #" &
Format(TXTFROM, "DD/MM/yyYY") & "# AND tDate <= #" &
Format(TXTTO, "DD/MM/yyYY") & "#  GROUP BY SCRIPTCODE"

Set RS1 = DB.OpenRecordset("SELECT
SCRIPTCODE,SUM(SOLDQTY*BRATE),SUM(BUYEDQTY)AS
TQTY FROM TMPTRANSACTION WHERE tDate >= #" &
Format(TXTFROM, "DD/MM/yyyy") & "# AND tDate <= #" &
Format(TXTTO, "DD/MM/yyyy") & "#   GROUP BY
SCRIPTCODE")
Set RS = DB.OpenRecordset("SELECT * FROM TMPRPT")
```

```
If Not (RS1.EOF And RS1.BOF) Then
   RS1.MoveFirst
   Do While Not RS1.EOF
   RS.FindFirst "CODE='" & RS1!SCRIPTCODE & "'"
   If RS.NoMatch = True Then
      RS.AddNew
   Else
      RS.Edit
   End If
      RS.Fields(0) = RS1.Fields(0)
      RS!SQTY = RS1.Fields(1)
      RS.Update
    RS1.MoveNext
    Loop
 End If
 If Len(dbcscript) > 0 Then
    CrystalReport1.SelectionFormula = "{TMPRPT.CODE}='" &
dbcscript & "'"
 Else
    CrystalReport1.SelectionFormula = ""
 End If
 CrystalReport1.ReportFileName = App.Path & "\bROWISE1.RPT"
 CrystalReport1.DiscardSavedData = True
 CrystalReport1.Action = 1

 End Sub

 Private Sub Command4_Click()
 Dim DB As Database
 Dim RS, RS1 As Recordset
  If TXTTO = "" Or TXTFROM = "" Then
    MsgBox "ENTER DATE", vbCritical + vbOKOnly
    Exit Sub
  End If
 Set DB = OpenDatabase(App.Path & "\ENTRY.MDB")
 DB.Execute "DELETE * FROM TMPRPT"

 DB.Execute " INSERT INTO TMPRPT " _
       & "SELECT TDATE AS DATES,COUNT(SOLDQTY) AS
 BQTY,SUM(BUYEDQTY)AS SQTY,SUM(BUYEDQTY*BRATE) AS
 TQTY FROM TMPTRANSACTION WHERE tDate >= #" &
 Format(TXTFROM, "DD/MM/yyYY") & "# AND tDate <= #" &
```

```
Format(TXTTO, "DD/MM/yyYY") & "# GROUP BY TDATE
ORDER BY TDATE"


Set RS = DB.OpenRecordset("SELECT * FROM TMPRPT")




CrystalReport1.ReportFileName = App.Path & "\bROWISE2.RPT"
CrystalReport1.DiscardSavedData = True
CrystalReport1.Action = 1

End Sub


Private Sub Form_Load()
datbroker.DatabaseName = App.Path & "\broker.mdb"
datbroker.RecordSource = "select * from brokermaster"

DATSCRIPT.DatabaseName = App.Path & "\SCRIPT.mdb"
DATSCRIPT.RecordSource = "select * from SCRIPTmaster"
End SUB


Dim dUtl As Double
Dim dBuyedQty As Double
Dim dSoldQty As Double
Dim dBuyedRate As Double

Dim rstrefno As Recordset

Set dbsellBuy = OpenDatabase(App.Path & "\entry.mdb")
Set rsBuyer = dbsellBuy.OpenRecordset("select * from buyer")
```

BUYER

```
Private Sub cmdsave_Click()

Dim dbsellBuy As Database
Dim rsBuyer As Recordset
Dim rsSeller, rstransaction As Recordset
Dim dRemain As Double
Set rstransaction = dbsellBuy.OpenRecordset("select * from
tmptransaction")
dbsellBuy.Execute " INSERT INTO buyer " _
      & "(ScriptCode,ScriptName,qty,rate,bdate,btime,BUYCODE)
VALUES " _
      & "('" & dbcScriptCode & "','" & txtScriptname & "','" &
CDbl(txtqty) & "','" & txtRate & "','" & lblDate & "','" & lbltime &
"','" & gCODE & "');"
          dbsellBuy.Execute "update SELLER set UTL=0 where
ISNULL(utl)"
Set rsSeller = dbsellBuy.OpenRecordset("SELECT * from seller where
SELLCODE <>'" & gCODE & "' AND qty > utl and scriptcode='" &
Trim(dbcScriptCode) & "' and rate <= " & txtRate _
& " order by rate,sdate,stime")
'Set rsSeller = dbsellBuy.OpenRecordset("select * from seller where
scriptcode=" & dbcScriptCode _
 & "and qty > utl and rate <='" & txtRate & "' order by
rate,sdate,stime")"

 Set rsBuyer = dbsellBuy.OpenRecordset("select * from buyer")
 'If Not (rsBuyer.EOF And rsBuyer.BOF) Then
  ' rsBuyer.MoveFirst
  ' End If
   rsBuyer.MoveLast
 If IsNull(rsBuyer!utl) Then
   rsBuyer.Edit
   rsBuyer!utl = 0
   rsBuyer.Update
 End If

 dRemain = rsBuyer!qty - rsBuyer!utl

 If Not (rsSeller.EOF And rsSeller.BOF) Then
```

```
  rsSeller.MoveFirst
End If
Do Until rsSeller.EOF Or dRemain = 0

   If IsNull(rsSeller!utl) Then
   rsSeller.Edit
   rsSeller!utl = 0
   rsSeller.Update
   End If
      rsBuyer.Edit
      rsSeller.Edit

   If rsSeller!qty - rsSeller!utl > rsBuyer!qty - rsBuyer!utl Then
      dBuyedQty = rsBuyer!qty - rsBuyer!utl
      rsBuyer!utl = rsBuyer!utl + rsBuyer!qty - rsBuyer!utl
      rsSeller!utl = rsSeller!utl + dBuyedQty
      dRemain = 0
      'dBuyedQty = rsBuyer!qty - rsBuyer!utl
      dSoldQty = dBuyedQty
      dBuyedRate = rsBuyer!Rate
      dSoldrate = rsSeller!Rate
    Else
      dRemain = dRemain - (rsSeller!qty - rsSeller!utl)
       dBuyedQty = rsSeller!qty - rsSeller!utl
      rsBuyer!utl = rsBuyer!utl + (rsSeller!qty - rsSeller!utl)
       rsSeller!utl = rsSeller!utl + dBuyedQty

       dSoldQty = dBuyedQty
       dBuyedRate = rsBuyer!Rate
      dSoldrate = rsSeller!Rate
    End If
    Set rstrefno = dbsellBuy.OpenRecordset("select max(refno) as ref
from tmptransaction")
    'If IsNull(rstrefno.Fields(0)) Then
      ' rstrefno.Fields(0) = 0
    'End If

      If IsNull(rstrefno.Fields(0)) Then
          updatetmptransaction dbcScriptCode, txtScriptname,
dBuyedQty, dSoldQty, dBuyedRate, dSoldrate, 1, rsBuyer!BUYCODE,
rsSeller!SELLCODE, Date, Time
       Else
```

```
   updatetmptransaction dbcScriptCode, txtScriptname, dBuyedQty,
dSoldQty, dBuyedRate, dSoldrate, rstrefno.Fields(0) + 1,
rsBuyer!BUYCODE, rsSeller!SELLCODE, Date, Time
   End If
   rsBuyer!contract = rsBuyer!contract & "-" & rstrefno.Fields(0) + 1
   rsSeller!contract = rsSeller!contract & "-" & rstrefno.Fields(0) + 1
   rsBuyer.Update
   rsSeller.Update
   rsSeller.MoveNext
 Loop
 dbsellBuy.Execute "update buyer set status='Completed' where
qty=utl"
  dbsellBuy.Execute "update seller set status='Completed' where
qty=utl"
clear
End Sub
Private Sub dbcScriptCode_Click(Area As Integer)
Dim rsscript As Recordset
Dim dbscript As Database
Set dbscript = OpenDatabase(App.Path & "\script.mdb")
If Area = 2 Then
Set rsscript = dbscript.OpenRecordset("select * from scriptmaster
where code='" & dbcScriptCode & "'")
txtScriptname = rsscript!Name
End If
End Sub
Private Sub Form_Load()
datScriptCode.DatabaseName = App.Path & "\script.mdb"
datScriptCode.RecordSource = "select distinct code from scriptmaster"
End Sub
Private Sub txtRate_LostFocus()
lblDate = Date
lbltime = Time
End Sub
Public Sub clear()
txtqty = ""
txtRate = ""
txtScriptname = ""
lblDate = ""
lbltime = ""
dbcScriptCode = ""
End Sub
```

SELLER

```
Private Sub cmdsave_Click()

Dim dbsellBuy As Database
Dim rsBuyer As Recordset
Dim rsSeller, rstransaction As Recordset
Dim dRemain As Double
Dim dUtl As Double
Dim dBuyedQty As Double
Dim dSoldQty As Double
Dim dBuyedRate As Double

Dim rstrefno As Recordset

Set dbsellBuy = OpenDatabase(App.Path & "\entry.mdb")
Set rsSeller = dbsellBuy.OpenRecordset("select * from seller")
Set rstransaction = dbsellBuy.OpenRecordset("select * from
tmptransaction")



dbsellBuy.Execute " INSERT INTO seller " _
      & "(ScriptCode,ScriptName,qty,rate,sdate,stime,SELLCODE)
VALUES " _
      & "('" & dbcScriptCode & "','" & txtScriptname & "','" &
CDbl(txtqty) & "','" & txtRate & "','" & lblDate & "','" & lbltime &
"','" & gCODE & "');"



dbsellBuy.Execute "update buyer set UTL=0 where ISNULL(utl)"

Set rsBuyer = dbsellBuy.OpenRecordset("SELECT * from buyer where
BUYCODE <>'" & gCODE & "' AND qty > utl and scriptcode='" &
Trim(dbcScriptCode) & "' and rate >= " & txtRate _
& " order by rate Desc,bdate,btime")


'Set rsSeller = dbsellBuy.OpenRecordset("select * from seller where
scriptcode=" & dbcScriptCode _
```

```
& "and qty > utl and rate <='" & txtRate & "' order by
rate,sdate,stime")"

