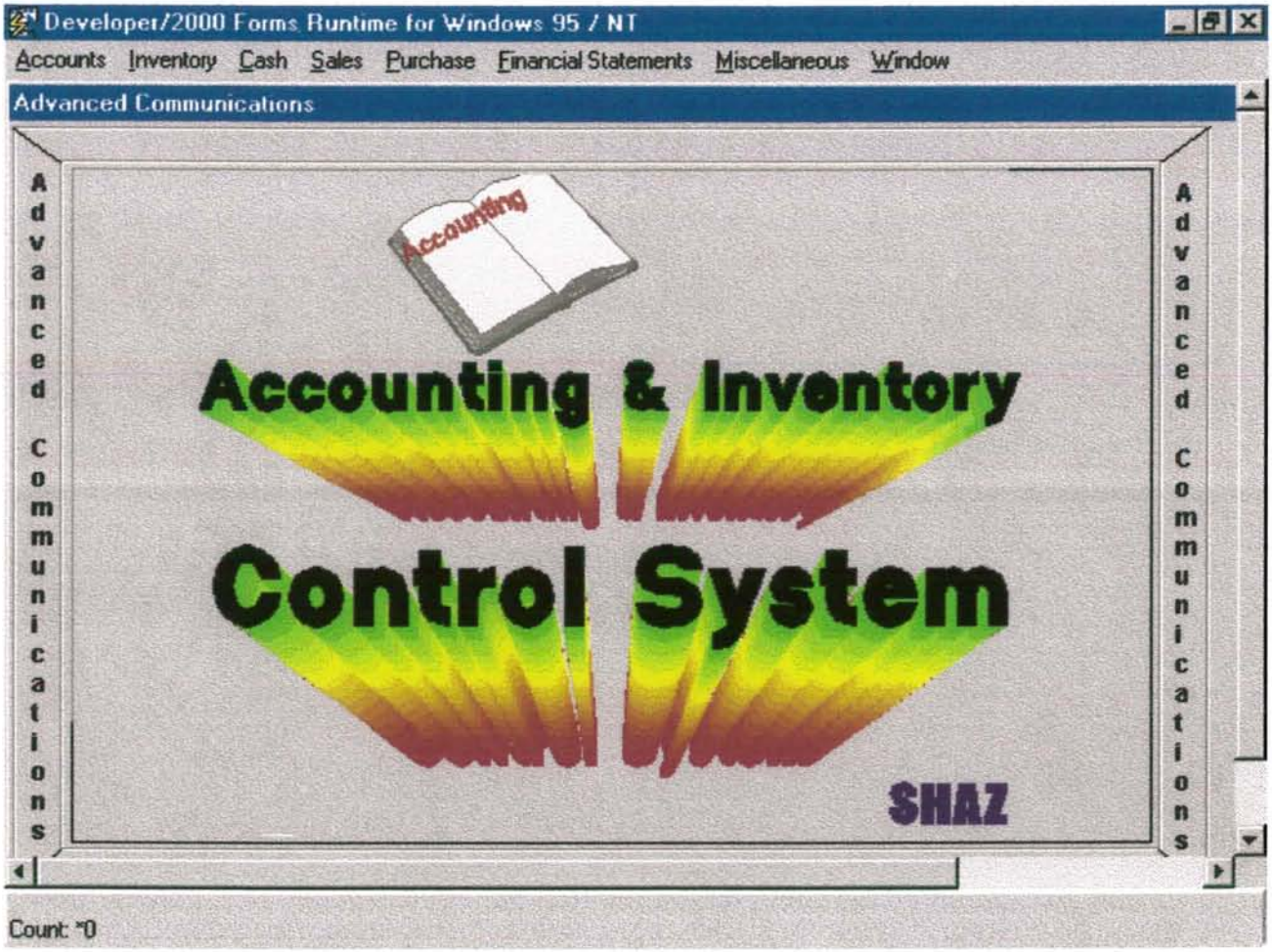
512


Developer/2000 Forms Runtime for Windows 95 / NT
Accounts Inventory Cash Sales Purchase Financial Statements Miscellaneous Window
Advanced Communications
Advanced Communications
Accounting
Accounting & Inventory
Control System
SHAZ
Advanced Communications
Count: *0

COM
511

IN THE NAME OF ALLAH, THE MOST GRACIOUS & COMPASSIONATE, THE MOST MERCIFUL AND BENEFICIENT, WHOSE HELP AND GUIDENCE ALWAYS SOLICIT AT EVERY STEP AND MOMENT

ACCOUNTING & INVENTORY CONTROL SYSTEM

BY

Amir Bakhsh

A report submitted to the Department of Computer Science, Quaid-e-Azam University Islamabad, as a partial fulfillment of the requirement for the award of the degree of MSc. In Computer Science

March, 98

DEPARTMENT OF COMPUTER SCIENCE
QUAID-I-AZAM UNIVERSITY
ISLAMABAD

Dated: May , 1998

FINAL APPROVAL

This is to certify that we have read the project report submitted by Mr. Amir Bakhsh and it is our judgment that this report is of sufficient standard to warrant its acceptance by the Quaid-i-Azam University, Islamabad for the degree of Master of Science in Computer Science.

COMMITTEE

1. **EXTERNAL EXAMINER**

   Dr. A. Faiz M. Ishaq
   Jaffar Brothers (Pvt) Ltd.
   26-D, 2nd Floor, Kashmir Road
   F-6/4, Blue Area
   Islamabad.

2. **SUPERVISOR**

   Mr. Iftikhar Azim Niaz
   Lecturer
   Deptt. Of Computer Science
   Quaid-i-Azam University
   Islamabad.

3. **CHAIRMAN**

   Dr. Saeed Akhtar Bhatti
   Deptt. Of Computer Science
   Quaid-i-Azam University
   Islamabad.

Project Brief

Project Title	Accounting & Inventory Control System
Organization	Advanced Communications Pvt.Ltd
Under Taken By	Amir Bakhsh
Supervised By	Mr.Iftikhar Azim Niaz (Q.A.U)
Starting Date	October, 1997
Date Of Completion	March, 1998
S/W Used	Oracle, Developer 2000
System Used	Pentium 100MH
Operating Used	Window 95/NT

Dedicated to

My affectionating Parents and Brothers Dr.Nazir Ahmed & Munir

Acknowledgments

I, at the first place, bow my head before the Allah Almighty who bestowed His countless blessings upon me, guided me towards the way of dignity, strengthened my mind and arms with spiritual and physical values, blessed me with the courage of facing problems and obstacles, enabled me to accomplish this project work.

I, from the core of my heart, pay special thanks to my sweet and loving parents, my Brothers Dr. Nazir Ahmad & Munir, my sisters and Bhabies who always felt prayers to Allah for my success, helped me all the time in all the ways to the extent that I could not find words to thank them. Really these are the people who are the most precious asset of my life and without their guidance and love I would have been laundering in the dark and dingy streets of the world.

Further more, I am thankful to Mr. Nauman Hashmi(MD. Advanced Communications) who from the very beginning to the end of the project assisted me with the useful and valuable guidelines and suggestions. He is the person whose help I would ever try to sought and wish to work with him.

I extend my thanks to my project supervisor Mr.Iftikhar Azim Niaz who helped me in understanding the basics of accounting that was the basic necessity of my work.

Amir Bakhsh

March 24, 1998

Q.A.U Islamabad

ABSTRACT

Any accounting transaction posted manually usually possesses the chances of errors, and, if such transaction is carried out at the beginning of the accounting period, then its effects on the final financial statements would be explosive. On the other hand, usage of electronic data processing equipment not only minimizes the chances of such errors but they are speedy and efficient as well. This "Accounting and Inventory Control System " has been engineered keeping this scenario in view. It provides an automation of the accounting system in which user needs only to enter a transaction only once and the remaining process including the production of the final financial statements is done automatically by the system .

Further more the system has modules for Sales, Purchase, Cash Receipt, Cash Payment transactions. An attempt has also been made to graphically visualize the financial statements and other reports to give a clue to upper management for the easy analysis of the accounting entities and transactions.

Preface

This report gives a comprehensive view of the system study, design and implementation phases of the project "Accounting and Inventory Control System". The entire work constitutes of five chapters followed by appendices and bibliography.

Chapter1 :
This chapter gives a brief introduction about the project, project initiation, and organization introduction.

Chapter 2:
Almost all the accounting terminology/concepts the implementation of which the system is composed of, are discussed in this chapter.

Chapter 3:
The techniques or methodology that are used while the designing of the system is carried out, are discussed in this chapter.

Chapter 4:
The detailed description of the process, hoe the system designing has been converted to an executable system.

Chapter 5:
How the system has been implemented and shortcoming of the new system are discussed in this chapter.

Table Of Contents

Chapter 1

Introduction

Chapter 1 Introduction

1.1 INTRODUCTION

Every business activity, whatever volume it possesses, maintains its accounting system, to observe each accounting transaction. This system not only provides a detailed information about each transaction that was, but the reports generated by this system assist the management to make crucial decisions for the betterment of the organization. This data and reports are also a valuable source of information for outsiders and government agencies that are interested in investing the company and intend to have a summarized vision about the financial strength and achievements of the company. A well formulated and free from faults and flaws accounting system smoothes the way for the outsiders to come and invest in the company.

1.2 INTRODUCTION OF ORGANIZATION

Advanced Communications being basically a merchandizing company was incorporated as a Limited Company in 1992. The major business the company deals in is the import of the computer equipment. The company maintains three offices one being in Lahore, other in Singapore and the last -the head office in Islamabad. Shipments from abroad are received either at Lahore or Islamabad office from where the sale is made. Within this limited life span, the company has been able to enroll its name in the list of the leading merchandizing companies of the country.

In addition to this, the company has recently launched a software development business at the head office, taking an appreciable step in the software development arena but this business is in its infancy.

