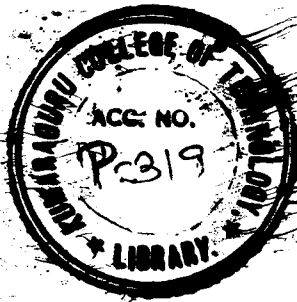
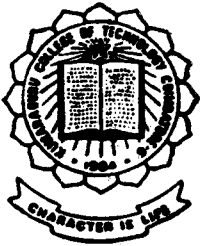


# Financial Accounting System

## PROJECT REPORT



Submitted by  
**R. KRITHIGA**  
**N. MANORANJANI**  
**M.R. RAMDAS**  
**K. SRIVIDYA**

Guided by  
**Ms. A. LAVANYA, B.E.,**

IN PARTIAL FULFILMENT OF THE REQUIREMENTS  
FOR THE AWARD OF THE DEGREE OF  
**BACHELOR OF SCIENCE IN**  
**APPLIED SCIENCES - COMPUTER TECHNOLOGY**  
OF THE BHARATHIAR UNIVERSITY

1997 - 98

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**Kumaraquru College of Technology**

**KUMARAGURU COLLEGE OF TECHNOLOGY**  
Coimbatore – 641 006

Department of Computer Science & Engineering


**CERTIFICATE**

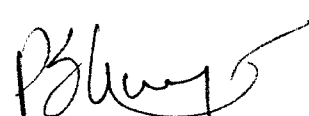
*This is to certify that the report entitled*

***Financial Accounting System***

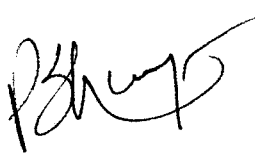
*has been submitted by*


Mr/Ms. \_\_\_\_\_  
in partial fulfillment of the requirements for the award of degree of Bachelor of Science-Applied Sciences (Computer Technology) in the Computer Science & Engineering branch of the Bharathiar University, Coimbatore-641 046 during the year 1997-98.

  
\_\_\_\_\_  
(Guide)

  
\_\_\_\_\_  
(Head of the Department)

Certified that the candidate was examined by us in the project viva-voce examination held on \_\_\_\_\_ and the University Register number is \_\_\_\_\_

  
\_\_\_\_\_  
(Internal Examiner)

  
\_\_\_\_\_  
(External Examiner) 28/4/98



## **K SOFT SYSTEMS LIMITED**

*The Software Solution Company*

249-D, T.V. Swamy Road (E), Coimbatore - 641 002. Tel : +91-422-452324 Fax : +91-422-396458  
e-mail : ksoft.cbe@rmg.sprintrpg.ems.vsnl.net.in

26 Mar 98

### **CERTIFICATE**

This is to certify that  
the following students have done the project titled

**“ FINANCIAL ACCOUNTING SYSTEM ”**

for our organization between December'97 and March'98. They have developed the application in **Power Builder** platform.

1. R. KRITHIGA
2. N. MANORANJANI
3. M.R. RAMDAS
4. K. SRIVIDYA

They have been regular & co-operative during the period of the project.

**KSOFT SYSTEMS LIMITED**

*R. Jayaprakash*

**(R.Jayaprakash)  
Managing Director**

## ACKNOWLEDGEMENT

We wish to express our sincere and heartfelt gratitude to our esteemed *Dr.S.Subramanian, M.Sc (Engg), Ph.D., S.MIEEE, MISTE, Principal of Kumaraguru College Of Technology* for giving us the needed encouragement in starting this project and carrying it out successfully.

Our hearty thanks and a deep sense of gratitude to our *Prof.P.Shanmugam M.Sc (Engg), M.S. (Hawaii), MIEEE, MISTE, Head Of The Department of Computer Technology* for his benevolent attitude to push up this project to its position of success.

We would also like to thank our guide *Ms.A.Lavanya B.E ,MISTE, Lecturer, Department of Computer Science And Engg* without whose motivation and guidance we would not have been able to embark on a project of this magnitude.

We are deeply indebted to *Mr.Kripakar* for enabling us to do the project in *K SOFT SYSTEMS*.

We reciprocate the kindness shown to us by the staff members of department of computer science, friends and parents and to all the members who have supported in getting this project done.

# *Table of Contents*

---

1. Synopsis
2. Introduction
  - 2.1 Organization profile
  - 2.2 About the system
3. System Study
  - 3.1 System requirements
  - 3.2 About Powerbuilder
4. System design and development
  - 4.1 Overview of the system
  - 4.2 Input design
  - 4.3 Code design
  - 4.4 Output design
  - 4.5 Module design
  - 4.6 Database design
5. Future Enhancement
6. Bibliography
7. Appendix
  - A. Data flow diagrams
  - B. Source code
  - C. Screen design
  - D. Reports.

## SYNOPSIS

This FINANCIAL SYSTEM has been developed for group TEXTILE MILLS. This system comprises of design and development of software package for accounting department in a textile mill. Various operations such as keeping track of daily transactions, maintaining various ledgers, preparing monthly / yearly reports etc are carried out in this system.

This system is divided into three modules as:

Master maintenance, Transaction maintenance, Report generation.

Various reports generated by this system also include trial balance, monthly profit and loss schedules, monthly balance sheet schedules etc.

Details about the chapters included are as follows:

The first chapter gives an INTRODUCTION about the system which includes disadvantages of existing manual system and need for this computerized system.

The second chapter is SYSTEM STUDY, which includes organization profile, hardware and software platforms adopted to develop this financial accounting system.

The third chapter is SYSTEM DESIGN which gives an overview about the master and transaction files maintained by this system.

The final chapter gives the IMPLEMENTATION and CONCLUSION about this system.

FLOWCHARTS are given under APPENDIX A.  
SOURCE CODE is given under APPENDIX B.  
SCREEN DESIGN is given under APPENDIX C.  
REPORTS are given under APPENDIX D.

# INTRODUCTION

FINANCIAL ACCOUNTING is defined as the process of recording, classifying and reporting the financial data of a business and preparation of the financial report.

## 1.1 ABOUT THE ORGANIZATION

KSOFT system is a premier & well established software firm. It handles many applications as well as system software projects. It has got many branches in CHENNAI & COIMBATORE. It is also a registered Y2K Training center & it has a tie up with MADURAI KAMARAJ UNIVERSITY. The project managers present in this firm are very efficient & infrastructure is also excellent.

In all it has been a very useful & beneficial experience for us working with this firm.

## 1.2 NEED FOR FINANCIAL ACCOUNTING:

Every business needs to systematically maintain records of all business transactions that take place. The management needs reports to appraise them in their planning.