Set rsSeller = dbsellBuy.OpenRecordset("select * from seller")
'If Not (rsBuyer.EOF And rsBuyer.BOF) Then
 '  rsBuyer.MoveFirst
  ' End If
   rsSeller.MoveLast
If IsNull(rsSeller!utl) Then
   rsSeller.Edit
   rsSeller!utl = 0
   rsSeller.Update
End If

dRemain = rsSeller!qty - rsSeller!utl

If Not (rsBuyer.EOF And rsBuyer.BOF) Then
   rsBuyer.MoveFirst
End If
Do Until rsBuyer.EOF Or dRemain = 0

   If IsNull(rsBuyer!utl) Then
   rsBuyer.Edit
   rsBuyer!utl = 0
   rsBuyer.Update
   End If
      rsSeller.Edit
      rsBuyer.Edit

   If rsBuyer!qty - rsBuyer!utl >= rsSeller!qty - rsSeller!utl Then
      dSoldQty = rsSeller!qty - rsSeller!utl
      rsBuyer!utl = rsBuyer!utl + rsSeller!qty - rsSeller!utl
      rsSeller!utl = rsSeller!utl + dSoldQty
      dRemain = 0
      'dBuyedQty = rsBuyer!qty - rsBuyer!utl
      dBuyedQty = dSoldQty
      dBuyedRate = rsBuyer!Rate
      dSoldrate = rsSeller!Rate
    Else
      dRemain = dRemain - (rsBuyer!qty - rsBuyer!utl)
       dSoldQty = rsBuyer!qty - rsBuyer!utl
      rsSeller!utl = rsSeller!utl + (rsBuyer!qty - rsBuyer!utl)
       rsBuyer!utl = rsBuyer!utl + dSoldQty
```

```
      dBuyedQty = dSoldQty
      dBuyedRate = rsBuyer!Rate
     dSoldrate = rsSeller!Rate
  End If
  Set rstrefno = dbsellBuy.OpenRecordset("select max(refno) as ref
from tmptransaction")
  If IsNull(rstrefno.Fields(0)) Then
     rstrefno.Fields(0) = 0
  End If



  updatetmptransaction dbcScriptCode, txtScriptname, dBuyedQty,
dSoldQty, dBuyedRate, dSoldrate, rstrefno.Fields(0) + 1,
rsBuyer!BUYCODE, rsSeller!SELLCODE, Date, Time

  rsBuyer!contract = rsBuyer!contract & "-" & rstrefno.Fields(0) + 1
  rsSeller!contract = rsSeller!contract & "-" & rstrefno.Fields(0) + 1
  rsBuyer.Update
  rsSeller.Update
  rsBuyer.MoveNext
 Loop

 dbsellBuy.Execute "update buyer set status='Completed' where
qty=utl"
  dbsellBuy.Execute "update seller set status='Completed' where
qty=utl"
 clear


 End Sub



 Private Sub dbcScriptCode_Click(Area As Integer)
 Dim rsscript As Recordset
 Dim dbscript As Database
 Set dbscript = OpenDatabase(App.Path & "\script.mdb")
 If Area = 2 Then
 Set rsscript = dbscript.OpenRecordset("select * from scriptmaster
 where code='" & dbcScriptCode & "'")
 txtScriptname = rsscript!Name
```

```
End If
End Sub

Private Sub Form_Load()
datScriptCode.DatabaseName = App.Path & "\script.mdb"
datScriptCode.RecordSource = "select distinct code from scriptmaster"
End Sub

Private Sub txtRate_LostFocus()
lblDate = Date
lbltime = Time
End Sub

Public Sub clear()
txtqty = ""
txtRate = ""
txtScriptname = ""
lblDate = ""
lbltime = ""
dbcScriptCode = ""
End Sub
```

ENTRY

```
Private Sub Form_Load()
DATENTRY.DatabaseName = App.Path & "\ENTRY.MDB"
DATENTRY.RecordSource = "SELECT * FROM ENTRY WHERE
CODE='" & gCODE & "'ORDER BY SNO"
DATSCRIPT.DatabaseName = App.Path & "\SCRIPT.MDB"
DATSCRIPT.RecordSource = "SELECT * FROM SCRIPTMASTER "
End Sub

Private Sub LSTTYPE_Click()
DBGENTRY.Columns(4).Text = LSTTYPE
LSTTYPE.Visible = False
End Sub

Public Sub WillingToSell()
Dim REMAIN As Double
Dim CON As Double
Dim RSBUY As Recordset



On Error GoTo AA


Set DBCTX = OpenDatabase(App.Path & "\ENTRY.MDB")


'If DBGENTRY.Columns(4).Text = "SELL" Then
Set RSBUY = DBCTX.OpenRecordset("SELECT * FROM ENTRY
WHERE TRNTYPE <> '" & DBGENTRY.Columns(4).Text & "'AND
SCRIPTNAME = '" & DBGENTRY.Columns(3).Text & "'AND RATE
>='" & DBGENTRY.Columns(6).Text & "'AND NAME <> '" &
gNAME & "'AND (STATUS =NULL OR STATUS = 'NOT
COMPLETED')  ORDER BY RATE,DATE,TIME")

'Set RSBUY = DBCTX.OpenRecordset("SELECT * FROM ENTRY
WHERE TRNTYPE <> '" & DBGENTRY.Columns(4).Text & "'AND
SCRIPTNAME = '" & DBGENTRY.Columns(3).Text & "'AND RATE
<='" & DBGENTRY.Columns(6).Text & "'AND NAME <> '" &
```

```
gNAME & "'AND (STATUS =NULL OR STATUS = 'NOT
COMPLETED') ORDER BY RATE,DATE,TIME")
'End If
If Not (RSBUY.EOF And RSBUY.BOF) Then
  RSBUY.MoveFirst
  'RSBUY.Edit
End If
Do While Not RSBUY.EOF
   RSBUY.Edit
   If IsNull(RSBUY!BUYED) Then
     RSBUY!BUYED = 0
   End If
   If IsNull(RSBUY!bALANCE) Then
     RSBUY!bALANCE = 0
   End If

If Val(RSBUY!qty) - Val(RSBUY!BUYED) <
Val(DBGENTRY.Columns(5).Text) Then
RSBUY!BUYED = Val(RSBUY!BUYED) + Val(RSBUY!qty) -
Val(RSBUY!BUYED)
DBGENTRY.Columns(12).Text = Val(DBGENTRY.Columns(5).Text) -
Val(RSBUY!BUYED) + Val(RSBUY!bALANCE)
RSBUY!bALANCE = RSBUY!qty - Val(RSBUY!BUYED)
REMAIN = DBGENTRY.Columns(12).Text
GoTo ab
Else

If RSBUY!BUYED <> "" Then
  RSBUY!BUYED = Val(RSBUY!BUYED) +
Val(DBGENTRY.Columns(5).Text)
  If RSBUY!BUYED > RSBUY!qty Then
     REMAIN = RSBUY!BUYED - RSBUY!qty
     RSBUY!BUYED = RSBUY!qty

   End If

Else
DBGENTRY.Columns(11).Text = DBGENTRY.Columns(5).Text

   RSBUY!BUYED = DBGENTRY.Columns(11).Text
    If Val(RSBUY!BUYED) > Val(RSBUY!qty) Then
      REMAIN = RSBUY!BUYED - RSBUY!qty
      RSBUY!BUYED = RSBUY!qty
```

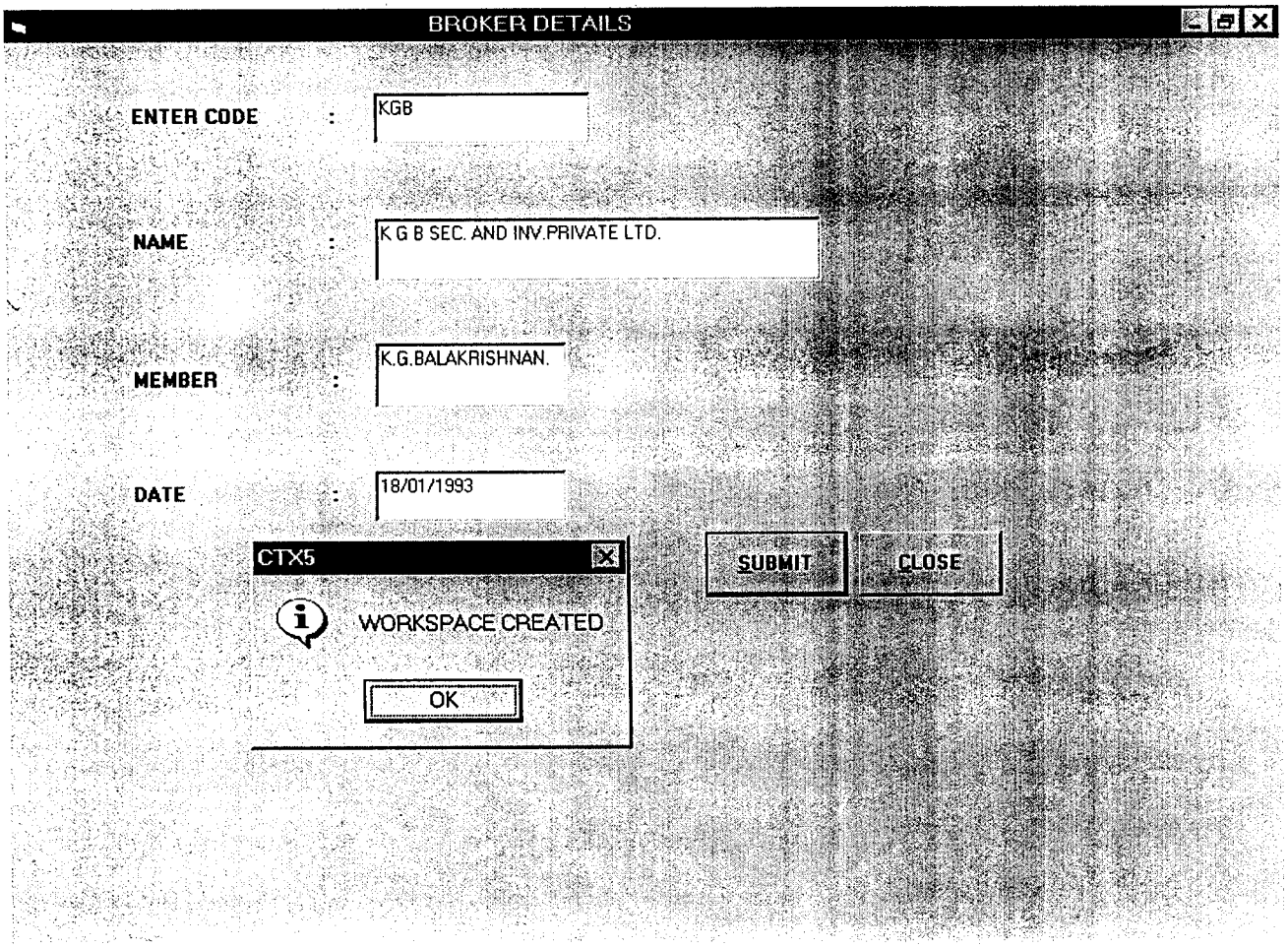
```
  End If
End If
RSBUY!bALANCE = RSBUY!qty - Val(RSBUY!BUYED)
End If