Contacts

1) Lahore Office

Address 45 Hafeez centre gulberg Lahore
Ph.No 5763620
Fax 5751960
E-mail mtc@advcom.net

2) Islamabad Office

Address 55 Fazal Plaza Blue Area Islam Abad

Ph.No 822492

822493

E-mail sales@advcom.net

3) Singapore Office

Address 05-05 Simling Square 1 Roucher Canal Road Singapore

Ph.No 339057

Fax 3394447

E-mail bliss@advcom.net

4) Web

www.advcom.net

www.advsoft.net

1.3 SCOPE OF THE PROJECT

The project includes the development of the general purpose accounting software. The software will have modules for General Ledger, Accounts Receivable, Accounts Payable, Sales and Purchases. The inventory side will track all inventory movement , will keep track of movement in cost prices of different items. It will also include the Cost of Goods Sold for each sale invoice based on the current cost of the inventory. It will readjust the cost of its inventory with each purchase using average costing method.

Reports generated will include Balance Sheet, Profit and Loss Statement, Trial Balance, Current position of inventory at one time, Movement of particular inventory item with many others.

Accounts and Inventory modules will be totally integrated together and a single voucher will post its effect both in accounts and inventory.

Database used will be Oracle, development tool used will be Developer 2000. Operating system will be Microsoft Windows.

Chapter 2

A brief Introduction to Accounting

Chapter 2 A Brief Introduction to Accounting

2.1 Accounting

The art of recording, classifying, reporting, and interpreting the financial data of an organization is termed as accounting.

2.2 Forms Of Business Organization

1) Single Proprietorship

An unincorporated business owned by one person is called a single proprietorship.

Small retail stores and service enterprises are commonly operated as single proprietorship.

2) Partnership

When a business is owned by two or more people as partners, it is called a partnership.

3) Corporation

A separate legal entity incorporated under the laws of a state or the federal government is a corporation. The owners are called *stockholder.*

2.3 Some Accounting Terminology

Transaction

Any dealing between two persons or things is called a transaction. The transaction may be either a debt or credit transaction.

Voucher

Any written evidence in support of a transaction is a voucher.

Asset

Any valuable thing in possession of the organization and is expected to be beneficial in the future is asset.

Liability

Things that the party is to due is liability.

Capital

Assets that are invested by the party to start a business and the amount later invested to flourish the business.

Revenues

Revenues are inflows of cash or other properties received in exchange for goods or services provided to customers. Rents and interest earned are also Revenues.

Expenses

Expenses are goods and services consumed in operating a business or other economic.

Sales

Goods sold to customer are sales.

Sales Return

Any defective or unsatisfactory merchandize returned by the customer by whom it was purchased.

Sales Discount

Discount given to customers for early payment on sales made on credit.

Purchases

Goods purchased are called purchases.

Purchase Return

Merchandize purchased , if found defective can be returned back to vendors from whom it was purchased.

Purchase Discount

The selling company regards a cash discount as a sales discount and the buying company calls the same discount as the purchase discount.

Credits Terms

The terms of payment for sales or purchase of goods on credit. For example a credit term may be "2/10,n/60" means that the credit period is 60 days but the debtor may deduct 2% from the invoice amount if payment is made within 10 days after the invoice date.

Accounting Period

The division of the life of a business into time periods of equal lengths is called accounting period of any 12 consecutive months is known as a fiscal year.

2.5 Double Entry Theory

The need for equal amounts of debit and credits entries to record every business transaction is called double-entry system of accounting. Every transaction is recorded by equal amounts of debits and credits. This equality enables us to locate many types of errors which might be made while maintaining accounting records.

6) Prepare an after closing trial balance Prove that equality of debit and credit balances in the ledger has not been upset by the adjusting and closing procedure.

2.7 The Ledger

A business may use from two dozens to several thousand accounts in recording its transactions. Each account is placed on a separate page in a book or a loose-leaf book, or on a separate card in a tray of cards. If the accounts are kept in a book, the book is called a *ledger*. If they are kept on cards in a file tray, the tray of cards is a ledger.

2.7.1 General Ledger

Each entry of a voucher, is, against its appropriate account, placed in a ledger called *general ledger*. All financial statements are generated from this ledger.

2.6 The Accounting Cycle

Each accounting period in the life of a business is a recurring accounting cycle beginning with transactions recorded in a journal and ending with a post-closing trial balance. The steps in the order of their occurrence are as follows.

1) **Journalize transactions** Analyze business transactions as they occur and record them promptly in a journal.

2) **Post to ledger accounts** Transfer debits and credits from journal entries to ledger accounts.

3) **Prepare a worksheet** Begin with a trial balance of the ledger, enter all necessary adjustments , sort the adjusted account balances between income statement accounts and balance sheets accounts , and determine the net income or net loss.

4) **Prepare financial statements** Utilize the information in the work sheet to prepare an income statement , a statement of owner's equity , and balance sheet.

5) Adjust and close accounts Using information in the work sheet as a guide, enter the adjusting entries in the journal. Post these entries to ledger accounts. Prepare and post journal entries to close the revenue and expense accounts into the Income Summary account and to transfer the net income or net loss to the owner's capital account. Also prepare and post a journal entry to close the owner's drawing account into the owner's capital account.

6) **Prepare an after closing trial balance** Prove that equality of debit and credit balances in the ledger has not been upset by the adjusting and closing procedure.

2.7 The Ledger

A business may use from two dozens to several thousand accounts in recording its transactions. Each account is placed on a separate page in a book or a loose-leaf book, or on a separate card in a tray of cards. If the accounts are kept in a book, the book is called a *ledger.* If they are kept on cards in a file tray, the tray of cards is a ledger.

2.7.1 General Ledger

Each entry of a voucher, is, against its appropriate account, placed in a ledger called *general ledger.* All financial statements are generated for this ledger.

2.8 The Journal

It is possible to record transactions by entering debit and credit entries directly in

Consequently, to link the debits and credits of each transaction and to provide in one place a complete record of each transaction it is universal practice in pen-and-ink systems to record all transactions in an accounting record called the *Journal.* The debit and credit information about each transaction is then copied from the journal to the

2.7 The Ledger

A business may use from two dozens to several thousand accounts in recording its transactions. Each account is placed on a separate page in a book or a loose-leaf book , or on a separate card in a tray of cards. If the accounts are kept in a book, the book is called a *ledger*. If they are kept on cards in a file tray, the tray of cards is a ledger.

2.7.1 General Ledger

Each entry of a voucher, is , against its appropriate account, placed in a ledger called *general ledger*. All financial statements are generated for this ledger.

2.8 The Journal

It is possible to record transactions by entering debit and credit entries directly in the ledger. Consequently, to link the debits and credits of each transaction and to provide in one place a complete record of each transaction, it is universal practice in pen-and-ink systems to record all the transactions in an accounting record called *Journal*. The debit and credit information about each transaction is then copied from journal to the ledger accounts. The process of recording transactions in a journal is called Journalizing transactions. A journal is also called a book of original entry.

2.8.1 GENERAL JOURNAL

Many businesses maintain several types of journals. The nature of operations and the volume of transaction in the particular business determine the number and the type journal needed. The simplest form of journal is called a *general journal*.

Recording Transactions in a General Journal

To record transactions in a general journal the steps are :

1) The data is written in small figures at the top of the first column.

2) The names of the accounts to be debited and credited and an explanation of the transaction is written in the Account Titles and the Explanation column.

3) The debit amount is written in the debit column opposite the name of the account to be debited.

4) The credit amount is written in the credit column opposite the account to be credited.

5) A single line is skipped between each journal entry to set the entries apart

6) At the time transactions are recorded in the general journal, nothing is written in the *Posting Reference*. However, when the debits and credits are copied from the journal to the ledger, the account numbers of the ledger accounts to which the debits and credits are copied, are entered in this column.

Posting Transaction Information

The process of copying journal entry information from the journal to the ledger is called posting. The major steps are

For the debit:

1) Find in the ledger the account name in the debit of the entry.

2) Enter in the account the date of the entry the journal page number from which the entry is being posted, and in the debit column the debit amount.

3) Determine the effect of the debit on the account balance and enter the new balance.

4) Enter in the Posting Reference column of the journal the account number of the account to which the amount was posted.

For the credit:

Repeat the forgoing steps, with the exception that the credit amount is entered in the credit column and has a credit effect on the account balance.

2.8.2 Special Journals

To record all the transactions of a business in a General Journal is too much writing and too much labor is needed in posting the individual debits and credits. One way to reduce the writing and the posting labor is to divide the transaction of a business into groups of like transactions and to provide a separate journal - the special journal for recording transactions in each group. They are Sales Journal, Purchase Journal, Cash Receipt Journal, Cash Disbursements Journal.

2.8.2.1 Sales Journal

Only Sales on credit are entered in the *Sales Journal*. The information listed on the sales invoice usually includes the date of the sale, the serial no. of the invoice, the customer's name, the amount of the sale, and the credit terms.

The individual sales recorded in the journal are posted each day to the proper customer accounts in the Accounts Receivable Ledger. The check marks indicate that sales recorded in the journal were individually posted to the customer accounts. At the end of the month Sales Journal Account's amount is totaled and the total is debited to the Accounts Receivable and credited to Sales. The credit records the month's revenue from charge sales , and the debit records the resulting increase in the accounts receivable.

2.8.2.2 Purchase Journal

The handling of purchase transactions when a purchase journal is used follows a pattern quite similar to the one described for the sales journal.

The only transactions recorded in the purchase journal are purchases of merchandize on credit. If the merchandize is purchased for cash rather on credit, the transaction should be recorded in the cash payments journal. When assets other than merchandize are being purchased, if this purchase is made for cash then cash payments journal is used , if this purchase is made on credit then general journal is used. The individual amounts are credited to each customer's account. At the end of the month the purchase journal is totaled and posted as follows

1) As a debit to the purchase account.