Apart from the management there are other people like investors, creditors, debtors and employees who are interested in the financial health of a company, which has to be communicated to them. Thus assessing the financial performance of a business is very important whose period the government determines.

The accounting procedure of recording and summarizing the business transaction is known as Book Keeping.



### 1. 3 EXISTING SYSTEM:

The above-mentioned procedures of financial accounting system are now being handled manually. This is accomplished through maintaining various ledgers and registers in which daily transactions are maintained. Each and every small bills and receipts are filed and kept.

### 1. 4 PROPOSED SYSTEM

The proposed system is to computerize all transactions that are taking place in the accounting department. It involves maintaining databases to record the transactions.

### 1. 5 WHY COMPUTERIZED SYSTEM?

We need the computerized system due to the following disadvantages faced in the existing system:

- (i) Time consuming ;
- (ii) Lack of accuracy;
- (iii) Inefficient error detection and correction;
- (iv) Lack of security;

The above mentioned problems can be overcome by the proposed system.

## SYSTEM STUDY

### 2.1 OBJECTIVES OF THE COMPUTERIZED SYSTEM

The following are the main objectives of the proposed computerized financial accounting system:

- (i) To maintain proper record keeping;
- (ii) To keep accuracy;
- (iii) To provide debtors/creditors list as and when required;
- (iv) To provide with various monthly/yearly reports;

## 2.2 HARDWARE PLATFORM

CPU Processor type	: Pentium 100MHz or higher
Storage Media	: 1.2 GB Hard disk or higher
Memory	: 16 MB or higher
Display	: SVGA color Monitor
Other Accessories	: Floppy drive 1.44MB

## 2.3 SOFTWARE PLATFORM

The software platforms adopted to develop this system are as follows:

OPERATING SYSTEM - Windows 95

FRONT END -POWERBUILDER VERSION 5.0

BACKEND - SYBASE SQL ANYWHERE 5.0

## ABOUT POWERBUILDER

Power builder is a graphical client-server application development environment that can be used to build object based database applications that features graphical user interface (GUI).

### GUI APPLICATION

GUI applications have a common look and feel. For example GUI application feature standardized window appearance, menus and dialog boxes. The consistency provided by these standardized features makes GUI applications easy for end users to learn and use. Power builder makes it easy for us to create these applications.

### CLIENT-SERVER COMPUTING

Client-server computing separates the user interface and the database access. Power builder allows to create applications that communicate with a wide variety of server and desktop database.

### OBJECT ORIENTATION

Object oriented programming is based on three major principles:

- (i) INHERITANCE
- (ii) ENCAPSULATION
- (iii) POLYMORPHISM

Power builder windows, menus and user objects can be used to define ancestor objects with encapsulated attributes, events and function. It can then be inherited from those objects to create descendant object. Object orientation benefits us by making our work more modular reusable, extensible, flexible and powerful.

### ADVANTAGES OF POWERBUILDER:

Power builder provides a state-of-the-art application development book for client/server database computing. What makes power builders applications unique is the data window, which is used to retrieve and display data.

Data window has the following advantages:

- (i) A data window can be built with little or no knowledge of SQL.
- (ii) To build a data window, the information to be retrieved from the database is specified by selecting the items in the data window painter.
- (iii) It reduces the number of system rescues required to represent data in forms, tables, or reports.

### STEPS TO BUILD SAMPLE APPLICATION:

1. Create a database table. This step is required only if the database does not already contain the data that is to be accessed in the application.
2. Build an application object. If the sample application is installed

them an application object is there already. However it is required to create a new application and make it the default one.

3. Build a window. The controls are placed in the window to communicate with the user. By clicking a control or selecting an item from a list, the user can initiate an activity, respond to a request from the application, retrieve data from the database or update the database.

4. Builds a data window objects to present manipulate and update information from a relational database or another data source.

5. Place a data window control in the window, associate the data window object built with the control and build scripts to connect to the database and retrieve, change, and update the data. After that, the users can use the data window to display , maintain and update data.

6. Build a menu for the window ( optional).

7. Associate the menu with the window.

8. Create an executable file so that the application can be delivered to end users and run from windows.

After completing these steps , a simple powerbuilder application is got which can be enhanced by using other powerbuilder painter and features .

# SYSTEM DESIGN AND DEVELOPMENT

This chapter explains the system design and development, the overview of the system, input design, output design, database design and the process design in detail.

## 3.1 OVERVIEW OF THE SYSTEM

Accounting is the art of recording, classifying and summarizing in a significant manner and in terms of money, transactions and events which are in part at least of a financial character and interpreting the results thereof.

Towards achieving this objective, the accounting system records day-to-day financial transactions in a systematic manner for producing a number of statements that aid the management in decision making.

Transaction is the activity involving the transfer of money or goods or service between two persons or two accounts.

The two important aspects in the accounting procedures are

- DEBTOR- One who receives a benefit without giving money or money's worth.
- CREDITOR- One who gives a benefit without receiving money or money's worth.

Voucher is a written document in support of a transaction. It is a proof that transaction has taken place the value stated in the voucher. Each transaction when closely analyzed reveals two aspects.

One aspect will be the 'receiving aspect' or the 'incoming' or the 'expense aspect'. This is known as debit aspect. Another aspect will be the 'giving aspect' or the creditor aspect. These two form the double entry system.

The rules for the double entry system are:

Debit the Receiver, Credit the Giver.

Debit what comes in, Credit what goes out.

Debit all expenses and losses, Credit all incomes and gains.

A journal is the book of original entries wherein transactions are first recorded. In journal each transaction is dealt with separately. Ledger is the main or principle book of accounts. Transactions recorded in the journal and subsidiary registers are transferred to the accounts concerned in the ledger in a summarized form. The transferring process is called posting.

The trial balance is a tabular statement which presents the consolidated list of all Ledger balances.

In our system the transactions in the computerized form are divided into four types. The first type depends on the voucher that is being raised, For example cash transactions correspond to one type and bank transactions correspond to another type. Also associated with each account head is an account type. The account type gives the category in which the account head falls.



## 3.2 INPUT DESIGN

Input design is a part of the overall system design which requires much attention. The objectives of the input design is to achieve the highest possible level of accuracy for the data captured as input. The error in the input data will end up with unexpected results. The input/output design process is given as flowchart in Appendix A.

Erroneous data entered during data entry are handled carefully as a lot of data validation facilities are provided in the system. On each and every important field appropriate validation should be done then and there. If the user makes a mistake or an error during data entry, the system displays a message. This message should give a clear indication regarding what is expected from the user .