If REMAIN <> 0 Then
DBGENTRY.Columns(12).Text = REMAIN
DBGENTRY.Columns(11).Text = Val(DBGENTRY.Columns(11).Text)
+ Val(DBGENTRY.Columns(5).Text) - REMAIN
Else
DBGENTRY.Columns(12).Text = Val(DBGENTRY.Columns(5).Text) -
Val(DBGENTRY.Columns(11).Text)
End If
If DBGENTRY.Columns(12).Text < 0 Then
DBGENTRY.Columns(12).Text = 0
End If
ab:
If DBGENTRY.Columns(12).Text <> 0 Then
DBGENTRY.Columns(9).Text = "NOT COMPLETED"
Else
DBGENTRY.Columns(9).Text = "COMPLETED"
End If
If RSBUY!bALANCE <> "0" Then
   RSBUY!Status = "NOT COMPLETED"
Else
   RSBUY!Status = "COMPLETED"
End If
RSBUY!REF = DBGENTRY.Columns(0)
RSBUY!AQTY = DBGENTRY.Columns(11)
RSBUY!REFNO = 1
RSBUY.Update


REMAIN = 0
RSBUY.MoveNext
Loop


Exit Sub
AA:
   Exit Sub

End Sub
```

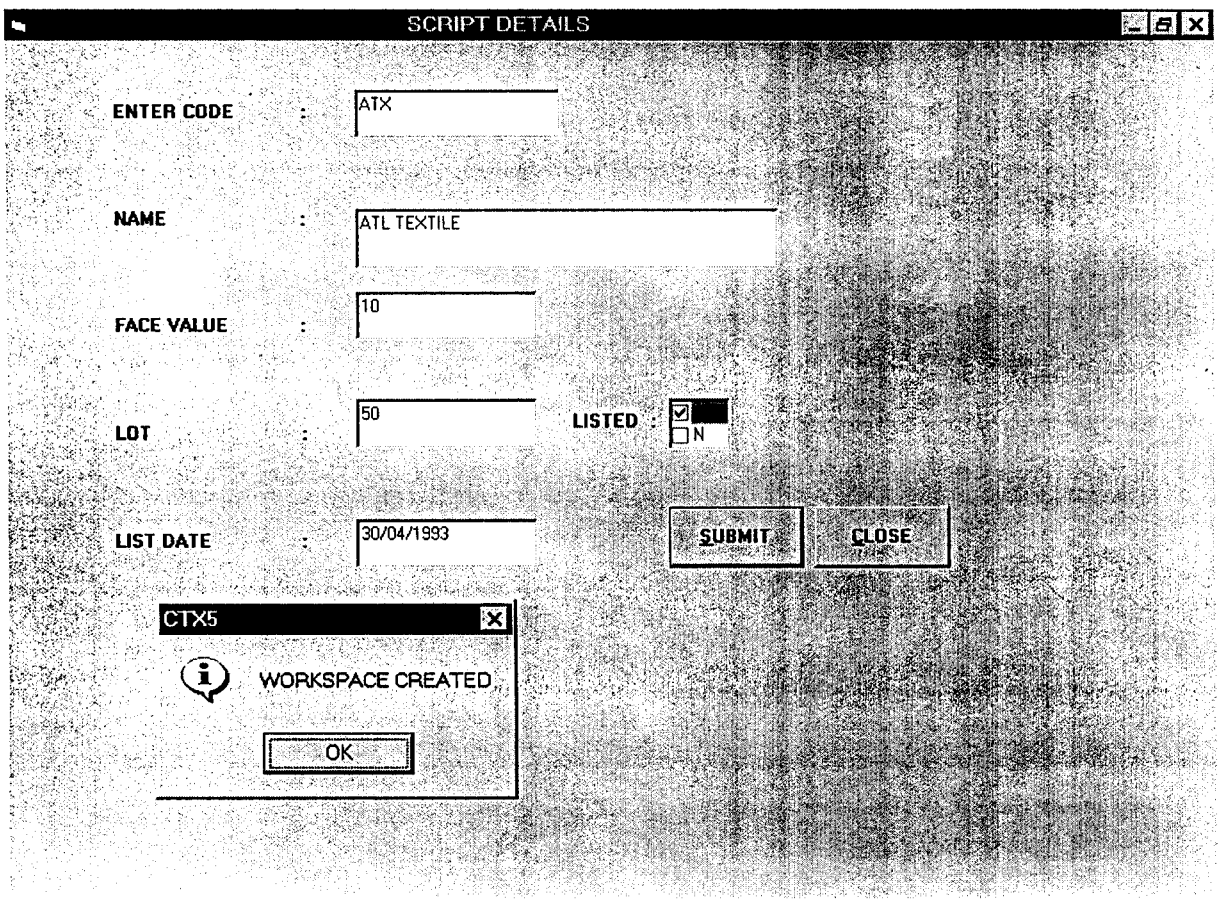
GRAPH

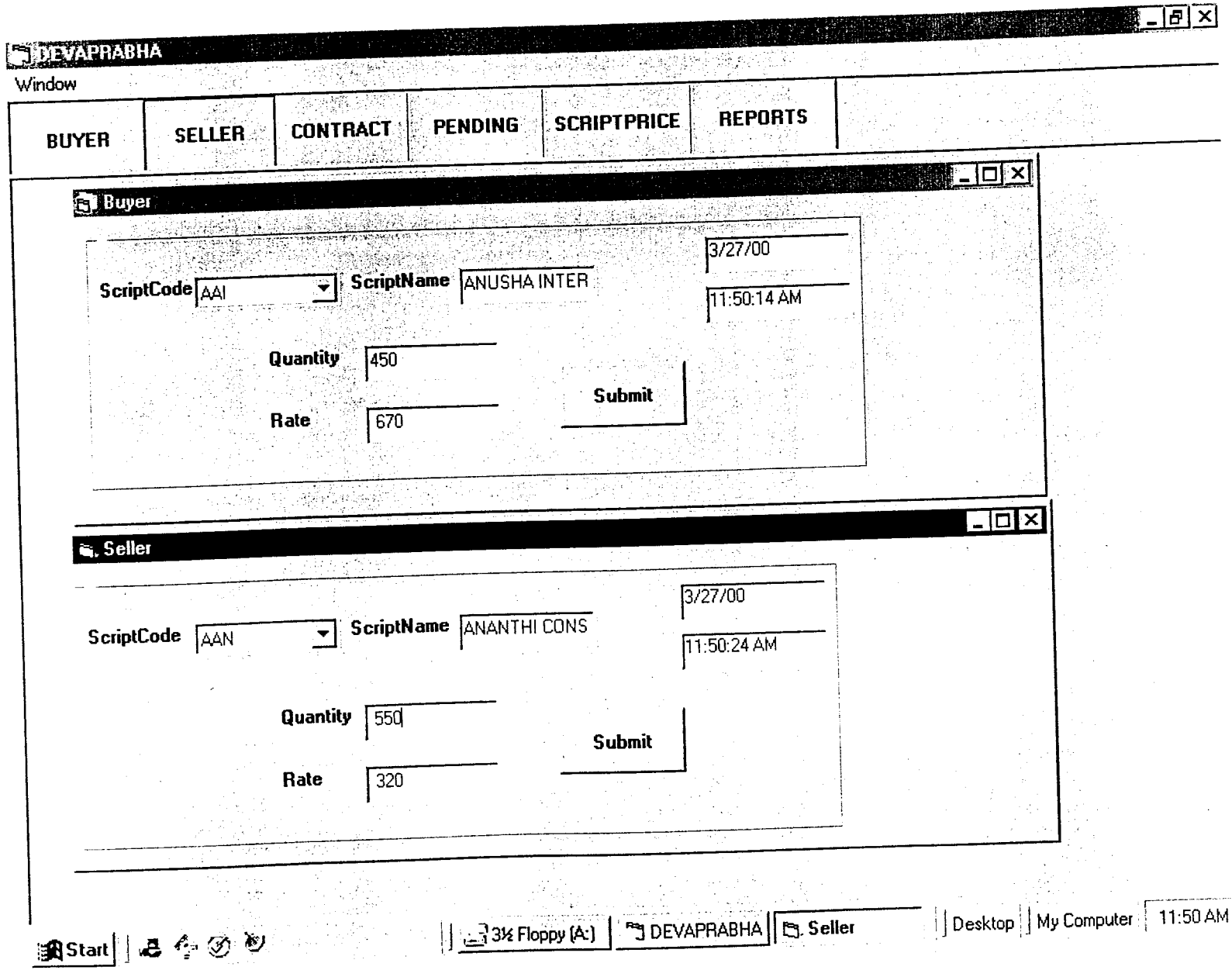
```
Option Explicit

Private Sub dbcScriptCode_Click(Area As Integer)
Dim rsGraph As Recordset
Dim dbGraph As Database
Dim dOpen As Double
Dim dHigh As Double
Dim dLow As Double
Dim Odate As Date
Dim dClose As Double
Dim rsscript As Recordset
Dim dbscript As Database
Set dbscript = OpenDatabase(App.Path & "\script.mdb")
Set dbGraph = OpenDatabase(App.Path & "\ENTRY.mdb")

dbGraph.Execute "DELETE FROM GRAPH"
dbGraph.Execute "DELETE FROM DRAWGRAPH"

If Area = 2 Then

      dbGraph.Execute " INSERT INTO DRAWGRAPH " _
      & "SELECT tTIME,BRATE " _
      & "FROM TMPTRANSACTION WHERE tdate=#" &
Format(gDATE, "mm/dd/yyyy") & "#  AND SCRIPTCODE='" &
dbcScriptCode & "'ORDER BY TTIME;"



   Set rsscript = dbscript.OpenRecordset("select * from scriptmaster
where code='" & dbcScriptCode & "'")
   txtScriptname = rsscript!Name