2) As a credit to accounts payable controlling account.

2.9 Financial Statements

Among the most important and most widely used of all accounting reports are financial statements. Financial Statements are main source of financial information to persons outside the business organization and also are useful to management.

The basic purpose of financial statements is to assist decision makers in evaluating the financial strength, profitability, and future prospects of a business. Thus, managers, major customers, and labor all have a direct interest in these reports.

2.9.1 The Trial Balance

A trial balance is a two-column scheduling listing the names and balances of all the accounts in the order in which they appear in the ledger. A trial balance is prepared by

1) Determining the balance of each account in the ledger.

2) Listing the accounts having debit balance in one column and credit balance in other column.

3) Adding the debit balances

4) Adding the credit balances.

5) Comparing the sum of debit balances with the sum of the credit balances.

Advanced Communications.

Trial Balance

December 31, 19 ----

Cash	Rs. 22,500	
Accounts Receivable	9,500	
Land	130,000	
Building	36,000	
Office Equpiments	5,400	
Accounts Payable		RS. 23,400
Nauman' Capital		180,000
	203,400	203,400

The trial balance provides proof that the ledger is in balance . This agreement gives assurance that

1) Equal debits and credited have been recorded for all transactions.

2) The debit or credit balance of each account has been correctly computed.

3) The addition of the account balances in the trial balance has been correctly performed.

2.9.2 Balance Sheet

The purpose of the balance sheet is to show the financial position of business on a specific date. It is often called a position statement . Financial position is shown by listing the assets of the business, its liabilities and the equity of the owner or owners. The name of the business and the date are given in the balance sheet heading.

The fundamental characteristic of every balance sheet is that the total figure for assets always equals the total figure for liabilities and owner's equity. This agreement in accounting books is termed as accounting equation.

Mathematically

Assets = Liabilities + Owner's equity.

Assets		Liabilities & Owner's Equity	
Cash	Rs. 7,500	Liabilities:	
Noted Receivable	8,000	Notes Payable	Rs. 52,000
Accounts Receivable	57,000	Accounts Payable	15,000
Supplies	1,500	Salaries Payable	3,000
Land	40,000	Total Liabilities	70,000
Building	44,000	Owner's equity :	
Office Equipments	12,000	Nauman's Capital	100,000
Total	170,000	Total	170,000

2.10 Inventory Valuation Methods

The prices of many kinds of merchandize are subject to frequent change. Several assumptions are exercised in order to compute the cost of the units in the ending inventory and each of them leads to a different amount in the financial statements. These assumptions are

- Specific identification
- Average Cost
- First in First Out
- Last in First Out

2.10.1 Specific Identification

This method assigns actual purchase costs to the specific units purchased and has intuitive appeal. This method is best suited to inventories of high-priced, low-volume items. If the units in the ending inventory can be identified as coming from specific purchases, they may be priced at the amounts listed on the purchase invoices. This approach, however, does not always provide the most useful accounting information for a company handling a large volume of identical units.

2.10.2 Average Cost Method

Average cost is computed by dividing the total cost of the goods available for sale by the number of units available for sale. This computation gives a weighted-average unit cost. A common criticism of the average cost method of pricing inventory is that it attaches no more significance to current prices than to prices which prevailed several months earlier.

2.10.3 First-In First-out Method

The first-in first-out method, which is often referred to as FIFO, is based on the assumption that the first merchandize acquired is the first merchandize sold. In other words, each sale is made out of the oldest goods in stock; the ending inventory consists of the most recently acquired goods.

2.10.4 Last-In First-out Method

It suggests that most recently acquired purchased are sold first and the ending inventory consists old goods acquired in the earliest purchases. For measuring income, the flow of costs may be more significant than the physical flow of merchandize. By using

this method the income computed should be based upon the current market positions which is a strong logical argument to support the LIFO method.

Chapter 3

System Design

3.1 Expectations of the New System

This system, as has been already discussed, has been designed to convert the book and pen system of [illegible] company into a computer program. Usually a merchandizing company holds information for accounts, inventory, cash flow, sales and purchases. This system would be able to post all types of transactions, ledgers, and vouchers, which are a must to cope with the requirements of the company. The end user would need only to input the data, all insertions/updation to different ledgers/accounts would be carried out by the system thus minimizing the potential chances of making spell/value errors, to the maximum extent. The self-posting nature of the system would really relieve the burden of passing entries into the huge registers.

The system would inherit the traditional GUI nature from the recently developed S/Ws thus making the use of the system as easy as possible.

One basic requirement of accounting is the generation of reports at specific intervals of time during the current financial year. Keeping this basic requirement in view, the system would generate reports without any contribution by the user. The format or designing of such reports wold exactly the same as set by the accountants.

Again, to make the system easy to use and attractive to look at, online help wold be provided. The person with the sound accounting background but not much s/w knowledge would be able to run this system without much pain.

Again wrong data input by the user would be best possibly tried to be trapped by the system. As a response of which the system would display message boxes giving the possible reasons of such errors. This would surely minimize the chances of any wrong data to become the permanent part of the storage.

The system would be developed by integrating the different low-coupled and functionally isolated modules; each of them would be capable of performing a specific and unique function. Each module would be written in a way that any change made in it would not be reflected to any of the modules outside it. This would ease the task for those who intend to extend this system to accommodate the needed requirements.

3.2 System Design

The system design phase is on the highest priority among the phases on which the S/W engineers mostly emphasize. The more your design is precise and agrees with what is required, the more reliable and efficient the resulting product is expected to behave. Any fault or flaw left or ignored during this phase not only becomes problematic in the long run (the implementation phase) but also reduces the efficiency and reliability of the resulting S/W to a remarkable extent.

Firstly, during this phase, the s/w engineer must have to keep in mind all possible aspects of

inputs in a way that no input data is skipped. Secondly, all the outputs wanted by the user in one form or the other have to be encircled. This process of notifying all inputs and outputs, in S/W engineering books is termed as the logical design.

After the logical design gets completed along the true lines, the data item or the fields being closely related and they collectively, if grouped together, provide required information about a particular entity of the system, are grouped together. This integration of most related fields results a table. The thing that demands most concentration by the designer is that no input data is lost by the system and no redundant data is stored by the system. This precise distribution of data items among the different tables is termed as the physical design. The precision and accuracy of the physical design sits on top of the logical design and totally depends on the accuracy values of the logical design.

3.2.1 Input Design

Any data that the user wants to become a part of his permanent storage enters the boundary of inputs. Input may range from a single character field to a memo field. A well-engineered system is required to be capable of notifying that the user inserting or updating a field or a table possesses sufficient privileges to perform this operation. Moreover, it is the strong point of an efficient s/w that the data being entered is correct and free from errors.

Input design includes

- Input From Designing
- Validation Checks
- Code Designing

Inputs forms provided by the system are

- Account Master Form

  The basic purpose of this form is to store informations about each account that the company is concerned with. A unique code is assigned to each account because a code is easier to remember rather than the long names that carry the maximum probability of making spell mistakes. Against each account, its description, balance, type etc. are entered.

- Inventory Master Form

  This form serves the purpose of storing informations about each item of the Inventory that the party deals in. Similar to the account master form a unique item no is devoted to each item of the inventory. Other informations include item quantity, its cost and description etc.

- General Ledger Form

  To post a general ledger, a separate form has been designed. The major columns of this form are the debit column, the credit column. The system would never let the user to post this ledger until the sum of the debit column agrees with the sum of the credit column.

- Cash Receipts Journal

  All the transactions that involve the cash receipt are recorded using this form as the source of interacting with the tables. The account code column lists those accounts from which the cash is received.

- Cash Payment Journal

  This form is, to some extent, similar to the cash receipt journal. The difference lies on the fact that this form is used when the cash is paid to vendors or other account holders.

- Purchase Form

  A voucher is attached with all the purchases made by the company. This form is reserved to record this purchase voucher.

- Sales Form

  This form agrees very well to the previous form. As the name implies, sales made by the company are made a part of the storage using this form.

- Purchase Returns

  Almost each merchandizing company has the right of returning any purchased item that is found to be defective or unsatisfactory. This form as the input source records any item that does not satisfy the required level of quality, and is eventually returned back to the vendor, from whom it was purchased.

- Sales Returns

  The customer usually returns back any merchandize found defective. This sales returns voucher is posted using this form.

3.2.2 Codes

For the sake of easiness, to avoid confusion and complexity, it is an appreciable approach to make use of codes. This usage minimizes the chances of errors and makes the data easy to be processed. The system under study follows the same strategy to attain maximum efficiency. The major codes the system includes are

- Acct_code assigned to each individual account.
- Item_no assigned to each item in the inventory.
- Inv_no assigned to each invoice generated on when a sale made.
- Pod_no assigned to each purchase order when a purchase is made.

3.2.3 Validation Checks

A careless user can be expected to input error-proned data, which is difficult to correct if it has been stored by the system. A good designer places maximum concentration on the

possibility of such mistakes, and limits the user to input the data in the format being expected. The system has great potential of implementing such validation checks. E.g.

- Primary key of a table can't be left empty.
- Values of some fields are limited to a specific range.
- Some fields have mandatory nature and must be filled up.
- The data type of each field must agree with the data type of the value with which the filed is being filled up.