### 3.2.1 MASTER MAINTENANCE

In any application master maintenance is meant to maintain data that are relatively permanent. Master has to be created before any transaction pertaining to the information in the master file occurs. Facilities that exists in this module are addition, modification, deletion and viewing. This is achieved through corresponding user friendly screens taking the first character of the option an action. This system maintains a number of Master Files.

### 3. 2. 2 TRANSACTION MAINTENANCE

The transaction maintenance in any application is meant to update the transaction files and master files. The data stored in the transaction are relatively temporary. The transaction files are created after creating the master files. Facilities that exists are addition , modification , deletion. This is achieved through corresponding user friendly screens taking the first character of the option an action. This system maintains a number of transaction fields.

### 3. 2. 3 ADDITION

Addition is used to append new data records into the file. On selection of addition (or) new button the addition module is invoked, the details of the master file to be added are sought and program tries to append the detail of the corresponding master file. Depending on the master file certain validation are carried out on the key field and on other fields and on no error condition, the record is added to the master file. Duplicate entries in the files are filtered and errors are pointed out using error messages.

### 3. 2. 4 MODIFICATION

Modification is used to change an existing database record. Any record that is inquired is captured in pointer variable and the user is allowed to enter in the corresponding text boxes. If the user presses cancel button without saving the corresponding original entries are fetched from table using the pointer and then displayed.

If the user presses save button then sufficient messages are given to confirm saving and for validations.

### 3. 2. 5 DELETION

Deletion option is used to delete a particular record which is already stored in the database table. Similar to modification the pointer concept is used here to fetch records and delete records. Even after selection of deletion option, the process can be terminated without the record being deleted. The deletion operation fetches the required record and after confirmation the record is deleted.

### 3. 3 CODE DESIGN.

The codes used in this system to identify transactions are

GLCODE - General ledger code ( GC--)

SLCODE - Sub ledger code ( SC--)

TCCODE - Transaction code

Various types of transactions and their corresponding codes are as follows:

<u>CODES</u>	<u>TRANSACTIONS</u>
--------------	---------------------

01 - 10	: Cash transactions .
---------	-----------------------

11 - 20	: Bank transactions .
---------	-----------------------

21 - 30 : Purchase transactions .

31 - 40 : Sales transactions .

41 : General journal transactions .

### 3.4 OUTPUT DESIGN :

The output generated by this system can be classified into two types as given below .

1. ACCOUNT BOOKS .

2. FINAL ACCOUNTS .

#### 3.4.1 ACCOUNT BOOKS .

This Account Books includes cash book , bank book, ledger summary , sales journal and purchase journal .

Cash book and bank book gives detail about credit amount, debit amount and current balance for each month. Sales journal and purchase journal gives list of party names and sales/purchase amount.

Ledger summary contains all account head details with their current balance.

SAMPLE REPORTS are presented in APPENDIX C.

#### 3.4.2 FINAL ACCOUNTS :

The Final Accounts can be classified into three options

as given below :

- \* TRIAL BALANCE .
- \* PROFIT AND LOSS ACCOUNT .
- \* BALANCE SHEET .

### TRIAL BALANCE .

The Trial Balance shows the initial preview for a balance with the listing of credit and debit balances under each Account Head . The totality of all credit and debit balances are shown at the end with pagewise total .

### TRADING ACCOUNT .

This is more specific to the trading activity of a company such as buying and selling of goods. This report contains two sides , creditor side and debtor side . The Opening stock , Purchases , Purchase returns , Direct expenses will come under Debtor side and Sales returns , Closing stock comes under Creditor side . The balance for each side is calculated and if there is surplus in creditor side , it is carried over to Debtor side as Gross profit and if there is surplus in debtor side , it is carried over to Creditor side as Gross loss , such that the net difference between credit and debit balance is zero . The sample report is given in APPENDIX C.

### PROFIT AND LOSS ACCOUNT .

This is directly related with Trading account but here the difference is that this report shows only Expenses and

Incomes . The final balance is shown for both debit and credit sides and surplus is carried over as in Trading account as Gross profit or Gross loss .

### BALANCE SHEET .

The Balance sheet is the consolidated output which combines both the Trading and P/L account . Balance sheet is divided into Liabilities and Assets . Liabilities include supplier / customer , other creditors , bank loan , credit card loan and Assets include customer , other debtors , stock in trade , cash on hand , cash on bank and credit card . The net profit is added to Liabilities if it is surplus in assets or net loss added to assets if there is a surplus in Liabilities such that both the balances are equalled. This is a most important report regarding company's performance in ended financial year which is published as a company's year end report .

### 3. 5 MODULE DESIGN .

The three modules of this system are Master maintenance module , Transaction maintenance module and Report generation module .

#### 3. 5. 1 MASTER MAINTENANCE MODULE

This module deals with the operations of addition , deletion , updation and modification of Master database files .

This module is again subdivided into various module such as

- \* Firm details.
- \* Ledger details.
- \* Transaction details.

### FIRM DETAILS.

This sub module gives complete information about various unit in this firm with their address and other details such as phone numbers fax etc. The firm details can be edited by selecting accordingly from main menu.

### LEDGER DETAILS

Ledger maintained can be classified into

- \* General ledger
- \* sub ledger

General ledger giving details about account heads and their corresponding codes. Sub ledger giving details about the customer and their codes.

### TRANSACTION DETAILS

This master module given details about the various types of transactions and their corresponding transaction codes.

### 3. 5. 2 TRANSACTION MODULE

Transaction occurs when inputs are received in the system in the form of Debit/Credit Vouchers, Receipts, Bills and other documents relating to exchange of money inside or outside the system which affects the Financial matters of the company directly or indirectly.

#### CASH TRANSACTION

Any kind of transaction involving cash is entered in sub module. The inputs for this module are Date, Voucher No, Party account No./Cheque No. and the amount involve in the transaction. This is enter in General transaction files with respective account head entries and transaction type either Debit or Credit.

#### BANK TRANSACTION

This is similar to cash transaction in fields and account head but the entries are made in general transaction files based on bank cheques.

#### SALES/PURCHASE TRANSACTIONS

The input to this module are date, invoice no or bill no, party name and amount in rupees. These are directly entered into transaction files and the Master files are updated simultaneously.



## JOURNAL TRANSACTION

This is a transaction which involves inputs such as date , reference no description of transaction and amount in rupees which are directly entered in the corresponding transaction file by passing the transaction type as journal transaction .