   Odate = DateAdd("D", -1, gDATE)


   Set rsGraph = dbGraph.OpenRecordset("select * from
tmptransaction where tdate=#" & Format(Odate, "mm/dd/yyyy") & "#
AND scriptcode='" & dbcScriptCode & "'")
    If rsGraph.RecordCount = 0 Then GoTo AA
   rsGraph.MoveLast
```

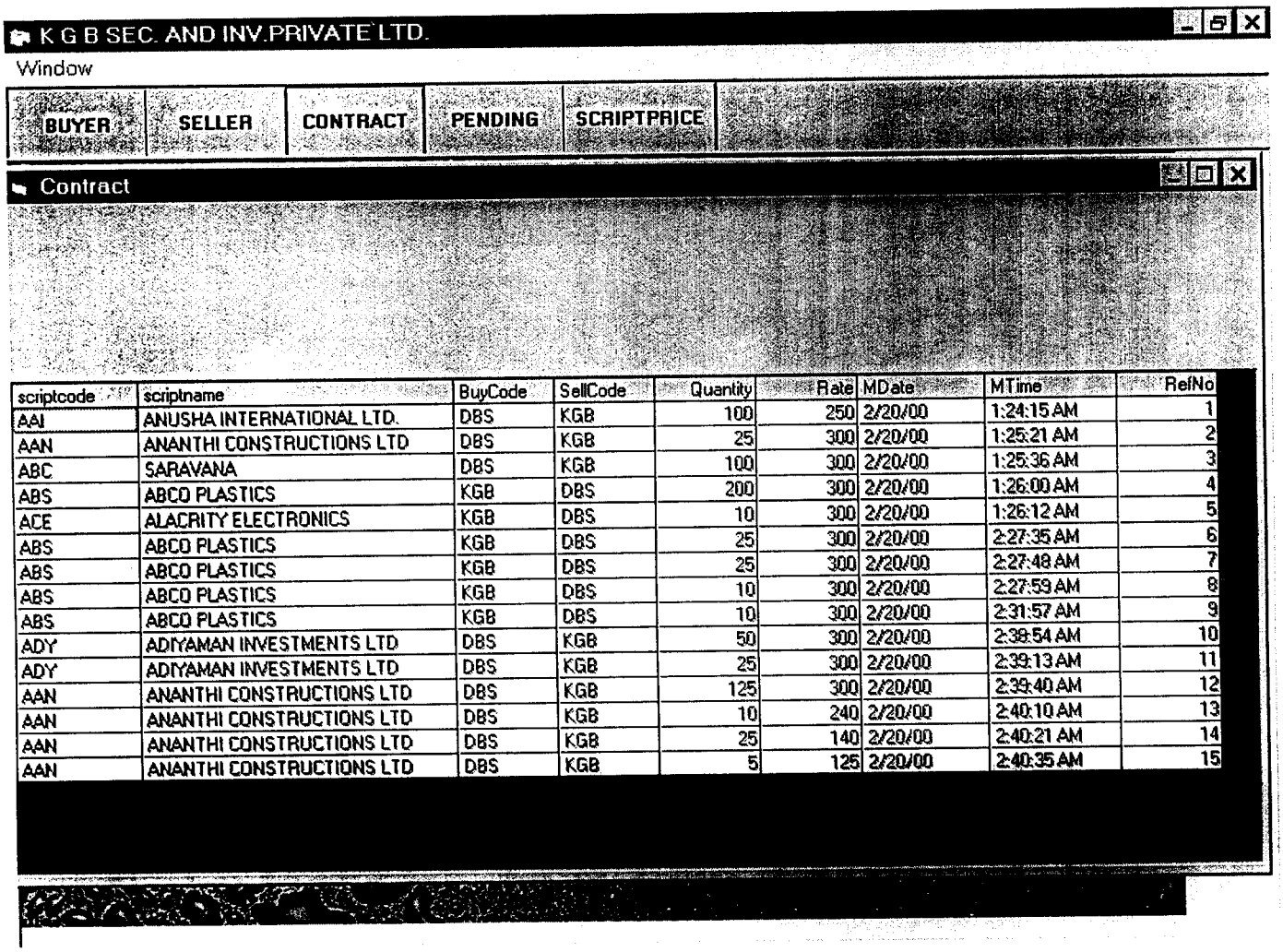
```
   dOpen = rsGraph!BRATE
AA:
      Set rsGraph = dbGraph.OpenRecordset("select * from
tmptransaction where tdate=#" & Format(gDATE, "mm/dd/yyyy") &
"# AND scriptcode='" & dbcScriptCode & "'order by BRate")
      If Not (rsGraph.EOF And rsGraph.BOF) Then
          rsGraph.MoveFirst
          dLow = rsGraph!BRATE
      End If
    If rsGraph.RecordCount = 0 Then GoTo CC
   rsGraph.MoveLast
   dHigh = rsGraph!BRATE
CC:
   Set rsGraph = dbGraph.OpenRecordset("select * from
tmptransaction where tdate=#" & Format(gDATE, "mm/dd/yyyy") &
"# AND scriptcode='" & dbcScriptCode & "'")
    If rsGraph.RecordCount = 0 Then GoTo DD
   rsGraph.MoveLast
   dClose = rsGraph!BRATE
  DD:
   dbGraph.Execute " INSERT INTO GRAPH " _
      & "(OPENPRICE,HIGH,LOW,CLOSEPRICE) VALUES " _
      & "('" & dOpen & "','" & dHigh & "','" & dLow & "','" & dClose
& "');"
    datgraph.DatabaseName = App.Path & "\ENTRY.MDB"
    datgraph.RecordSource = "SELECT * FROM GRAPH"
    datgraph.Refresh
   msGraph.Visible = True