3.3 Output Design

The out put design of a system is a best way to, statistically and in a summarized way, represent the processed data. Obviously this data comes from what we say the inputs. Monitor screen or printed reports are some of the formats in which the output can be presented. User or system's requirements decide the data that is to be plotted, the format and the number of reports. Reports have vital significance for the accounting department of a company, for, these serve the purpose to visualize the inventory and financial position of the company. "Financial Statements", for example, are among the reports the outsiders and investors are always interested in. Some reports when designed in connection with accounting books, must have to blindly follow the format, the data columns and even the rules and regulations the accountants have set for. As a result, such reports demand the designer to have good accounting background and must be well aware of the crystal logic working behind the generation of such reports. This is basic key, particularly in accounting, towards the good output design.
Some of the reports the system is expected to generate are

- Trial Balance1
- Trial Balance2
- Balance Sheet
- Income Statement
- Chart of Accounts
- Inventory Value
- Profit item wise
- Inventory Movement

3.4 Physical Design

Physical design, sometimes pronounced as attribute analysis, is the stage of the s/w development life cycle, where sound and correct knowledge of database is a must. An aggregate of related data items or fields make up a record, which collectively become the entries of the table or file. Physical design basically demands the suitable distribution of fields among the different tables in way that must reduce the redundancy. Computer scientists, who have set rules for attribute analysis, have made standards in order to measure the correctness and the accuracy of the design. These standards can be called the Normal Forms. Any table that attribute analysis results must fall into any of these normal forms. A table being in the 1st normal form can be split up into two or more tables following the rules of data base design. The resulting tables are, and must be related through some attribute being common in the both the tables. The splitting up of a table into two or more tables is, in data base literature, termed as the Normalization of the table. A unique characteristic of the normalized table being, for example in the 3rd normal form, is that it always falls into its lower normal forms up to the 1st normal form.

Many and more chances of loosing data are potentially there during the normalization of the table if special care is not taken. Trying to reduce the redundancy of the data by normalization usually, on the other side looses some wanted data. So it is strongly recommended by the data base professionals that during attribute analysis both aspects must be kept in mind in parallel. Because of this dual concentration required by attribute analysis, this can be considered to be the toughest stage compared to other stages.

3.4.1 Attribute Analysis

(a) Table Name: Acct_mast

Primary Key: Acct_code

Description: To each party or person, the company deals with, a unique account no. is assigned. Moreover, for other entities e.g. Purchases or Sales, an account no. is devoted. The basic and unique purpose of this table is to record the details and balance of all these types of accounts.

(b) Table Name: Acct_tran

Description: Any transaction either a cash, sales or purchase, without any exception are recorded in this table. Information of this table indicates that at what date, against which account and to how many amounts that particular transaction was made. Any conflict or contrast, if arises between the expected and present results can be eliminated without much pain using this table's data.

(c) Table Name: Inv_mast

Primary key: Item_no

Description: Almost all merchandizing companies are mainly concerned with purchase or sale of items. To all the items that the companies in, a unique item no. is assigned. All purchases or sales are made against one or more items among *these items*.

So a separate table has been designed to keep record of all these items.

(d) Table Name: S_inv_h

Primary key: s_inv_no

Description: As the name says, this table is devoted for transactions that are related to sales. It is possible that at a particular date or against a particular invoices no. more than one entry of distinct items can be made. So to reduce redundancy sales transactions are posted to two to tables, one being master other the detailed. This table serves as the master table.

(e) Table Name: s_inv_d

Description: This is the detailed table connected to master table through s_inv_no. This table holds the information that what item, against which invoice no., in what quantity and at what price was sold.

(f) Table Name: p_ord_h

Primary key: p_ord_no

Description: This table in terms of its functionality and purpose resembles with the s_inv_h table. The prior is reserved for sale but the later is devoted to purchase only.

(g) Table Name: p_ord_d

Description: P_ord_d being the detailed table of the p_ord_h table does the same as the s_inv_d does. But the difference between these tables is again the same as between p_ord_h and s_inv_h tables.

(a)

Field Name	Description	Data Type	Constraint	Valid Values
Acct_code	Account code	Char(8)	Not Null	Numeric
Acct_desc	Account description	Char(25)		Alpha numeric
Gen_det	General or Detailed account	Char(1)	Not Null	G/D
Gen_acct	General account	Char(8)	Not Null	Numeric
Acct_level	Account level	Char(1)	Not Null	1..6
Acct_type	Account type	Char(10)	Not Null	Asset,Capital, Liability,Revenue,Expens e
Amnt_cur	Cur amount of the account	Number(12,2)		Numeric
Amnt_open	Opening amount of the account	Number(12,2)		Numeric
Amnt_typer	This period amount	Number(12,2)		Numeric

(b)

Field Name	Description	Data Type	Constraint	Valid Values
Ledger	Ledger type	Char(12)		Alphabet
Tran_date	Transaction date	Date	Not Null	Date
Tran_no	Transaction no	Number(4)		Numeric
Acct_code	Account code	Char(8)	Not Null	Numeric
Inv_no	Invoice no	Char(10)	Not Null	Numeric
Debt	Debit	Number(12,2)		Numeric
Credit	Credit	Number(12,2)		Numeric
Pod_no	Purchase order no	Number(10)	Not Null	Numeric

(c)

Field Name	Description	Data Type	Constraint	Valid Values
Item_no	Item code	Char(15)	Not Null	Numeric
Item_desc	Item description	Char(40)		Alpha numeric
Unit	Unit of the item	Char(10)		Alpha numeric
Total_qty	Quantity in stock	Number(6)	Not Null	Numeric
Price	Selling price	Number(12,2)		Numeric
Cost	Purchased price	Number(12,2)	Not Null	Numeric
Reord_qty	Reorder quantity	Number(6)		Numeric
Ven_code	Supplier code	Char(8)		Numeric
Open_stock	Opening stock	Number(6)		Numeric
Bad_stock	Items that get spoiled/lost	Number(6)		Numeric

Good_Stock	Items being in the satisfactory condition	Number(6)		Numeric
In_rma	Items that have been lent to some	Number(6)		Numeric
Out_rma	Items that have been borrowed	Number(6)		Numeric

(d)

Field Name	Description	Data Type	Constraint	Valid Values
Cus_code	Customer code	Char(8)		Numeric
Inv_date	Inventory date	Date		Date
Inv_no	Invoice date	Char(10)	Not Null	Numeric

(e)

Field Name	Description	Data Type	Constraint	Valid Values
Inv_no	Invoice no	Char(10)	Not Null	Numeric
Item_no	Item no	Char(15)	Not Null	Numeric
Price	Selling price	Number(12,2)		Numeric
Cost	Purchased price	Number(12,2)		Numeric
Qty	Quantity sold	Number(12,2)		Numeric
Item_ret	Item returned	Number(12,2)		Numeric

(f)

Field Name	Description	Data Type	Constraint	Valid Values
Pod_no	Purchase order no	Number(10)	Not Null	Numeric
Pod_date	Purchase order date	Date		Date
Bill_no	Bill no	Number(6)		Numeric
Acct_code	Account code	Char(8) code		Numeric

Field Name	Description	Data Type	Constraint	Valid Values
Pod_no	Purchase order no	Number(10)	Not Null	Numeric
Item_no	Item no	Char(15)	Not Null	Numeric
Price	Price	Number(12,2)	>0	Numeric
Cost		Number(12,2)	>0	Numeric
Qty		Number(12)	>0	Numeric
Ret_qty		Number(12)	>=0	Numeric

Chapter 4

System Developemt

4.1 Introduction

The aspects and factors, on which the S/W engineers, during the design phase emphasize, are different from that of the implementation phase. A solid and sound knowledge of the implementation phase and crystal programming skills are among the major factors that influence this phase. The tool the system is going to be implemented in, plays an important role towards the success of the new system. Possibilities often arise when the implementation language does not support a construct demanded by the designer. Eventually a good programmer seeks alternatives by making use of the provided constructs to get the wanted output. So it is required to have good knowledge about the implementation tool before selecting it. The tool must provide or offer the constructs to convert the design into an efficient executable S/W. Tool selection mainly depends upon the type of the work, you do. The task that mostly includes the processing of records (Insertion, Deletion ,Updation) is often implemented using one of the DBMS among the pool of such 4GLs, for, such tools have been engineered keeping objective in view and they are easy to run because of their simpler nature. Though such tasks can be implemented using some high level language but from the programmer this demand an extensive and painstaking job.

4.2 Why Oracle is selected as the Implementation Tool ?

The most important feature is that it is multi user software. The application developed in ORACLE can be connected together into a powerful, distributed database environment.

It provides a powerful client-server relationship between server and its terminals. In client-server relationship part of the processing is performed at the user's terminal thus a considerable increase in the speed of the processing.

ORACLE provides strict security of application developed in the package by enforcing the user name and password. Without the password it is not possible for any body to access the system. Also it is possible to grant different type of accesses to different user e.g. updation, addition or deletion rights may be provided only to the selected personnel while the rest may be allowed to view the records.

ORACLE provides a number of sophisticated tools for the development of application. Some of these tools are

1. SQL *Plus

SQL *Plus is an interface through which SQL commands may be entered and executed. There are number of SQL *Plus commands which can further process and format the output from SQL command, and provide facility for editing and saving SQL commands.

2. Forms Designer

Forms Designer provides facility to design forms. These forms can be used to fast and easy data entry, updation, deletion and queries to an ORACLE database.

3. Report Writer

3. Report Writer

Report Writer can be used to create an ordinary letter or tabular report. It can be used to produce a report derived from ORACLE table, with column heading, column of database information and total as desired.

A number of other utilities are also available which allow easy manipulation of data structure along with the data store in these structures. For example ORACLE provides export /import utilities with the help of it is possible to move structure along with data contained in these files from one system to another.

The ORACLE RBDMS is fully portable over 80 different H/W and operating system platform, including VMS, MVS, UNIX, MS-DOS, OS/2, Macintosh ORACLE 's unrolled portability and connectivity enables all the system in an organization to be linked in to single, integrated computing resource.

ORACLE provides a powerful procedural language extension to SQL known PL/SQL. PL/SQL significantly increase application performance and developer productivity, while enhancing the power and functionality of other ORACLE approaches.