### 3.6 DATABASE DESIGN.

The various tables used in the project are as follows:

#### 3.6.1 MASTER DATABASE FILES

The files which have permanent entries come under master files category. These files are maintained using the transaction files, which contain changing data. The master files are generally given more secured access controls than the transaction files.

The master files maintained are:

##### 1. Unit Master

The unit master gives details about the various units in the company each identified by a unique unit code.

FIELD NAME	TYPE	WIDTH
UNIT CODE	VARCHAR	4
UNIT NAME	VARCHAR	40
ADDRESS	VARCHAR	40
STATE	VARCHAR	20
PHONE	NUMERIC	14
TELEX	NUMERIC	14
FAX	NUMERIC	14

##### 2. General Ledger Master

The general ledger master is a composite combination of two other general masters namely the General ledger

name master & a General ledger master . The General ledger name master contains the information of account head and its corresponding code whereas the General ledger master has the other financial details.

TABLE NAME : GENERAL LEDGER NAME MASTER

FIELDS	TYPE	WIDTH
GLCODE	VARCHAR	5
GLHEAD	VARCHAR	40

TABLE NAME:GENERAL LEDGER MASTER

FIELDS	TYPE	WIDTH
UNITCODE	VARCHAR	4
GLCODE	VARCHAR	5
YEAR OPENING BALANCE	NUMERIC	14,2
CURRENT BALANCE	NUMERIC	14,2
DB_4	NUMERIC	14,2
CR_4	NUMERIC	14,2
DB_5	NUMERIC	14,2
CR_5	NUMERIC	14,2
DB_6	NUMERIC	14,2
CR_6	NUMERIC	14,2
DB_7	NUMERIC	14,2
CR_7	NUMERIC	14,2
DB_8	NUMERIC	14,2
CR_8	NUMERIC	14,2
DB_9	NUMERIC	14,2
CR_9	NUMERIC	14,2
DB_10	NUMERIC	14,2
CR_10	NUMERIC	14,2
DB_11	NUMERIC	14,2
CR_11	NUMERIC	14,2
DB_12	NUMERIC	14,2
CR_12	NUMERIC	14,2
DB_1	NUMERIC	14,2
CR_1	NUMERIC	14,2
DB_2	NUMERIC	14,2

CR_2	NUMERIC	14,2
DB_3	NUMERIC	14,2
CR_3	NUMERIC	14,2
LAST TRANSACTION DATE	DATE	

### 3.Sub Ledger Master

The sub ledger master is again a composite combination of two sub ledger masters namely the sub ledger name master & the sub ledger master which in a similar way in accordance to the previous master files hold the sub ledger financial details & sub ledger headers.

TABLE NAME :SUB LEDGER NAME MASTER

FIELDS	TYPE	WIDTH
SLCODE	VARCHAR	7
SLHEAD	VARCHAR	40
ADDRESS	VARCHAR	40
CITY	VARCHAR	20
STATE	VARCHAR	20
PINCODE	CHAR	6
PHONE	NUMERIC	13
FAX	NUMERIC	13
TELEX	NUMERIC	13

TABLE NAME: SUB LEDGER MASTER

FIELDS	TYPE	WIDTH
UNITCODE	VARCHAR	4
GLCODE	VARCHAR	5
SLCODE	VARCHAR	7
YEAR OPENING BALANCE	NUMERIC	14,2
CURRENT BALANCE	NUMERIC	14,2
DB_4	NUMERIC	14,2

CR_4	NUMERIC	14,2
DB_5	NUMERIC	14,2
CR_5	NUMERIC	14,2
DB_6	NUMERIC	14,2
CR_6	NUMERIC	14,2
DB_7	NUMERIC	14,2
CR_7	NUMERIC	14,2
DB_8	NUMERIC	14,2
CR_8	NUMERIC	14,2
DB_9	NUMERIC	14,2
CR_9	NUMERIC	14,2
DB_10	NUMERIC	14,2
CR_10	NUMERIC	14,2
DB_11	NUMERIC	14,2
CR_11	NUMERIC	14,2
DB_12	NUMERIC	14,2
CR_12	NUMERIC	14,2
DB_1	NUMERIC	14,2
CR_1	NUMERIC	14,2
DB_2	NUMERIC	14,2
CR_2	NUMERIC	14,2
DB_3	NUMERIC	14,2
CR_3	NUMERIC	14,2
LAST TRANSACTION DATE	DATE	

#### 4.Transaction Control Master

The transaction control master has the transaction names & its corresponding codes.

FIELDS	TYPE	WIDTH
UNIT CODE	VARCHAR	4
TCCODE	VARCHAR	2
TCHEAD	VARCHAR	40
GLCODE	VARCHAR	5

SLCODE	VARCHAR	7
LAST VOC NO	CHAR	6

### 3.6.2 TRANSACTION FILES

The various transaction files are as follows:

#### 1. Cash bank transaction

The cash bank transaction file holds the details about cash transactions involved in the organization & the corresponding updations are made in General ledger master.

FIELDS	TYPE	WIDTH
UNIT CODE	VARCHAR	4
GLCODE	VARCHAR	5
TCCODE	CHAR	2
VOC NO	CHAR	6
VOC DATE	DATE	
VOCS NO	CHAR	3
REF NO	CHAR	5
REF DT	DATE	
CHQ NO	CHAR	5
CHQ DATE	DATE	
AMOUNT	NUMERIC	14,2
DB/CR	CHAR	1
CONTRA	CHAR	1
CAN FLAG	CHAR	1

#### 2. Sales transactions

The sales transaction again has two sub transaction tables namely sales header & sales trailer. The sales header contains information about the sales transactions for each SLCODE maintained in sub ledger master . The sales trailer contains information about various transactions under the single SLCODE i.e. the sales trailer is a detail block for the master sales header and the corresponding updations are done in debit field of sub ledger master.

TABLE NAME: SALES HEADER

FIELDS	TYPE	WIDTH
UNIT CODE	VARCHAR	4
SLCODE	VARCHAR	7
TCCODE	CHAR	2
VOC NO	CHAR	6
VOC DATE	DATE	
INV NO	CHAR	6
INV DATE	DATE	
TAX CODE	CHAR	5
TAX AMOUNT	NUMERIC	14,2
TOTALAMOUNT	NUMERIC	14,2
ROUND OF	NUMERIC	14

TABLE NAME :SALES TRAILER

FIELDS	TYPE	WIDTH
UNIT CODE	VARCHAR	4
SLCODE	VARCHAR	7
VOC NO	CHAR	6
VOCS NO	CHAR	3
PRODUCT CODE	CHAR	3
QUANTITY	NUMERIC	5
RATE	NUMERIC	5,2
AMOUNT	NUMERIC	10,2
CAN FLAG	CHAR	1

### 3. Purchase transactions

The purchase transaction again has two sub transaction tables namely purchase header & purchase trailer which holds information in a similar way as sales transactions but the difference is that the corresponding updations in sub ledger master are made in credit field for the corresponding SLCODE in the corresponding month.