  End If

   Exit Sub
End Sub

Private Sub Form_Load()
datScriptCode.DatabaseName = App.Path & "\script.mdb"
datScriptCode.RecordSource = "select distinct code from scriptmaster"
End Sub
```

SCREEN LAYOUTS

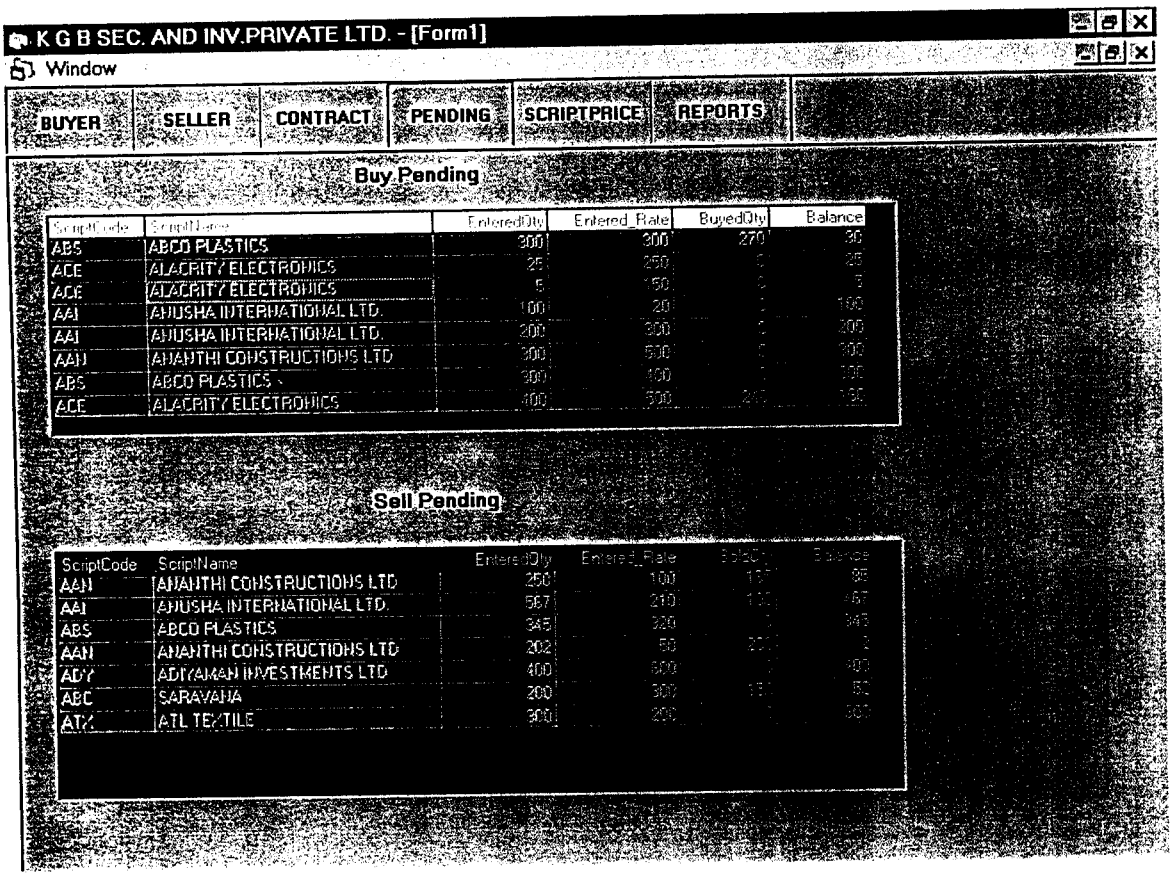

BROKER DETAILS
ENTER CODE :
KGB
NAME :
K G B SEC. AND INV.PRIVATE LTD.
MEMBER :
K.G.BALAKRISHNAN.
DATE :
18/01/1993
CTX5
WORKSPACE CREATED
OK
SUBMIT
CLOSE

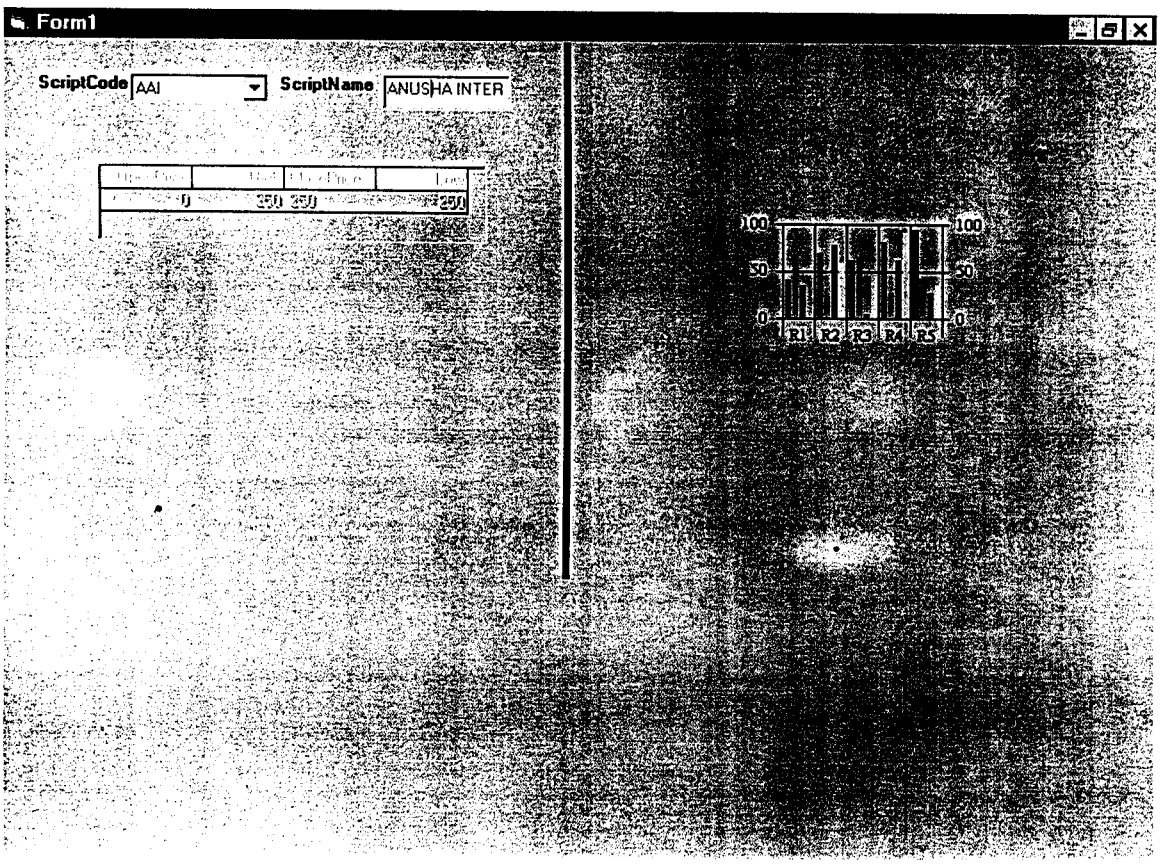

SCRIPT DETAILS
ENTER CODE :
ATX
NAME :
ATL TEXTILE
FACE VALUE :
10
LOT :
50
LISTED :
N
LIST DATE :
30/04/1993
SUBMIT
CLOSE
CTX5
WORKSPACE CREATED
OK

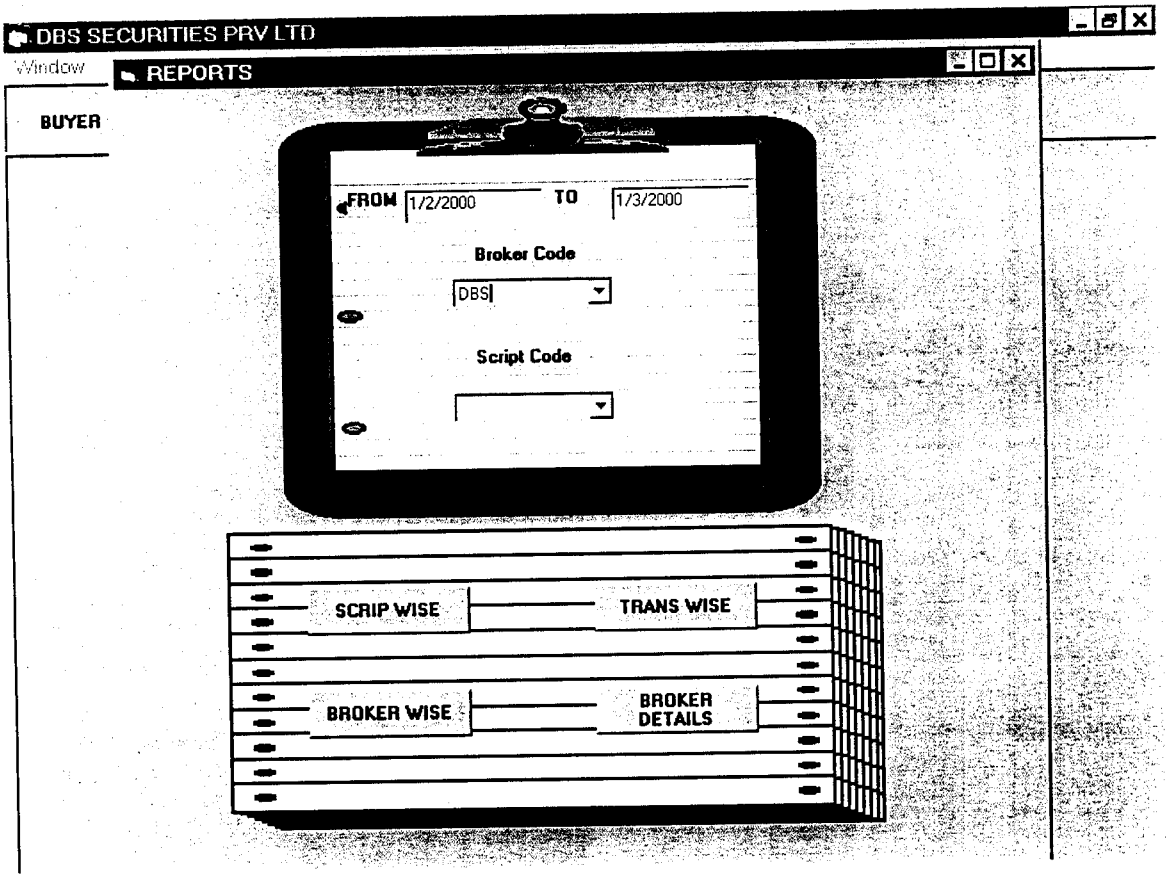

DEVAPRABHA
Window
BUYER
SELLER
CONTRACT
PENDING
SCRIPTPRICE
REPORTS
Buyer
ScriptCode
AAI
ScriptName
ANUSHA INTER
3/27/00
11:50:14 AM
Quantity
450
Rate
670
Submit
Seller
ScriptCode
AAN
ScriptName
ANANTHI CONS
3/27/00
11:50:24 AM
Quantity
550
Rate
320
Submit
Start
3½ Floppy (A:)
DEVAPRABHA
Seller
Desktop
My Computer
11:50 AM

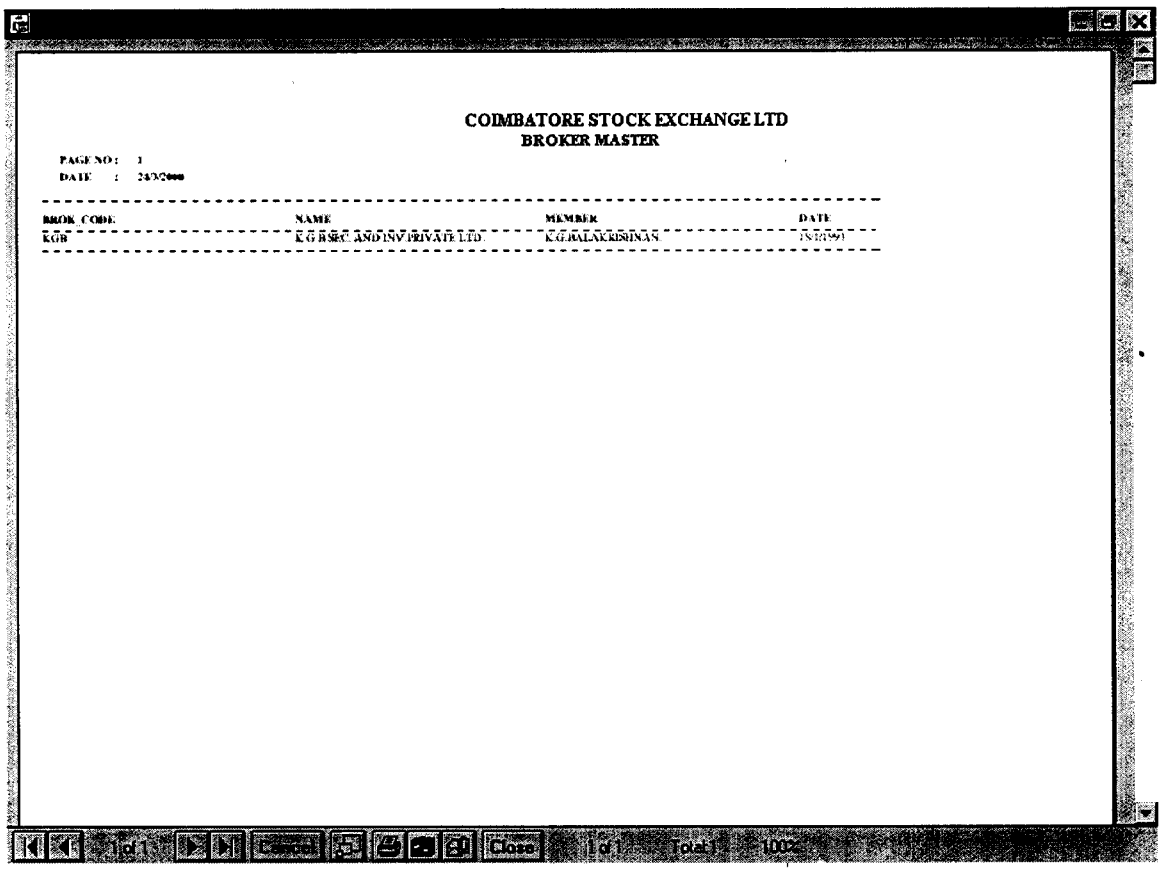


scriptcode	scriptname	BuyCode	SellCode	Quantity	Rate	MDate	MTime	RefNo
AAI	ANUSHA INTERNATIONAL LTD.	DBS	KGB	100	250	2/20/00	1:24:15 AM	1
AAN	ANANTHI CONSTRUCTIONS LTD	DBS	KGB	25	300	2/20/00	1:25:21 AM	2
ABC	SARAVANA	DBS	KGB	100	300	2/20/00	1:25:36 AM	3
ABS	ABCO PLASTICS	KGB	DBS	200	300	2/20/00	1:26:00 AM	4
ACE	ALACRITY ELECTRONICS	KGB	DBS	10	300	2/20/00	1:26:12 AM	5
ABS	ABCO PLASTICS	KGB	DBS	25	300	2/20/00	2:27:35 AM	6
ABS	ABCO PLASTICS	KGB	DBS	25	300	2/20/00	2:27:48 AM	7
ABS	ABCO PLASTICS	KGB	DBS	10	300	2/20/00	2:27:59 AM	8
ABS	ABCO PLASTICS	KGB	DBS	10	300	2/20/00	2:31:57 AM	9
ADY	ADIYAMAN INVESTMENTS LTD	DBS	KGB	50	300	2/20/00	2:38:54 AM	10
ADY	ADIYAMAN INVESTMENTS LTD	DBS	KGB	25	300	2/20/00	2:39:13 AM	11
AAN	ANANTHI CONSTRUCTIONS LTD	DBS	KGB	125	300	2/20/00	2:39:40 AM	12
AAN	ANANTHI CONSTRUCTIONS LTD	DBS	KGB	10	240	2/20/00	2:40:10 AM	13