4.3 How Development was Carried Out?

As the design is composed of input design and output design so is the implementation. The system under discussion has been developed following the famous proverb " Divide and Conquer" i.e. the whole system has been divided into modules each of them has been implemented separately and then integrated at the last. This strategy not only did make me easy but also reduced the probability of logical errors.

4.3.1 Accounts

A separate module has been devoted in order to register the accounts. The major inputs include

- Account Code
- General/Detail
- General Account
- Opening Balance

A unique name is required to be input for each account to be registered being either a general or detailed account. The general account is necessary to be already there I the data base. The system would internally assign level no and account type to each of the account. The level no of the account must be 1 greater than the level no of the general account of that account .The account and the general account must agree on their account type and the first digit of their codes.

Transactions:

(i) Current amount := Opening amount + Amount this period

(ii) Amount current of all the accounts being at the same level and having the same account type as of this account are sum up. This total amount is assigned to the

amount current of the general account of these accounts. The same procedure is repeated for the accounts being at the level as the general account and the type as of this account. This loop goes on until it reaches the level 1.

Checks Implemented:

(i) Account description would accept alphanumeric including spaces only.

(ii) The first digit of the account and of the general account must fall in the range from 1 to 6 and from 1 to 5 respectively.

(iii) If the trial is not balanced, the system would not let the open amount, this period amount and current amount to be input rather the system would assign '0' to these three data items.

4.3.2 Inventory

Similar to the accounts a similar module is designed to register the items of the inventory. The major inputs include

- Item no
- Quantity
- Selling price
- Cost

Transactions:

A new record is inserted with the unique item no.

Checks Implemented

(i) A unique numeric item no would be accepted only.

(ii) Item description is composed of string of alphanumeric and spaces only.

(iii) Selling price must be either greater or equal to the cost.

(iv) The formula "total quantity = open stock + good stock + bad stock + in rma – out rma " must be satisfied.

(v) Vendor code must be there in the database.

(vi) Insertion would only be allowed if the trial is not balanced.

4.3.3 Cash Payment

All the transactions that are related to cash payment are posted using a separate module "Cash Payment". The major inputs include the account no to whom, by how much amount, at what date and against which purchase order no the payment is made.

Transactions:

As the provision of multiple records is there so for each record

(i) A record is inserted into the acct_tran table

(ii) Account no is debited by corresponding "debt" amount, in the master file.

(iii) REVERSEUPDATE function is called.

Note: The description of this function is coming later in this chapter Note: The description of this func

Moreover the cash account is credited by the "total_debt" and REVERSEUPDATE is again called.

Checks Implemented:

(i) CV would be assigned by the system to voucher type and this is un-updateable. Similar is the case with the account description

(ii) Account code can only be inserted via the list of all accounts provided by the system

(iii) The system would not let any of the transaction to posted if the trial is not balanced.

4.3.4 Cash Receipt

All the transactions that are related to cash receipt are posted using a separate module "Cash Receipt". The major inputs include the account no, from which the payment is drawn, by how much amount, at what date and against which purchase order no.

Transactions:

As the provision of multiple records is there so for each record

(i) A record is inserted into the acct_tran table

(ii) Account no is credited by corresponding "credit" amount, in the master file.

(iii) REVERSEUPDATE function is called.

Moreover the cash account is debited by the "total_credit" and REVERSEUPDATE is again called.

Checks Implemented:

(i) CV would be assigned by the system to voucher type and this is un-updateable.

(ii) Account code can only be inserted via the list of all accounts provided by the system

(iii) The system would not let any of the transaction to posted if the trial is not balanced.

4.3.5 Journal Voucher

The transactions that are of miscellaneous type, according to accounting rules, usually need a separate book, filled with the entries of these transactions. Keeping this necessity in view a separate form has been designed for this purpose. The major inputs include

- Account code
- Debt
- Credit
- Transaction date

Transactions:

As the layout says, multiple records in are inserted in a single gallop. For each record

(i) A row is inserted into the acct_tran table

(ii) If the account no being either of the "Asset" or "Expense" type, is being debited, the amount current of this account would be increased by the "debt" amount.

(iii) If the account no being either of the "Capital" or "Liability" or "Revenue" type, is being debited, the amount current of this account would be decreased by the "debt" amount.

(iv) If the account no being either of the "Asset" or "Expense" type, is being credited, the amount current of this account would be decreased by the "credit" amount.

(v) If the account no being either of the "Capital" or "Liability" or "Revenue, is being credited the amount current of this account would be increased by the "credit" amount.

(vi) REVERSEUPDATE is called against each record for the corresponding account no.

Checks Implemented:

(i) JV would be assigned to "ledger type" and would be un-updatable.

(ii) Account code could only be selected from the list of the accounts provided by the system. This would certainly reduce the probability of the occurrence of the errors.

(iii) For each record either debt or credit would be allowed to be input but not both. Moreover both of them, for the same record, would never be allowed to be left empty.

(iv) The user would only be allowed to commit this form if "total debt" and "total credit" agree.

(v) Insertions would be successful only if the trial is balanced.

4.3.6 Sales Voucher

Sales is one of the major departments of any merchandizing company. The more this sales process is precise in its operational and computational aspects, the more the company is expected to earn revenue and strength its financial status. For each of the sale the company makes a voucher is prepared. Later on this voucher is posted which affect the accounts and inventory values. This system also maintains a separate module, which takes care of all the aspects. The major inputs include

- Party code
- Invoice no
- Transaction date
- Item no
- Quantity
- Selling price

Transactions:

(i) Four rows are inserted into the acct_tran table

(a) Sales account is credited by the "grand total" amount

(b) Party is debited by the " grand total" amount

(c) For each record, the cost of the corresponding item is picked up from the inv_mast table. This cost is multiplied by the quantity of that item. For all the records this multiplicand amount is sum up. At the last COGS and inventory accounts are debited and credited respectively.

(ii) Two rows are inserted in the acct_mast table

(a) Amount current of the party is increased by the "grand total" amount and REVERSEUPDATE is called.

(b) Sales account is increased by the "grand total" amount and REVERSEUPDATE is called.

(iii) for each record two rows of the acct_mast table and one row of the inv_mast table are updated

(a) COGS account is increased by the " quantity * cost of the item" amount

(b) Inventory account is decreased by the " quantity * cost of the item" amount

(c) The quantity of the corresponding item is decreased by the "quantity" amount in the inv_mast table.

Checks Implemented:

(i) Both the party-code and item no would be selected only from the provided respective lists.

(ii) A unique invoice no must be assigned whenever a new sales voucher is going to be posted.

(iii) The quantity of the item being sold must be either less or equal to the quantity of that item in the stock.

4.3.6 Purchase Order

Similar to the sales, a voucher is prepared each time when the company makes a purchase. Again similar to the sales voucher, when this purchase voucher is posted it would effect the current amount of the party from whom the purchase is made. Moreover the inv_mast table also require to be updated. The major inputs include

- Party code
- Purchase order no
- Purchase order date
- Item no
- Quantity
- Cost

Transactions:

(i) Two rows are inserted into the acct_tran table

(a) Purchase account is debited by the "grand total" amount

(b) Party is credited by the " grand total" amount

(ii) Two rows are updated in the acct_mast table

(a) Current amount of the party is decreased by the " grand total" amount and REVERSEUPDATE is called.

(b) Purchase account is increased by the "grand total" amount and REVERSEUPDATE is called.

(iii) For each individual record inv_mast is updated

(a) The total quantity of the corresponding item is increased by the "qty" amount

(b) The new cost of the corresponding item is calculated as

New cost := ((old cost * old quantity) +("qty" * "cost")) / (old quantity + "qty").

Checks Implemented:

(i) Party code and item no would be selected only from the provided respective lists.

(ii) The form can't be committed if the trial is not balanced.

(iii) A unique purchase order no is required to be input for each of the purchase voucher

4.3.7 Sales Return Voucher

The well-organized merchandizing companies usually give rights to their customers to return back any item found defective or damaged in any aspect. To record any item returned back to the company needs a voucher "Sales Return Voucher". This voucher, when posted, does revert the accounts and the inventory in a way as no such sale was never made. To revert the changes demands extensive care from the system. The major inputs include

- Invoice no.
- Items returned

Transactions:

(i) The user needs only to specify the invoice no against which the sale was originally made. The system would internally display all the details regarding the input invoice no.

(ii) If any item, against the specified invoice no, has already been returned, the system would prompt that particular item(s) against their respective quantities returned.

(iii) The system would also let the user know the maximum no. of items that can be returned, against each item no. This quantity would be the minimum of the quantity of that item in the stock and " quantity sold – quantity already returned ".

(iv) Four rows are inserted into the acct_tran table.

(a) Sales account is credited by the "grand total" amount.

(b) Party account is debited by the " grand total" amount.

(c) For all the records " items returned * cost of that item in the inv_mast table " is computed and then sum up. COGS account is debited by this amount while the inventory account is credited by this amount.

(v) Two rows are updated in the acct_mast table.

(a) Party account is increased by the " grand total" amount and REVERSEUPDATE is called.

(b) Sales account is also increased by the " grand total" amount and REVERSEUPDATE is called.

(vi) For each individual record

(a) The inventory account is decreased by the " cost of the item in the inv_mast table * items returned " amount

(b) COGS account is increased by the " cost of the item in the inv_mast table * items returned" amount

(c) The quantity of each item, in the inv_mast table is increased by the "qty " amount

Checks Implemented:

(i) Invoice no can only be selected from the provided list

(ii) Items returned must always be either less or equal to the " items to be returned".

4.4 Output

Any system developed in any environment usually demand the input data to be displayed in a summarized and informative style. This well formatted data boosts the upper management to a birds eye view of all the transactions and prevent them to deeply look at the individual transactions. The user and the system requirements usually decide the format and data columns to be displayed on a particular report from a pool of almost uncountable no. of rows. For the systems, particularly accounting system, these reports are much valuable as compared to other systems. The data columns and formats of some reports, generated by such systems, are almost the same. Other reports, in their style of representation, differ from system to system. The major reports the system would generate are :

4.4.1 Chart of Accounts

The chart of accounts listing provides a listing of all the accounts. The chart of accounts listing is stored by account no.