#### TABLE NAME: PURCHASE HEADER

FIELDS	TYPE	WIDTH
UNIT CODE	VARCHAR	4
SLCODE	VARCHAR	7
TCCODE	CHAR	2
VOC NO	CHAR	6
BILL NO	CHAR	6
BILL DATE	DATE	
BILL AMT	NUMERIC	14,2

#### TABLE NAME: PURCHASE TRAILER

FIELDS	TYPE	WIDTH
UNIT CODE	VARCHAR	4
SLCODE	VARCHAR	7
VOC NO	CHAR	6
VOCS NO	CHAR	5
PRODUCT CODE	CHAR	5
QUANTITY	NUMERIC	5
RATE	NUMERIC	5,2
AMOUNT	NUMERIC	10,2
CAN FLAG	CHAR	1



## *Future Enhancement*

As Accounting is one of the major operations performed in any organization this project by performing that task simplifies the procedure thus making it useful

This project being a highly interactive one facilitates the easy approach by the user & provides a simple path for performing all the necessary transactions.

Another significant aspect of this project is that it can be further modified so as to suit the needs that may change from time to time. As it does not adhere to a rigid concept it does not pose a threat to future modifications.

# *Bibliography*

The following books were referred during the course of the project both for reference & for modifications:

1. *Power Builder 5.0* by David McLanahan
2. *Power Builder & Applications* by Ivan Bayross
3. *Power Builder 5.0 Unleashed*
4. *Financial Accounting System* by F.J.Taylor