AAN	ANANTHI CONSTRUCTIONS LTD	DBS	KGB	25	140	2/20/00	2:40:21 AM	14
AAN	ANANTHI CONSTRUCTIONS LTD	DBS	KGB	5	125	2/20/00	2:40:35 AM	15


K G B SEC. AND INV.PRIVATE LTD. - [Form1]
Window
BUYER
SELLER
CONTRACT
PENDING
SCRIPTPRICE
REPORTS
Buy Pending
ScriptCode
ScriptName
EnteredQty
Entered_Rate
BuyedQty
Balance
ABS ABCO PLASTICS 300 300 270 30
ACE ALACRITY ELECTRONICS 25 250 2 23
ACE ALACRITY ELECTRONICS 5 150 2 3
AAI ANUSHA INTERNATIONAL LTD. 100 20 1 100
AAI ANUSHA INTERNATIONAL LTD. 200 300 2 200
AAN ANANTHI CONSTRUCTIONS LTD 300 500 1 300
ABS ABCO PLASTICS 300 400 1 100
ACE ALACRITY ELECTRONICS 400 300 240 160
Sell Pending
ScriptCode
ScriptName
EnteredQty
Entered_Rate
AAN ANANTHI CONSTRUCTIONS LTD 250 100
AAI ANUSHA INTERNATIONAL LTD. 987 210
ABS ABCO PLASTICS 345 320
AAN ANANTHI CONSTRUCTIONS LTD 202 50
ADY ADITYAMAN INVESTMENTS LTD 400 500
ABC SARAVANA 200 300
ATL ATL TEXTILE 300 200


Form1
ScriptCode
AAI
ScriptName
ANUSHA INTER
100
50
0
R1
R2
R3
R4
R5


DBS SECURITIES PRV LTD
Window
REPORTS
BUYER
FROM 1/2/2000
TO 1/3/2000
Broker Code
DBS
Script Code
SCRIP WISE
TRANS WISE
BROKER WISE
BROKER DETAILS

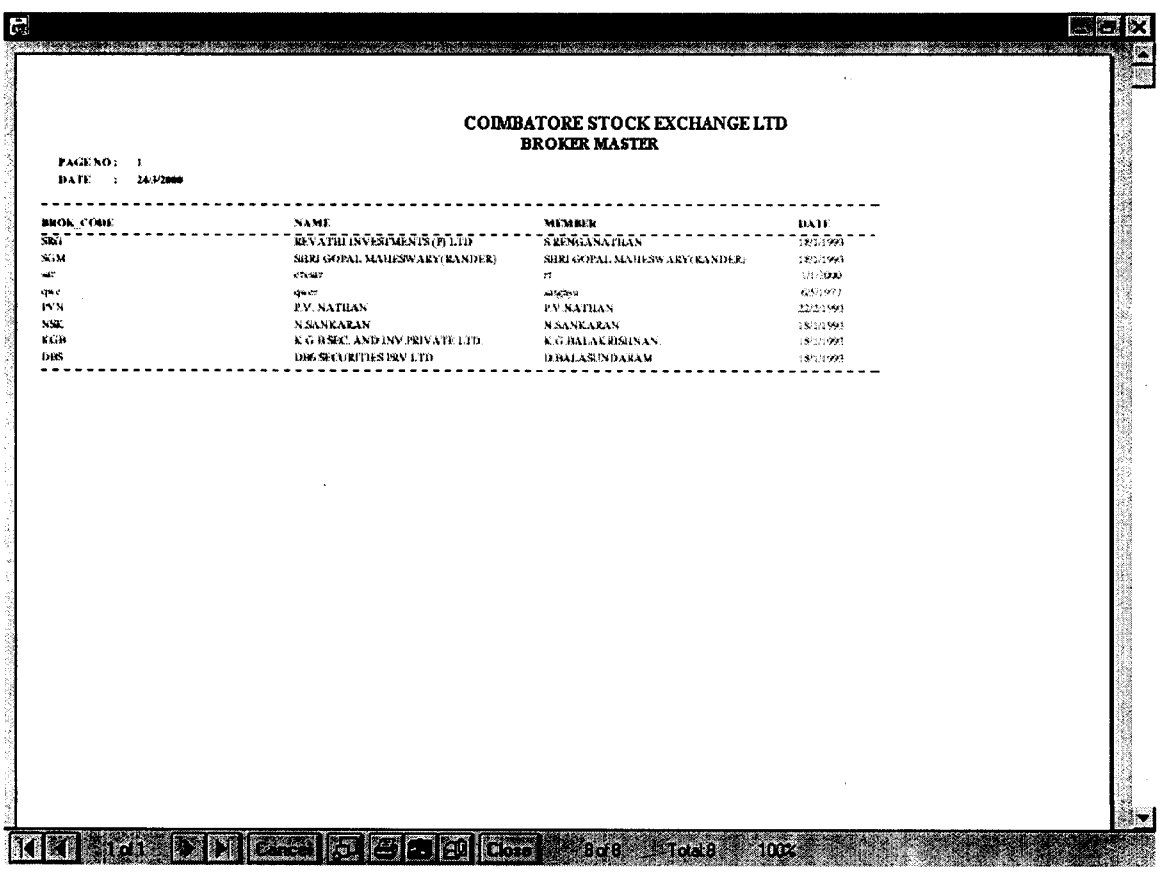


COIMBATORE STOCK EXCHANGE LTD
BROKER MASTER

PAGE NO : 1
DATE : 24/2/000

BROK CODE	NAME	MEMBER	DATE
KGB	K G BARC AND INV PRIVATE LTD	K.G.BALAKRISHNAN	[illegible]

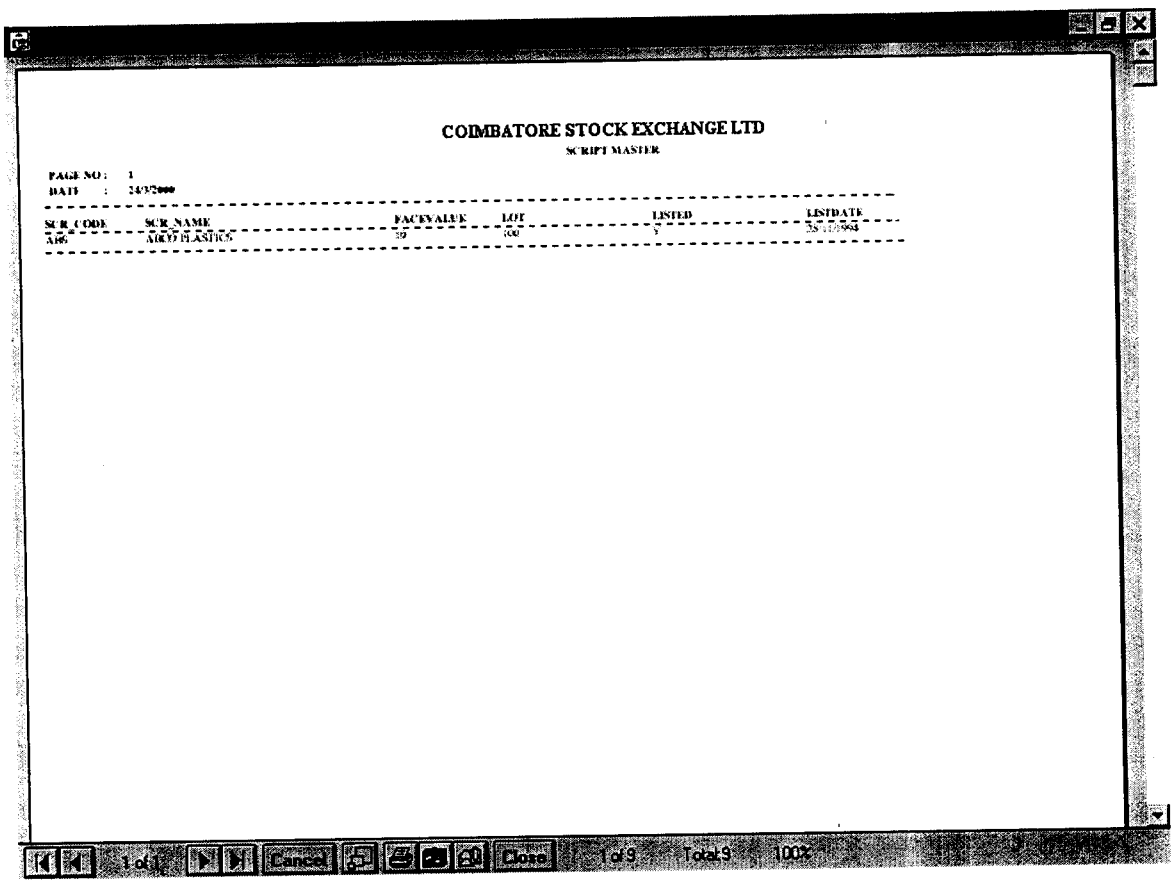
Cancel Close



COIMBATORE STOCK EXCHANGE LTD
BROKER MASTER

PAGE NO : 1
DATE : 24/3/2000

BROK_CODE	NAME	MEMBER	DATE
SRI	SRIVATHI INVESTMENTS (P) LTD	S.RENGANATHAN	18/3/1993
SGM	SHRI GOPAL MAHESWARY(RANDER)	SHRI GOPAL MAHESWARY(RANDER)	18/3/1993
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
PVN	P.V. NATHAN	P.V.NATHAN	22/3/1993
NSK	N.SANKARAN	N.SANKARAN	18/3/1993
KGB	K.G.B.SEC. AND INV.PRIVATE LTD.	K.G.BALAKRISHNAN.	18/3/1993
DBS	DBS SECURITIES INV.LTD	D.BALASUNDARAM	18/3/1993