The columns included on the chart of accounts are :

- Account no
- Account description
- Account type
- Account level
- General/Detailed
- General Account

Query used :

```
Select Acct_code,Acct_desc,Acct_type,Acct_level,Gen_det,Gen_acct
    From Acct_mast
```

```
Order by acct_code.
```

4.4.2 Trial Balance 1

The Trial Balance lists balance information for the accounts in the chart of accounts. Included on the report are columns for Opening, This Period and Current Balances. This is used as a means to verify system balance. Total debits should equal to total credits at the bottom of the three columns. The total amounts represent the total of detail accounts only. The columns include are

- Account code
- Account description
- Opening amount
- This period amount
- Current amount
- Debt/Credit

If an account has negative balance, the balance prints in the opposite column. Negative values are never printed on the trial balance.

Query Used :

```
Select acct_code,acct_desc,amnt_open,amnt_typer,amnt_cur
   from acct_mast
where acct_code in('10000000','20000000','30000000','40000000','50000000')
order by acct_code.
```

4.4.3 Trial Balance 2

To display the current balance of the detailed accounts. The major columns include are

- Account code
- Account description
- Amount current
- Account type

Query used :

```
Select acct_type,acct_code,acct_desc,amnt_cur
    From  acct_mast
    Where gen_det = 'D'
    Group by acct_type
```

4.4.4 Profit item wise

To measure the profit earned by the items being in the inventory. The major columns included are

- Item no
- Quantity
- Price

- Cost
- Profit
- % profit
- Profit/unit
- Unit selling price
- Unit purchase price

Query used :

```
Select s_inv_d.item_no,sum(s_inv_d.qty),sum(s_inv_d.price),sum(inv_mast.cost)
    From s_inv_d,inv_mast
    Where s_inv_d.item_no = inv_mast.item_no
        group by s_inv_d.item_no.
```

4.4.5 Inventory Value

The basic purpose of this report is, as the name says, to give the summarized as well as the detailed information of the inventory value. The major columns included are

- Item no
- Item description
- Quantity
- Cost

Query used :

```
Select Acct_code,acct_desc,cost,total_qty from inv_mast
```

4.4.6 Income Statement

The income statement lists Revenue and Expense accounts and indicated the profit and loss for the period. The major columns included are

- Account code
- Account description
- Debt/Credit

Query used :

```
Select acct_code,acct_desc,amnt_cur
    From acct_mast
    Where acct_type = 'Revenue' and
    Gen_det = 'D'
Select acct_code,acct_desc,amnt_cur
    From acct_mast
    Where acct_type ='Expense' and
    Gen_det = 'D'.
Select amnt_cur from acct_mast
    Where acct_code = '51000000'
```

4.4.7 Balance Sheet

Balance sheet is used to verify the standard accounting equation(Assets = Liability + Capital). The first half of the Balance sheet lists Assets and the second half lists the Capital and Liability accounts. The major columns included are

- Account type
- Account code
- Amount current

Query used

```
select acct_code,acct_desc,amnt_cur from acct_mast
        where acct_type = 'Capital' and
        gen_det = 'D'
select acct_code,acct_desc,amnt_cur from acct_mast
        where acct_type = 'Liability' and
        gen_det = 'D'
select amnt_cur from acct_mast where acct_code ='11000000'
select acct_code,acct_desc,amnt_cur from acct_mast
        where acct_type = 'Asset' and
        gen_det = 'D'
```

4.5 Library used

For the sake of simplicity and to reduce the coding complexity, the most common and frequent used procedures/functions have been gathered in the form of a library "c:\shaz_lib\shaz_liv.pll". This is again a strong point that goes in the favor of the argument why Oracle has been selected as the implementation tool. Only one has to attach the library, which contains the compiled code, remaining task is done by Oracle. The modules are

4.5.1 REVERSEUPDATE()

As has been already stated, that, the time an account being at the lower level is updated, the accounts being at the upper level are also updated. The function accepts "acct_level", "gen_acct", "amount" as the input. The "amnt_cur" of all the accounts of level "acct_level" and of general account "gen_acct" is sum up and assigned to the "amnt_cur" of the account no "gen_acct". Again the "amnt_cur" of all the accounts being at the level as the "gen_acct" is, is sum up and assigned to the "amnt_cur" of their general account. This loop goes on , until level 1 is reached.

```
PROCEDURE ReverseUpdate(gen_acct in char,level_no in char,amount in number) IS
  general_acct  char(8)   DEFAULT gen_acct;
  account_level number    DEFAULT to_number(level_no);
  cur_level     char(8)   DEFAULT level_no;
  v_total_amnt  number;
  temp          char(8);
begin
```

```
while account_level >1 loop
 select sum(amnt_cur) into v_total_amnt
        from acct_mast
        where acct_mast.gen_acct = general_acct;
  update acct_mast set
        amnt_cur = v_total_amnt
    where acct_code = general_acct;
  select gen_acct into temp from acct_mast where
   acct_code = general_acct;
   general_acct := temp;
   account_level := account_level -1;
end loop;
end;
```

4.5.2 CHECKTRIAL()

This function is used to check, before any transaction is committed, whether the trial is balanced or not. This function sums up the balance of all the accounts of type "Asset" and "Expense". This summation is checked against the sum of the balances of the accounts being of type "Liability","Revenue" and "Capital". If the two results agree on their totals, it returns "true" else returns "false".

```
PROCEDURE checktrial(status out boolean) is
total_asset        number(12,2);
total_liability    number(12,2);
total_capital      number(12,2);
total_revenue      number(12,2);
total_expense      number(12,2);
inventory          number(12,2);
stock_inv          number(12,2);
total_debt         number(12,2);
total_credit       number(12,2);
is_balance         boolean default FALSE;
begin
select sum(amnt_cur) into total_asset
                   from acct_mast
                   where acct_type ='Asset';
 select sum(amnt_cur) into total_liability
                   from acct_mast
                   where acct_type ='Liability';
 select sum(amnt_cur) into total_capital
                   from acct_mast
```

```
                    where acct_type ='Capital';
select sum(amnt_cur) into total_revenue
                    from acct_mast
                    where acct_type ='Revenue';
select sum(amnt_cur) into total_expense
                    from acct_mast
                    where acct_type ='Expense';
total_debt   := NVL(total_asset,0) + NVL(total_expense,0);
total_credit := NVL(total_capital,0) + NVL(total_revenue,0) +  NVL(total_liability,0);
select amnt_cur into inventory from acct_mast
        where acct_code = '21000000';
        inventory := NVL(inventory,0);
select sum(cost * total_qty) into stock_inv
        from inv_mast;
        stock_inv := NVL(stock_inv,0);
if (total_debt = total_credit and total_debt <> 0) AND (inventory =  stock_inv) then
                    update data_path
                            set trial_balanced = 'TRUE'
                            where company = '1000';
                            status := TRUE;
else
        status := FALSE;
end if;
```

4.5.3 Average_Cost()

This function is to some extent more complex and is used specially in the purchase return module. It picks up all the rows from "p_ord_h" and "p_ord_d" tables that match with the provided "pod_no" and then the rows whose "item_no" matches with the "item_no" to be returned are selected and average cost is computed.

```
Procedure Average_Cost(p_pod_no in number,p_item_no in char,p_cost out number,
   p_qty in number)is
        s_cost          number(12,2) default 0;
        s_qty           number(12,2) default 0;
        v_pod_no        number(10) default 0;
        v_item_no       char(15);
        v_cost          number(12,2) default 0;
      v_qty           number(12,2) default 0;
      cursor p_r_cursor is
              select pod_no,item_no,cost,qty from p_ord_d;
```

```
begin
        open p_r_cursor;
        loop
        fetch p_r_cursor into v_pod_no,v_item_no,v_cost,v_qty;
        if v_item_no = p_item_no and v_pod_no = p_pod_no then
                if v_qty <> p_qty then
        s_cost:=((s_cost*s_qty)-(v_qty*v_cost))/
                (s_qty+v_qty- p_qty);
                  s_qty := s_qty + p_qty - v_qty;
                end if;
        else
                if v_item_no = p_item_no then
                        s_cost := ((s_cost * s_qty)+(v_qty * v_cost)) /
                    (s_qty + v_qty);
                        s_qty := s_qty + v_qty;
                end if;
        end if;
        exit when p_r_cursor%notfound;
        next_record;
end loop;
        close p_r_cursor;
        p_cost := s_cost;
end;
```

Chapter 5

Implementaion, Testing & Evaluation

5.1 User Interface

The fundamental feature of an well-engineered S/W is the measure of the ease the user finds, when he/she runs it. Attractive and meaningful interface raises its popularity among the users as it is easy to run, use, and handle. A new user, even having no basic knowledge about running S/Ws, can, without much pain, bring it home that what the system does, if the engineered S/W offers a good interface. " Any input/output screen should clearly indicate the purpose it serves and how this purpose can be successfully achieved" is the basic characteristic of a good interface. To achieve this goal the interface should inherit the characteristics such as; firstly the screen in front of the user should at the first appeal towards the purpose. Secondly the time when the user focuses on a particular item either by mouse or keys, an understandable message should appear, e.g. on the status bar, stating the length and the valid values it can accept. Thirdly any invalid or wrong input, if intentionally or unintentionally the user inputs, should be prompted by the system asking for valid values. Fourthly the screen designing and color(s) selection should be done in a way that the screen should catch the eye at its first look.