# PROCESS DESIGN

MASTER  
MAINTENANCE

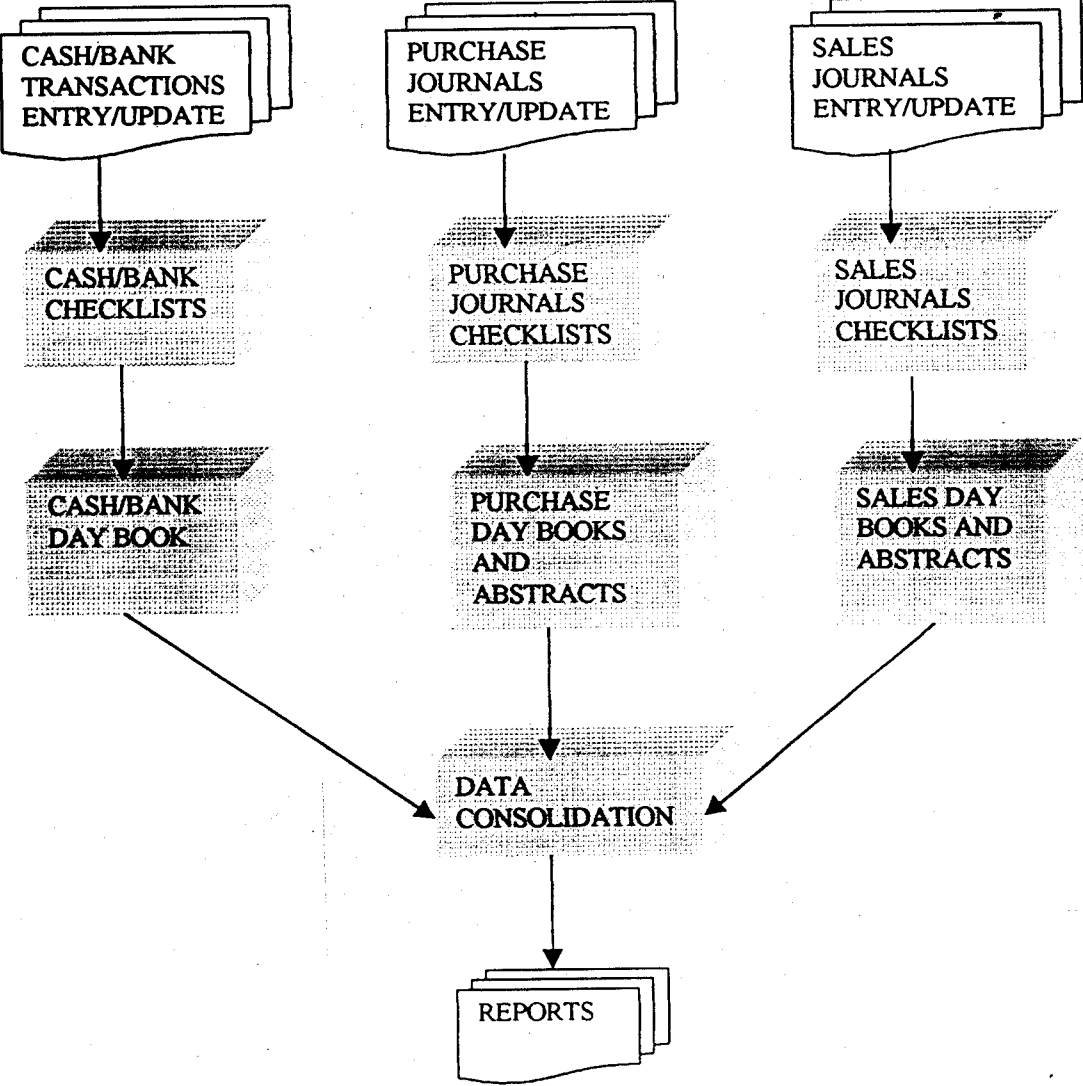
TRANSACTION  
MAINTENANCE

FINANCIAL  
ACCOUNTING  
SYSTEM

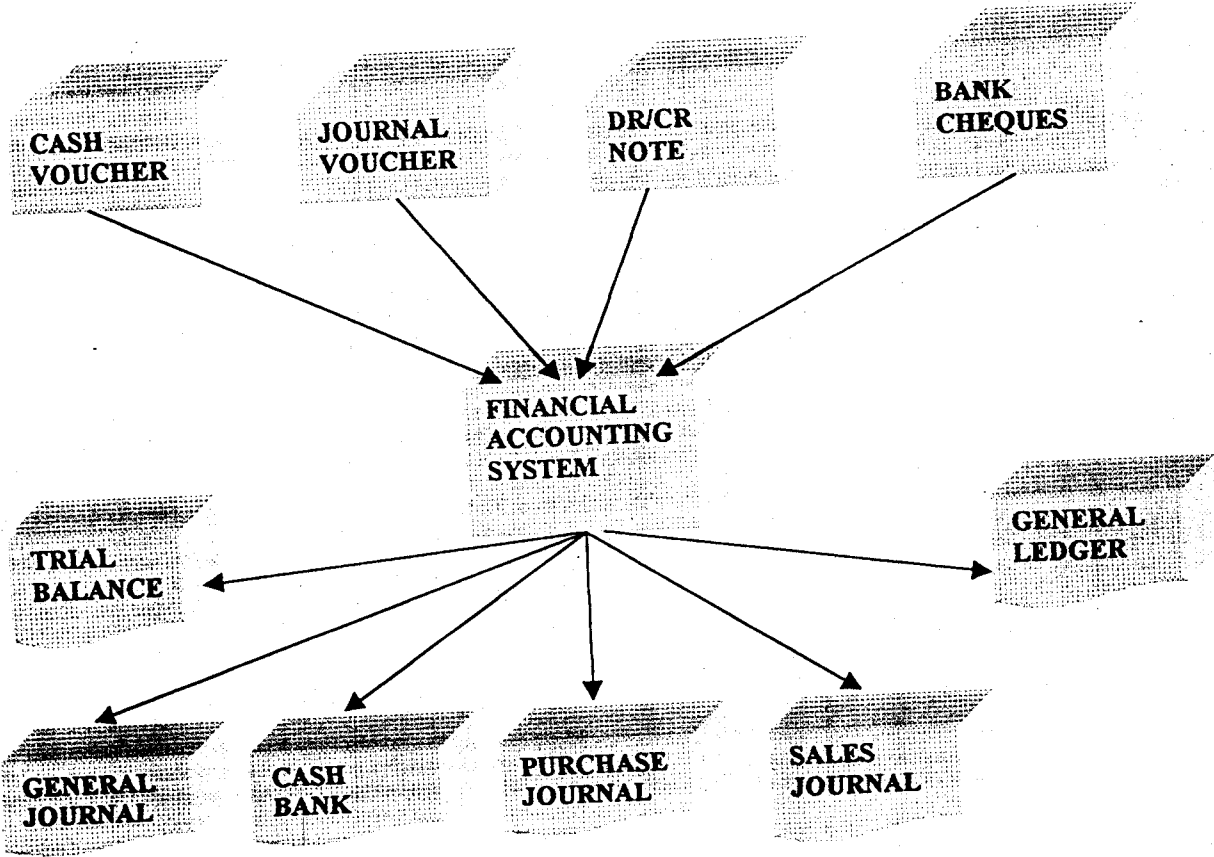
REPORTS

```
graph TD; MS[MASTER MAINTENANCE] --> FAS[FINANCIAL ACCOUNTING SYSTEM]; TM[TRANSACTION MAINTENANCE] --> FAS; FAS --> R[REPORTS];
```

**INFORMATION FLOW IN THE FINANCIAL ACCOUNTING SYSTEM**



INPUT/OUTPUT DESIGN



# MENU FORMAT

**FINANCIAL  
ACCOUNTING  
SYSTEM**

**TRANSACTIONS**

1)CASH  
BANK  
2)SALES  
JOURNAL  
3)PURCHASE  
JOURNAL

**MASTERS**

1)GENERAL  
LEDGER  
MASTER  
2)SUB  
LEDGER  
MASTER  
3)UNIT  
MASTER  
4)TRANSACTION  
CONTROL  
MASTER

**REPORTS**

1)TRIAL  
BALANCE  
2)GENERAL  
LEDGER  
3)SUB  
LEDGER  
4)SALES  
JOURNAL  
5)PURCHASE  
JOURNAL  
6)CASH BANK  
7)PARTY  
TRIAL  
BALANCE

## CODING

### Coding for Addition

```
dis_abl(pb_add,pb_mod,pb_del,pb_view,pb_save,pb_exit)
dw_1.reset()
dw_1.insertrow(0)
dw_1.setfocus()
```

### Coding for Modification

```
dis_abl(pb_mod,pb_add,pb_del,pb_view,pb_save,pb_exit)
dw_1.retrieve()
dw_1.setfocus()
```

### Coding for Deletion

```
disable(pb_del,pb_mod,pb_add,pb_view,pb_save,pb_exit)
dw_1.deleterow(0)
```

### Coding for View

```
dw_1.retrieve(0)
```

### Coding for Save

```
dw_1.update()
dw_1.reset()
dw_1.insertrow(0)
enab(pb_add,pb_del,pb_mod,pb_save,pb_view,pb_exit)
messagebox("Information","Updatons made")
```

### Coding for Exit

```
close(parent)
```

## Coding For Main Data Window

```
CHOOSE CASE flag
  CASE "um"
    dw_1.dataobject='d_um'
    dw_1.title="UNITMASTER"
  case "gn"
    dw_1.dataobject='d_gle'
    dw_1.title="GENERAL LEDGER NAME MASTER"
  case "gl"
    dw_1.dataobject='d_gled'
    dw_1.title="GENERAL LEDGER MASTER"
  case "sn"
    dw_1.dataobject='d_sn'
    dw_1.title="SUB LEDGER NAME MASTER"
  case "sl"
    dw_1.dataobject='d_sl'
    dw_1.title="SUB LEDGER MASTER"
  case "cm"
    dw_1.dataobject='d_con'
    dw_1.title="TRANSACTION CONTROL MASTER"
END CHOOSE
dw_1.settransobject(sqlca)
dw_1.insertrow(0)
dw_1.retrieve()
pb_exit.setfocus()
```

## Coding for Transaction(Main)

```
CHOOSE CASE flag
  CASE "ca"
    dw_1.dataobject='d_cba'
    dw_1.title="Cash Header Record"
    dw_2.dataobject='d_cbal'
    dw_2.title="Cash Trailer Record"
    flag1="ct"
  CASE "pur"
    dw_1.dataobject='d_phdr'
    dw_1.title="Purchase Header Record"
    dw_2.dataobject='d_pt'
    dw_2.title="Puchase Trailer Record"
```



```

        flag1="pt"
    case "ba"
        dw_1.dataobject='d_cba'
        dw_1.title="Bank Header Record"
        dw_2.dataobject='d_cba1'
        dw_2.title="Bank Trailer Record"
        flag1="ct"
    case "sa"
        dw_1.dataobject='d_sahd'
        dw_1.title="Sales Header Record"
        dw_2.dataobject='d_st'
        dw_2.title="Sales Trailer Record"
        flag1="st"