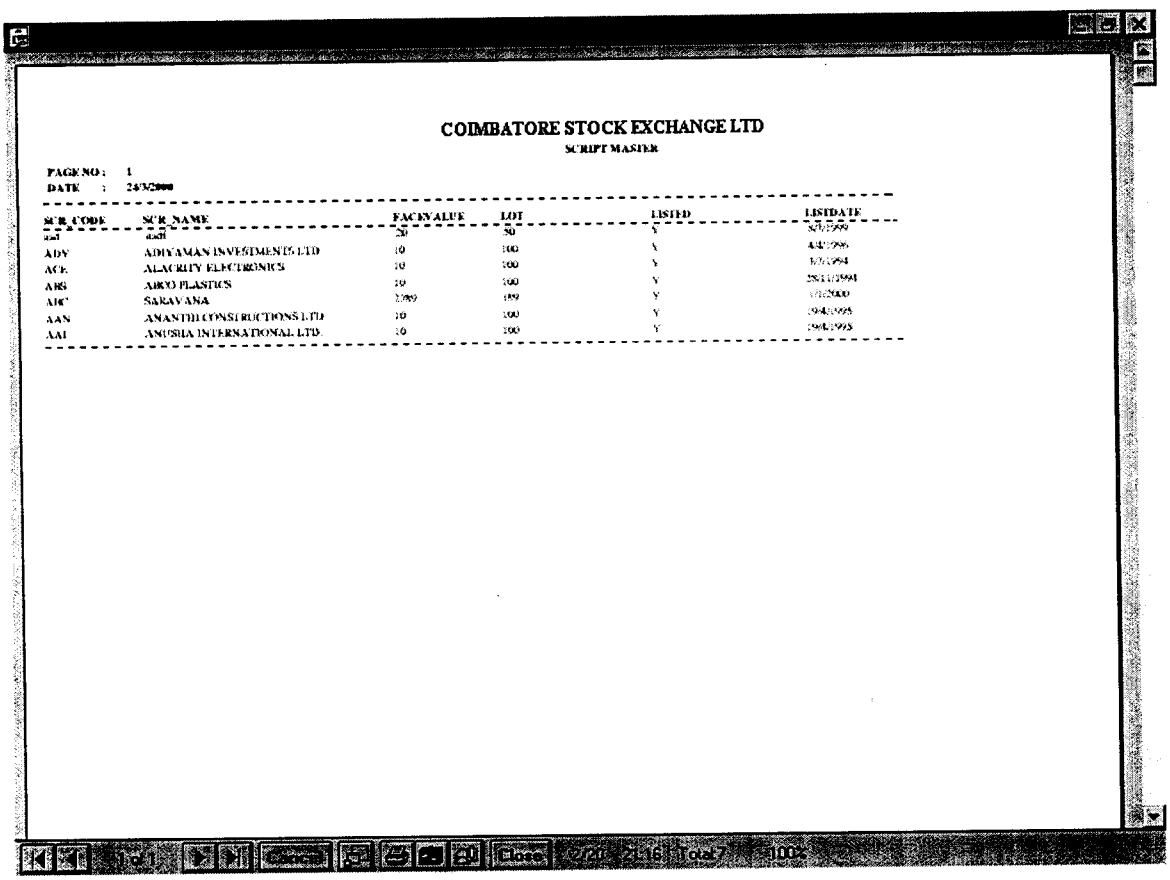
COIMBATORE STOCK EXCHANGE LTD

SCRIPT MASTER

PAGE NO : 1
DATE : 24/3/2000

SCR_CODE	SCR_NAME	FACEVALUE	LOT	LISTED	LISTDATE
AI06	AIKYO PLASTICS	10	100	Y	28/11/1994

1 of 1 Cancel Close 1 of 9 Total 9 100%

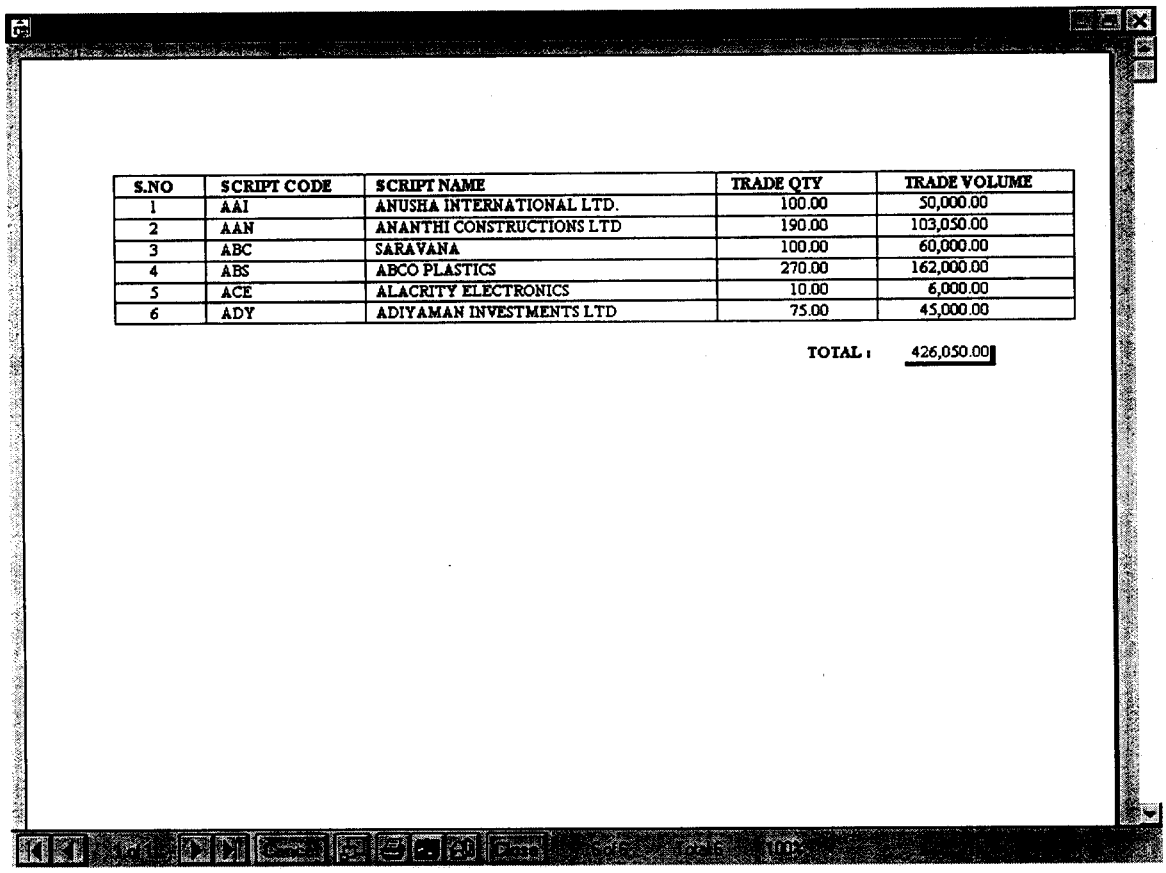


COIMBATORE STOCK EXCHANGE LTD

SCRIPT MASTER

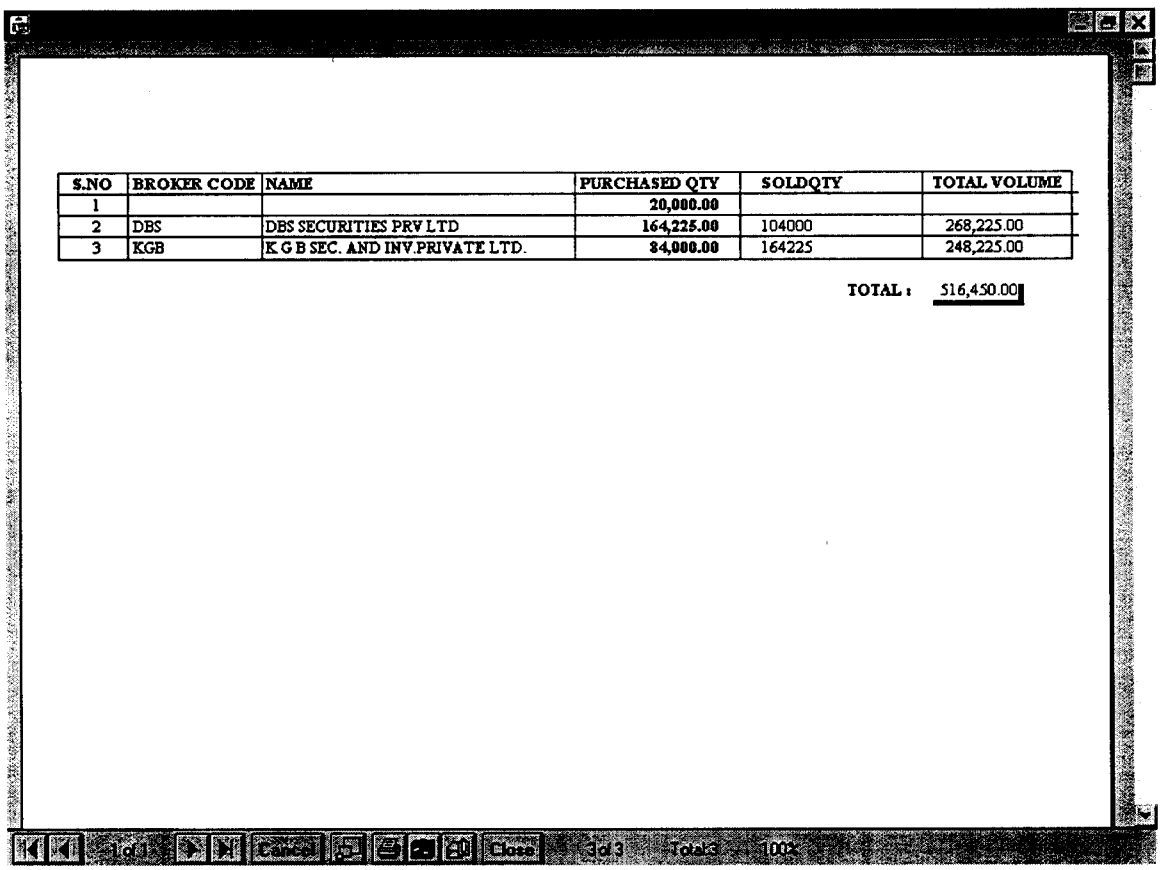
PAGE NO : 1
DATE : 24/5/2000

SCR_CODE	SCR_NAME	FACEVALUE	LOT	LISTED	LISTDATE
[illegible]	[illegible]	20	50	Y	[illegible]
ADV	ADITYAMAN INVESTMENTS LTD	10	100	Y	[illegible]
ACE	ALACRITY ELECTRONICS	10	100	Y	[illegible]
ARS	ARCO PLASTICS	10	100	Y	[illegible]
ARC	SARAVANA	[illegible]	[illegible]	Y	[illegible]
AAN	ANANTHI CONSTRUCTIONS LTD	10	100	Y	[illegible]
AAI	ANUSHA INTERNATIONAL LTD	10	100	Y	[illegible]



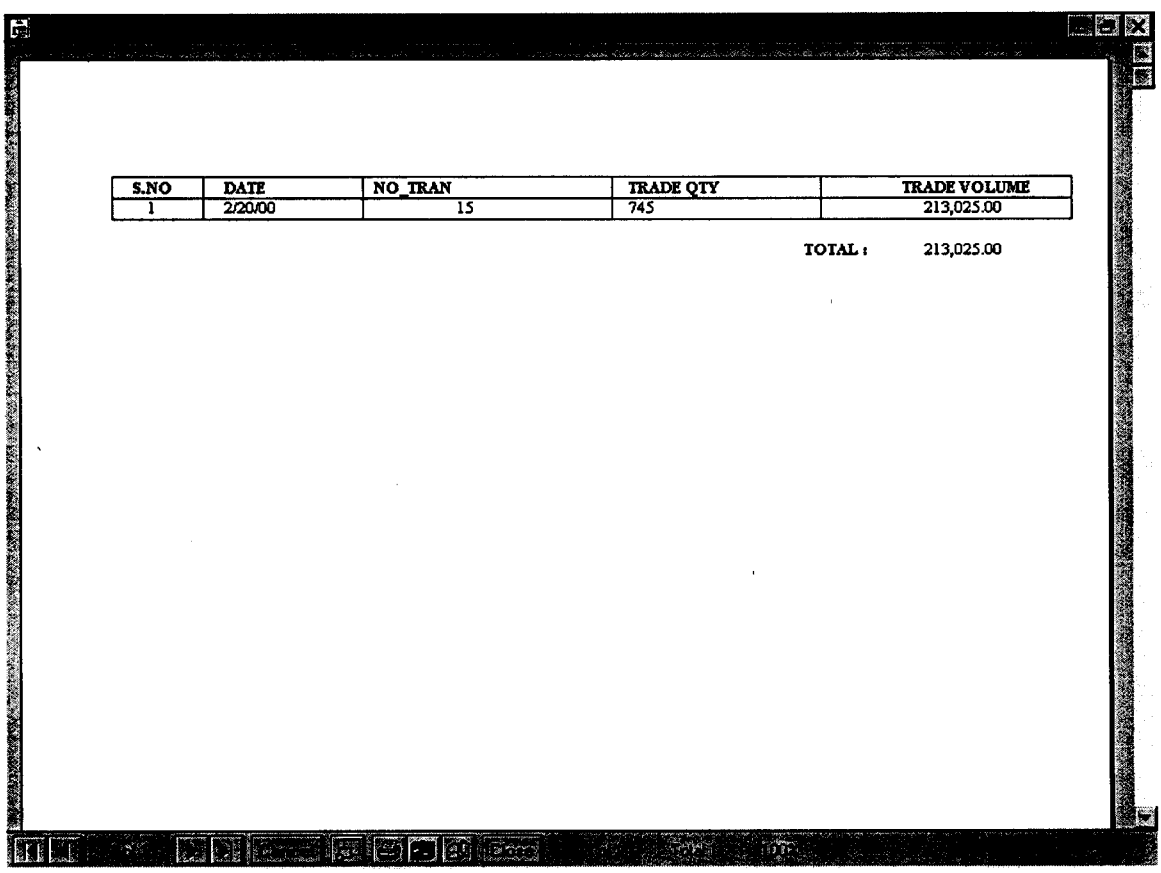
S.NO	SCRIPT CODE	SCRIPT NAME	TRADE QTY	TRADE VOLUME
1	AAI	ANUSHA INTERNATIONAL LTD.	100.00	50,000.00
2	AAN	ANANTHI CONSTRUCTIONS LTD	190.00	103,050.00
3	ABC	SARAVANA	100.00	60,000.00
4	ABS	ABCO PLASTICS	270.00	162,000.00
5	ACE	ALACRITY ELECTRONICS	10.00	6,000.00
6	ADY	ADIYAMAN INVESTMENTS LTD	75.00	45,000.00

TOTAL : 426,050.00



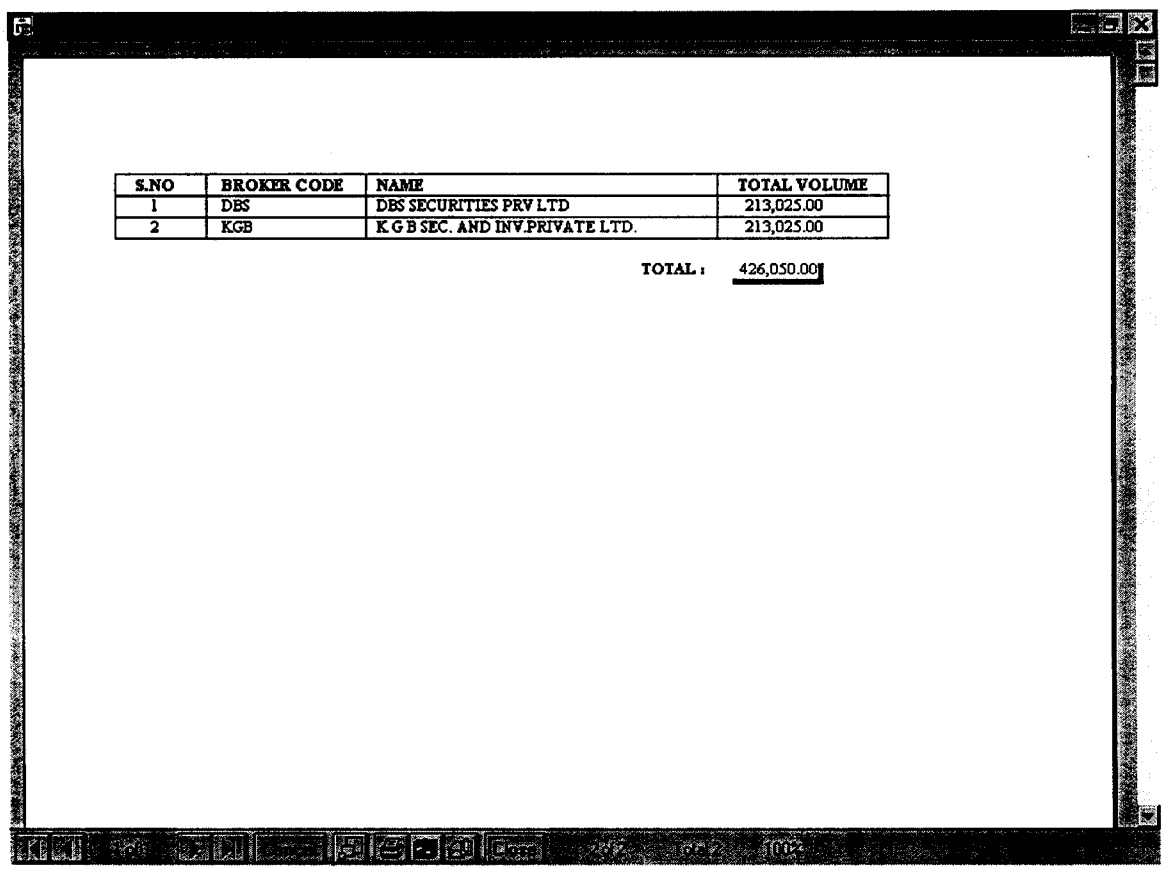
S.NO	BROKER CODE	NAME	PURCHASED QTY	SOLDQTY	TOTAL VOLUME
1			**20,000.00**		
2	DBS	DBS SECURITIES PRV LTD	**164,225.00**	104000	268,225.00
3	KGB	K G B SEC. AND INV.PRIVATE LTD.	**84,000.00**	164225	248,225.00

TOTAL : 516,450.00



S.NO	DATE	NO_TRAN	TRADE QTY	TRADE VOLUME
1	2/20/00	15	745	213,025.00

TOTAL : 213,025.00



S.NO	BROKER CODE	NAME	TOTAL VOLUME
1	DBS	DBS SECURITIES PRV LTD	213,025.00
2	KGB	K G B SEC. AND INV.PRIVATE LTD.	213,025.00

TOTAL : 426,050.00

IMPLEMENTATION

IMPLEMENTATION

After the system analysis and design phase completion, coding for the project was carried out. This was followed by the implementation phase.