Best possible attempts have been made, to enroll the system we are discussing, in the long list of S/Ws that offer good GUI. Only the core module is menu-driven, other modules have push buttons having iconic nature. All the remaining modules have been integrated into the core module; each of them can be activated and executed by the appropriate selection from the menu-driven environment. The wanted item can be selected either by using mouse or keys. Each module when gets its job done, returns control to the core module, so as next selection, if required can be made.

The remaining, button-driven forms, basically are the input source for all the transactions the posting of which is supported by the system. The system during its execution span, places full hold on enabling or disabling a certain button depending on the condition. For example, when the form has been cleared, "save" option must be disabled as there are no changes to save, or, if the form has been committed then the attempt to recommit the form can be blocked if the "save" option is disabled after the form has been committed. This pendulum motion between enabling and disabling functional buttons according to what the common sense and environment say, limits the user to be valid all the times and save his/her time by avoiding mistakes. Moreover the data entry and length limit checks have been imposed on the items where necessary. To increase the readability of items, particularly of numeric type, format masks have been set appropriately both in the input and output layouts.

A good interface also demands the trapping of errors made by the user either at the time of input or when the form is being committed. Though Oracle itself takes care of all the constraints and displays messages if any constraint is being violated. But these messages, can't be of much value for an ordinary user other than the one having Oracle knowledge, for, they are not clearly defined and don't indicate towards the reasons of such errors. The system would, for the selected errors having maximum frequency of occurrence, get error code from Oracle and against each

error display the errors and reasons if known. This would certainly mark for the user the data or value that caused the error.

Simple but attractive input screens and output reports have been designed that resulted from a suitable combination of color(s) and font style and weight. The color selected to paint the screen is "gray" so my system in case of color resembles with the windows environment. The font used is "Arial" which I personally like to be suitable for an attractive display.

5.2 User Guide

5.2.1 Installation of the S/W

When the new system is delivered to the vendor(s), it is always required to let the new user know how the system would be installed. If this is, precisely and according to the prescribed directions, not done the running of the new system would be doubtful. Because the integration of different modules need to reside at the predefined paths.

Do always strictly follow the following steps, if you intend to install this system on your computer.

❶ Create the following folders on your hard disk

- ➲ c:\accounts
- ➲ c:\accounts\sforms
- ➲ c:\accounts\sreports
- ➲ c:\accounts\sgraphs
- ➲ c:\accounts\smaps
- ➲ c:\accounts\shaz_lib

❷ Copy *.fmx from your disk to the already created folder c:\accounts\sforms

❸ Copy *.rep from your disk to the folder c:\accounts\sreports

❹ Copy *.ogr from your disk to the folder c:\accounts\sgraphs

❺ Copy *.bmp from your disk to the folder c:\accounts\smaps

❻ Copy shaz_lib.pll from your disk to the folder c:\accounts\shaz_lib

5.2.2 Logging into the System

Turn on your computer, and get logged into windows. Find the "Personal Oracle for Window95" and click on the "Start Database", which in response, would start your Oracle Database. Now click "SQL 3.3" which is enclosed under "Oracle for Window95" menu item of your startup menu. This would ask you, your user name and password. After you have input valid input, it would take you to the "SQL" prompt. Now start "Forms Designer" under the menu item "Developer 2000 for Win95". From the file menu open "Accounts.fmb" . Again from the file menu select "Run" , this would run your system.

5.2.3 How to have "Access" to the wanted item!

As has already been stated, the main and core module is menu driven and you would always come across this screen whenever the system is run. Other wanted modules are executed from this screen. All the modules the system is composed of, have been, according to the nature of the task they perform, categorized among different menu items. The main heads are

① Accounts

② Inventory

③ Cash

④ Sales

⑤ Purchases

⑥ Financial Statements

⑦ Help

The main characteristic of this proper distribution is the naming which appeal towards their corresponding purposes.

5.2.3.1 Accounts:

This head when expanded would expose three items and a submenu. The menu items are

❶ Register Accounts

❷ Journal Voucher

❸ Quit

By clicking on any of these would take you to the corresponding modules with the exception of "Quit" which as the name says would close this system. The submenu named as "Reports" contains a report " chart of accounts" which can be run if clicked.

5.2.3.2 Inventory:

All the modules that are related to inventory have been grouped together into this head. This head is composed of a menu item and a submenu. The menu item "register items" if clicked would lead you to the register items module, if you intend to register a new item in your inventory. The submenu contains three reports

❶ Price list

❷ Profit item wise

❸ Inventory movement

A selection from this list of reports would run the corresponding report. This report when closed would bring you back to this main menu.

5.2.3.3 Cash

The name of this head appeals, that it would enclose the menu items closely related to cash transactions. This head contains two menu items and a submenu. The menu items are

❶ Cash Receipt

❷ Cash Payment

The submenu contains a report named "Cash flow" which represents the flow the cash, either received or paid, during the current accounting period.

5.2.3.4 Sales:

Basically two modules have been provided by the system, which are related to sales. These are

❶ Sales Voucher

❷ Sales Return

This head encloses I itself these two modules, each of them can be executed by selecting and clicking at it.

5.2.3.5 Purchases

A merchandizing company, in the normal condition, posts two transactions related to purchases. One, when purchases are made from the vendors and other when unsatisfactory items are returned back to the vendors. My system provides these two modules under the headings

- ❶ Purchases
- ❷ Purchase Return

To activate any of them from the main menu, they are provided under the head " Purchases".

5.2.3.6 Financial Statements:

Financial statements are the most important reports, usually all the leading merchandizing companies generate. The reports provided by the system for this purpose are

- ❶ Trial Balance1
- ❷ Trial Balance2
- ❸ Balance Sheet
- ❹ Income Statement

All these reports have been enclosed under the heading "Financial Statements" from this any of them can be run.

5.2.4 How a particular Transaction/Action can be carried out!

The user must decide the type of the transaction he intends to post and which module supports the posting of this transaction. How this module can be activated, has already been discussed. Almost all the forms have similar no. and type of push buttons with the exception of two modules. These buttons are

- ➲ New
- ➲ Save
- ➲ Clear
- ➲ Exit

When any of the form is run through the main menu, the focus is originally on the "new" item other buttons are disabled. When "new" is pressed the focus goes on to the first navigable item on the form and "save", "clear" gets enables. You can access a particular item on the form by either using the mouse or tab key. When you have completed input into the current focused item, press enter, the next navigable item comes under focus. Some of the fields can never be left empty, so when they get focus they would never let you go to other items if you are trying to leave this item empty. Moreover the system would never let you input characters into the fields having numeric data type. Similarly the description of items or accounts in the "register items" and "register items" modules respectively would accept only alphabets and spaces only, which is their basic demand. The format mask of some fields e.g. purchase order date, is fixed, so it is a must to enter the date in the predefined format. The length of items that stand as codes in some modules is fixed so you can never leave such items until the items get full length of input.

In case of forms which provide the facility of inserting multiple rows you could go to the next record after you have pressed enter at the last item of that row, but you wouldn't. Though this is against what the user excepts, yet imagine the scenario " You are at the last item of the row and you pressed "enter", the focus goes to the first navigable item of the next record which is the primary key. Also the last row was the last one you wanted to insert. Now when you are on the next record i.e. you have created a new record with its primary key being empty. When you commit this form, your try would not be successful as you are trying to insert a row with null primary key."

It is strongly recommended that press "enter" to go to the next item on the form. If you use mouse to move among different items, then there are chances that some calculations would not be performed by the system.

After you have input data in all the rows you want, press "save" to commit the form. If you were violating any imposed limitation, the system would prompt a meaningful message, just pick the reason of the violation you have made, correct the error and try again. If you have instead of "save" pressed "clear" even the form is teeming with unsaved data, the system would clear the form without asking you your willing. When you press "clear", "save" would get disabled and "exit" is enabled. Again press "new" if you still intend to insert more records.

The execution of report modules is simpler as compared to form modules. As has already discussed that which report does reside under which menu head. Just click the report name you intend to activate, the corresponding report would start running. It would at the middle prompt you the parameter value, but this would be the case with just a few reports. For other reports just press "Run Report". For reports which require the parameter value the default value is already, if you intend to change this default value, just select the value from the provided list of values. When the run-form of the report is in front of you, you can print it, create a new of it and close it by using the button palette being at the top of the report.

5.3 Testing Phase

Almost all the systems whatever amount of work has been put into during its development, always potentially has the chances of making runtime and logical errors, so the last and most important phase, almost all the good systems have to go through, is the "Testing Phase". It is basically the process to functionally evaluate the correctness and the precision of the functions offered by the system. It also validates the fact that each of the transaction when posted produces the same result as is being expected from. A standard testing procedure requires that the system must be tested even at the individual transaction level giving both the valid and the invalid inputs and then the exact agreement between the output and the expected one must be identified. The reliability and efficiency of the newly developed systems mostly depend upon the measure of the work put into, and the testing technique used during this phase. A well tested and free from bugs systems get appreciation from its users which, alternatively saying, is the success of the developers.

press "Run Report". For reports which require the parameter value the default value is already, if you intend to change this default value, just select the value from the provided list of values. When the run-form of the report is in front of you, you can print it, create a new of it and close it by using the button palette being at the top of the report.

5.3 Testing Phase

Almost all the systems whatever amount of work has been put into during its development, always potentially has the chances of making runtime and logical errors, so the last and most important phase, almost all the good systems have to go through, is the "Testing Phase". It is basically the process to functionally evaluate the correctness and the precision of the functions offered by the system. It also validates the fact that each of the transaction when posted produces the same result as is being expected from. A standard testing procedure requires that the system must be tested even at the individual transaction level giving both the valid and the invalid inputs and then the exact agreement between the output and the expected one must be identified. The reliability and efficiency of the newly developed systems mostly depend upon the measure of the work put into, and the testing technique used during this phase. A well tested and free from bugs systems get appreciation from its users which, alternatively saying, is the success of the developers.