```

END CHOOSE

```

dw_1.settransobject(sqlca)
dw_1.retrieve()

```

```

dw_2.settransobject(sqlca)
dw_2.retrieve()

```

```

string dl
//real sl2,sl1
date d
choose case flag1
    case "pt"
        u=dw_1.getitemstring(dw_1.getrow(),"ph_um_unitcd")
        s=dw_1.getitemstring(dw_1.getrow(),"ph_sm_slcode")
        t=dw_1.getitemstring(dw_1.getrow(),"ph_cm_tccode")
        select "cm_lvocno" into :v from "cont_mas" where
            "cl_um_unitcd"=:u and "cm_tccode"=:t and
            "cm_sm_slcode"=:s;
        dw_1.setitem(dw_1.getrow(),"ph_vocno",v)
        d=dw_1.getitemdate(dw_1.getrow(),"ph_vocdt")
        am=dw_1.getitemnumber(dw_1.getrow(),"ph_billamt")
        dl=string(d)
        m=mid(dl,1,2)
        sle_1.text=m
        sle_2.text=string(am)

```

```

v2=integer(v)
case "st"
u=dw_1.getitemstring(dw_1.getrow(),"sa_um_unitcd")
s=dw_1.getitemstring(dw_1.getrow(),"sa_sm_slcode")
t=dw_1.getitemstring(dw_1.getrow(),"sa_cm_tccode")
select "cm_lvocno" into :v from "cont_mas" where
"cl_um_unitcd"=:u and "cm_tccode"=:t and
"cm_sm_slcode"=:s;
dw_1.setitem(dw_1.getrow(),"sa_vocno",v)
d=dw_1.getitemdate(dw_1.getrow(),"sa_vocdt")
am=dw_1.getitemnumber(dw_1.getrow(),"sa_totamt")
dl=string(d)
m=mid(dl,1,2)
sle_1.text=m
sle_2.text=string(am)
v2=integer(v)
case "ct"
u=dw_1.getitemstring(dw_1.getrow(),"ft_um_unitcd")
s=dw_1.getitemstring(dw_1.getrow(),"ft_gm_glcode")
t=dw_1.getitemstring(dw_1.getrow(),"ft_cm_tccode")
select "cm_lvocno" into :v from "cont_mas" where
"cl_um_unitcd"=:u and "cm_tccode"=:t and
"cm_gm_glcode"=:s;
dw_1.setitem(dw_1.getrow(),"ft_vocno",v)
d=dw_1.getitemdate(dw_1.getrow(),"ft_vocdt")
dl=string(d)
m=mid(dl,1,2)
sle_1.text=m
v2=integer(v)
case "gj"
u=dw_1.getitemstring(dw_1.getrow(),"fj_um_unitcd")
s=dw_1.getitemstring(dw_1.getrow(),"fj_gm_glcode")
t=dw_1.getitemstring(dw_1.getrow(),"fj_cm_tccode")
select "cm_lvocno" into :v from "cont_mas" where
"cl_um_unitcd"=:u and "cm_tccode"=:t and
"cm_gm_glcode"=:s;
dw_1.setitem(dw_1.getrow(),"fj_vocno",v)
d=dw_1.getitemdate(dw_1.getrow(),"fj_vocdt")
dl=string(d)
m=mid(dl,1,2)
sle_1.text=m
v2=integer(v)

```

```
dc=dw_1.getitemstring(dw_1.getrow(),"fj_dbcr")
am=dw_1.getitemnumber(dw_1.getrow(),"fj_amount")
```

```
flag1="ct"
end choose
```

```
choose case flag1
  case "ct"
```

```
dc=dw_2.getitemstring(dw_2.getrow(),"fd_dbcr")
am=dw_2.getitemnumber(dw_2.getrow(),"fd_amount")
```

```
sle_1.text=m
```

```
sle_2.text=string(am)
```

```
//case "bt"
```

```
//u=dw_1.getitemstring(dw_1.getrow(),"ft_cm_unitcd")
```

```
//s=dw_1.getitemstring(dw_1.getrow(),"ft_gm_glcode")
```

```
//t=dw_1.getitemstring(dw_1.getrow(),"ft_cm_tccode")
```

```
//select "cm_lvocno" into :v from "cont_mas" where
```

```
"cl_um_unitcd"=:u and "cm_tccode"=:t and
```

```
"cm_gm_glcode"=:s;
```

```
//dw_1.setitem(dw_1.getrow(),"ft_vocno",v)
```

```
//d=dw_1.getitemdate(dw_1.getrow(),"ft_vocdt")
```

```
//dc=dw_2.getitemstring(dw_2.getrow(),"ft_dbcr")
```

```
//am=dw_2.getitemnumber(dw_2.getrow(),"ft_amount")
```

```
//d1=string(d)
```

```
//m=mid(d1,1,2)
```

```
//sle_1.text=m
```

```
//sle_2.text=string(am)
```

```
end choose
```

### Coding for Save

```
real sa
```

```
choose case flag1
```

```
  case "pt"
```

```
  update "cont_mas" set "cont_mas"."cm_lvocno"=string(:v2 +
```

```
  1) where "cl_um_unitcd"=:u and "cm_tccode"=:t and
```

```
  "cm_sm_slcode"=:s;
```

```
  choose case m
```

```
    case "12"
```

```

        select "sl_cr12" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
        am=am + sa
        UPDATE "sled_mas"
        SET "sled_mas"."sl_cr12" = :am where "sl_um_unitcd"=:u
and "sl_sm_slcode"=:s;
        messagebox("amount", am)
case "11"
        select "sl_cr11" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr11"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "10"
        select "sl_cr10" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr10"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "9/"
        select "sl_cr9" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr9"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "8/"
        select "sl_cr8" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr8"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "7/"
        select "sl_cr7" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr7"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;

```

```

case "6/"
    select "sl_cr6" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr6"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "5/"
    select "sl_cr5" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr5"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "4/"
    select "sl_cr4" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr4"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "3/"
    select "sl_cr3" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr3"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "2/"
    select "sl_cr2" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr2"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "1/"
    select "sl_cr1" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am+sa
update "sled_mas"
set "sled_mas"."sl_cr1"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;