The implementation phase is an important period where the project is tested for its validity and correctness at real time environment. There are three types of implementation like

1.Implementation of a computer system to replace a manual system.

2.Implementation of a new computer system to replace the existing one.

3.Implementation of a modified application to replace the existing one using the existing one.

The second type of implementation was undergone.

The users, that is, the brokers were given an idea about how the project works and how different it was from the existing one. Sample data were entered and tested for its validity. User's suggestions about the new system were gathered and few modifications required to meet their convenience were undertaken.

Thus the project was implemented to the users satisfaction.

CONCLUSION

CONCLUSION

Thus the project on "screen-based trading" was undertaken in a systematic way and proceeded. The system was implemented by the methodologies suggested in system analysis and system design whereby a system development life cycle was undergone following each of the phases step by step.

This project work no doubt will serve as the best software for the brokers trading at the Coimbatore Stock Exchange and serve them to perform more better in their trading business.

This would also make the easy design and implementation of other operations, as the required master tables have been provided.

As a future enhancement, where the backend is changed to ORACLE, the interaction between the brokers can be easy and also efficient.

BIBLIOGRAPHY - REFERENCES

1. VISUAL BASIC 6.0 COMPLETE REFERENCE

   - *NOEL JERKE*, TATA MCGRAW-HILL

2. VISUAL BASIC 6.0

   - *PAUL SHERIFF*,P.H.I

3. VISUAL BASIC 6.0 PROGRAMMING SECRETS

   - *HAROLD DAVIS*,

   - COMDEXPUBLISHING