```

```

end choose
case "st"
update "cont_mas" set "cont_mas"."cm_lvocno"=string(:v2 +1)
where "cl_um_unitcd"=:u and "cm_tccode"=:t and
"cm_sm_slcode"=:s;
choose case m
    case "12"
        select "sl_db12" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
        sle_3.text=string(sa)

        am=am + sa
        UPDATE "sled_mas"
        SET "sled_mas"."sl_db12" = :am where "sl_um_unitcd"=:u
and "sl_sm_slcode"=:s;
        messagebox("amount", am)
    case "11"
        select "sl_db11" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
        sle_3.text=string(sa)
        am=am + sa
    update "sled_mas"
    set "sled_mas"."sl_db11"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
    case "10"
        select "sl_db10" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
        am=am + sa
    update "sled_mas"
    set "sled_mas"."sl_db10"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
    case "9/"
        select "sl_db9" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
        am=am + sa
    update "sled_mas"
    set "sled_mas"."sl_db9"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
    case "8/"
        select "sl_db8" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;

```

```

am=am+ sa
update "sled_mas"
set "sled_mas"."sl_db8"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "7/"
    select "sl_db7" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am + sa
update "sled_mas"
set "sled_mas"."sl_db7"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "6/"
    select "sl_db6" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am + sa
update "sled_mas"
set "sled_mas"."sl_db6"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "5/"
    select "sl_db5" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am + sa
update "sled_mas"
set "sled_mas"."sl_db5"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "4/"
    select "sl_db4" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am + sa
update "sled_mas"
set "sled_mas"."sl_db4"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "3/"
    select "sl_db3" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am + sa
update "sled_mas"
set "sled_mas"."sl_db3"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "2/"
    select "sl_db2" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;

```

```

am=am + sa
update "sled_mas"
set "sled_mas"."sl_db2"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;
case "1/"
    select "sl_db1" into :sa from "sled_mas" where
"sl_um_unitcd"=:u and "sl_sm_slcode"=:s;
am=am + sa
update "sled_mas"
set "sled_mas"."sl_db1"=:am where "sl_um_unitcd"=:u and
"sl_sm_slcode"=:s;

    end choose
case "ct"
    update "cont_mas" set
"cont_mas"."cm_lvocno"=string(:v2 + 1) where
"cl_um_unitcd"=:u and "cm_tccode"=:t and "cm_gm_glcode"=:s;

choose case dc
    case "c"
choose case m
    case "12"
        select "gl_cr12" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
        sle_3.text=string(sa)
        am=am + sa
        UPDATE "gled_mas"
SET "gled_mas"."gl_cr12" = :am where "gl_gm_unitcd"=:u
and "gl_gm_glcode"=:s;
        messagebox("amount", am)
case "11"
    select "gl_cr11" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
    sle_3.text=string(sa)
    am=am+sa
    messagebox("amount", am)
    update "gled_mas"
set "gled_mas"."gl_cr11"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;

case "10"

```



```

        select "gl_cr10" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr10"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "9/"
        select "gl_cr9" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr9"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "8/"
        select "gl_cr8" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr8"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "7/"
        select "gl_cr7" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr7"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "6/"
        select "gl_cr6" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"

```

```

set "gled_mas"."gl_cr6"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "5/"
    select "gl_cr5" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr5"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "4/"
    select "gl_cr4" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr4"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "3/"
    select "gl_cr3" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr3"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "2/"
    select "gl_cr2" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_cr2"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "1/"
    select "gl_cr1" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)

```

```

am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_crl"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
end choose
    case "d"
choose case m
    case "12"
        select "gl_db12" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
        sle_3.text=string(sa)
        am=am + sa
        UPDATE "gled_mas"
        SET "gled_mas"."gl_db12" = :am where "gl_gm_unitcd"=:u
and "gl_gm_glcode"=:s;
        messagebox("amount", am)
    case "11"
        select "gl_db11" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
        sle_3.text=string(sa)
        am=am+sa
        messagebox("amount", am)
        update "gled_mas"
        set "gled_mas"."gl_db11"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
    case "10"
        select "gl_db10" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
        sle_3.text=string(sa)
        am=am+sa
        messagebox("amount", am)
        update "gled_mas"
        set "gled_mas"."gl_db10"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
    case "9/"
        sle_1.text=m
        select "gl_db9" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
        sle_3.text=string(sa)
        am=am+sa

```

```

messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db9"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "8/"
    sle_1.text=m
    select "gl_db8" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db8"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "7/"
    sle_1.text=m
    select "gl_db7" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db7"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "6/"
    sle_1.text=m
    select "gl_db6" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db6"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "5/"
    sle_1.text=m
    select "gl_db5" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"

```

```

set "gled_mas"."gl_db5"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "4/"
    sle_1.text=m
    select "gl_db4" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db4"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "3/"
    select "gl_db3" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db3"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "2/"
    select "gl_db2" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db2"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
case "1/"
    select "gl_db1" into :sa from "gled_mas" where
"gl_gm_unitcd"=:u and "gl_gm_glcode"=:s;
sle_3.text=string(sa)
am=am+sa
messagebox("amount", am)
update "gled_mas"
set "gled_mas"."gl_db1"=:am where "gl_gm_unitcd"=:u and
"gl_gm_glcode"=:s;
end choose
end choose
end choose

```

```
dw_1.update()
dw_2.update()
messagebox("Hello", "Updatons made")
enab(pb_add, pb_del, pb_exit, pb_mod, pb_view, pb_save)
```

### Coding for Add

```
dis_abl(pb_add, pb_del, pb_mod, pb_view, pb_save, pb_exit)
//choose case flag1
    //case "pt"
choose case a
    case 1
        dw_1.update()
        dw_1.reset()
        dw_1.insertrow(0)
        dw_1.setfocus()
        dw_2.update()
        dw_2.reset()
        dw_2.insertrow(0)
    case 2
        dw_2.update()
        dw_2.insertrow(0)
        dw_2.setfocus()
end choose
```

### Coding for Modify

```
dis_abl(pb_mod, pb_del, pb_add, pb_view, pb_save, pb_exit)
choose case a
    case 1
        dw_1.retrieve()
        dw_1.setfocus()
    case 2
        dw_2.retrieve()
        dw_2.setfocus()
end choose
```

### Coding for Deletion

```
dis_abl(pb_del,pb_mod,pb_add,pb_view,pb_save,pb_exit)
integer r,i

choose case a
  case 1
    dw_1.deleterow(0)
    r=dw_2.rowcount()
    FOR i=1 TO r
      dw_2.deleterow(0)
    NEXT

  case 2
    dw_2.deleterow(0)
end choose
```

### Coding for View

```
integer i,r
CHOOSE CASE a
  CASE 1
    dw_1.retrieve()
    r=dw_2.rowcount()
    for i=1 to r
      dw_2.retrieve(i)
    next
  case 2
    dw_2.retrieve()
END CHOOSE
```