Many different testing techniques are in practice, e.g. Black Box Testing, each of them having a different view and criteria towards testing. The technique, I have exercised, is " Unit Testing". The first and the strong argue in support of this selection is the criteria of this technique which requires that each module must be tested individually and verified its validity and then integrated later on. If a module has been individually tested, when integrated with other modules, if any error is found. This indicates that the cause of the error lies not in the module that was integrated but in the module with which the integration was made. Surely this eases the job of finding errors.

The testing and the test data of each individual module are as

5.3.1 Register Accounts(Before Transactions Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	0
21000000	Inventory	Asset	2	20000000	0

Transaction1 :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
23000000	Computer	Asset	3	21000000	15,000

After Transaction1(Acct_mast Position):

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	15,000
21000000	Inventory	Asset	2	20000000	15,000
23000000	Computer	Asset	3	21000000	15,000

Transaction 2 :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
24000000	Printers	Asset	3	21000000	12,000

After Transaction2(Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	27,000
21000000	Inventory	Asset	2	20000000	27,000
24000000	Printers	Asset	3	21000000	12,000

Transaction3:

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
25000000	Building	Asset	2	20000000	250,000

After Transaction3(Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	277,000
25000000	Building	Asset	2	20000000	250,000

5.3.2 Register Items :

Transaction1:

Item_code	Item_desc	Quantity	Price	Cost	Open_Stock
222222222222222	Computers	50	200	100	50
555555555555555	Accessories	60	250	200	60

After Transaction1(Inv_mast Position):

Item_code	Item_desc	Quantity	Price	Cost	Open_Stock
222222222222222	Computers	50	200	100	50
555555555555555	Accessories	60	250	200	60

5.3.3 Cash Receipt(Before Transaction Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	277,000
22000000	Cash account	Asset	2	20000000	0
27000000	Nauman account	Asset	2	20000000	0

Transaction 1:

Acct_code	Debt	Inv_no	Date	Ledger_type
27000000	1200	345235235	28-Mar-98	RV

After Transaction1(Acct_mast position):

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	27,000
22000000	Cash account	Asset	2	20000000	1200
27000000	Nauman account	Asset	3	21000000	-1200

After Transaction1(Acct_tran position):

Ledger_type	Acct_code	Inv_no	Debt	Credit	Pod_no
RV	22000000		1200		345235235
RV	27000000	345235235		1200	

5.3.4 Cash Payment(Before Transaction Acct_mast position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	277,000
22000000	Cash account	Asset	2	20000000	0
27000000	Nauman account	Asset	2	20000000	0

Transaction 1:

Acct_code	Credit	Pod_no	Date	Ledger_type
27000000	1200	345235235	28-Mar-98	RV

After Transaction1(Acct_mast position):

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	27,000
22000000	Cash account	Asset	2	20000000	-1200
27000000	Nauman account	Asset	3	21000000	1200

After Transaction1(Acct_tran position):

Ledger_type	Acct_code	Inv_no	Debt	Credit	Pod_no
RV	22000000	345235235		1200	
RV	27000000		1200		345235235

5.3.5 Journal Voucher(Before Transaction Acct_mast position)

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
26000000	Shaz account	Asset	2	20000000	1900
20000000	Asset account	Asset	1	20000000	259,400
27000000	Nauman account	Asset	2	20000000	-17,200

Transaction 1:

Acct.code	Acct_desc	Debt	Credit
26000000	Shaz account		200
27000000	Nauman account	200	

After Transaction1(Acct_mast position):

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
20000000	Asset Account	Asset	1	0	259,400
26000000	Shaz account	Asset	2	20000000	1700
27000000	Nauman account	Asset	2	20000000	-17000

After Transaction1(Acct_tran position):

Ledger_type	Acct_code	Inv_no	Debt	Credit	Pod_no
JV	26000000	345235235		200	
JV	27000000		200		345235235

5.3.6 Sales Voucher(Before Transaction Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
21000000	Inventory account	Asset	2	20000000	27000
20000000	Asset account	Asset	1	20000000	259,400
27000000	Nauman account	Asset	2	20000000	-17,000
40000000	Revenue Account	Revenue	1	0	0
41000000	Sales account	Revenue	2	40000000	0
51000000	COGS account	Expense	2	50000000	0
50000000	Expense account	Expense	1	0	0

Before Transaction1(Inv_mast Position):

Item_code	Item_desc	Quantity	Price	Cost	Open_Stock
222222222222222	Computers	50	200	100	50
555555555555555	Accessories	60	250	200	60

Transaction1:

Party Account : 27000000
Date : 29-Mar-98
Inv_no : 111111111

Item code	Item desc	Quantity	Unit Price
222222222222222	Computer	10	200
555555555555555	Accessories	15	250

(After Transaction1 Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
21000000	Inventory account	Asset	2	20000000	21,150
20000000	Asset account	Asset	1	0	230,000
27000000	Nauman account	Asset	2	20000000	-11,250
40000000	Revenue Account	Revenue	1	0	5,750
41000000	Sales account	Revenue	2	40000000	5,750
51000000	COGS account	Expense	2	50000000	4,000
50000000	Expense account	Expense	1	0	4,000

After Transaction1(Inv_mast Position):

Item_code	Item_desc	Quantity	Price	Cost	Open_Stock
222222222222222	Computers	40	200	100	50
555555555555555	Accessories	45	250	200	60

After Transaction1(Acct_tran Position):

Ledger_type	Date	Acct_code	Inv_no	Debt	Credit	Pod_no
SV	29-MAR-98	41000000	1111111		5750	
SV	29-MAR-98	27000000	1111111	5750		
SV	29-MAR-98	51000000	1111111	4000		
SV	29-MAR-98	21000000	1111111		4000	

After Transaction1(S_inv_h Position):

Party Account : 27000000
Date : 29-Mar-98
Inv_no : 111111111

After Transaction1(S_inv_d Position):

INV_NO	ITEM_NO	PRICE	QTY
1111111	222222222222222	200	10
1111111	555555555555555	250	15

5.3.7 Purchase Voucher(Before Transaction Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
21000000	Inventory account	Asset	2	20000000	23000
20000000	Asset account	Asset	1	0	261,150
26000000	Shaz account	Asset	2	20000000	1700
50000000	Expense Account	Expense	1	0	4000
52000000	Purchase account	Expense	2	50000000	0

Before Transaction1(Inv_mast Position):

Item_code	Item_desc	Quantity	Price	Cost	Open_Stock
222222222222222	Computers	40	200	100	50
555555555555555	Accessories	45	250	200	60

Transaction1:

Party Account : 27000000
Date : 29-Mar-98
Inv_no : 98090080

Item code	Item desc	Quantity	Unit Price
222222222222222	Computer	20	150
55555555555555	Accessories	15	230

(After Transaction1 Acct_mast Position) :

Acct.code	Acct_desc	Acct_type	Acct_level	Gen_acct	Amnt_cur
21000000	Inventory account	Asset	2	20000000	23,000
20000000	Asset account	Asset	1		261,150
26000000	Shaz account	Asset	2	20000000	-4,750
51000000	Expense account	Expense	1	0	10,450
52000000	Purchase account	Expense	2	50000000	6,450

After Transaction1(Inv_mast Position):

Item_code	Item_desc	Quantity	Price	Cost	Open_Stock
222222222222222	Computers	60	200	116.67	50
555555555555555	Accessories	60	250	200	207.5

After Transaction1(Acct_tran Position):

Ledger_type	Date	Acct_code	Inv_no	Debt	Credit	Pod_no
PV	29-MAR-98	52000000		6450		980980080
PV	29-MAR-98	26000000			6450	980980080

After Transaction1(P_ord_h Position):

Party Account : 26000000
Date : 29-Mar-98
Inv_no : 980980080

After Transaction1(P_ord_d Position):

Pod_NO	ITEM_NO	PRICE	QTY
980980080	222222222222222	150	20
980980080	555555555555555	230	15

5.4 Evaluation

Some of the major features/services, if rules of justification are followed, would surely earn praise and appreciation for this system. As this system, has been developed in accordance with the steps the s/w engineers suggest, and inherits some features from the newly developed systems, so it has close resemblance with such systems. Some of the major features are

- An attractive and eye catching interface
- Automation of the posting transactions
- Automation of the generation of selected reports
- Errors and bugs free
- Ease of its usage

The major feature, according to my best known knowledge, is that it is functionally accurate and precise to the maximum extent.

On the other hand the system lacks some of the facilities if provided would certainly move the revised system towards its absoluteness. Some are

- The provision of the backup & recovery facility
- The rolling back a transaction

(Note : I have tried but failed to facilitate the system with this feature(Rolling Back).I hope that I would implement this provision if feasible and possible).

5.4.1 Some Limitations imposed by the System

For the easy system development, it was a must to fix some account codes for some accounts and let not the user to use them for later registered accounts. The user may feel uncomfortable to be restricted of not using some particular values but no alternative other than this was available. This would make as least as possible the occurrence of the runtime errors. In case of any complexity that, if arises in a rare case, the user is recommended to view the "Chart of Accounts" report. It is expected that the user would get out of this trouble if any. The details are as

Acct code	Acct desc	G/D	Gen Acct	Acct type	Acct level	Open amnt	Typer amnt	Currt Amnt
		G		Capital	1	0		
		G		Asset	1	0		
		G		Liability	1	0		
		G		Revenue	1	0		
		G		Expense	1	0		
		D	10000000	Capital	2	0		
		G	20000000	Asset	2	0		
		G	20000000	Asset	2	0		
		G	40000000	Revenue	2	0		
		G	40000000	Revenue	2	0		
		G	50000000	Expense	2	0		
		G	50000000	Expense	2	0		
		G	50000000	Expense	2	0		

Bibliography

Michael Abbey, Michael Corey. Oracle: A Beginner Guide

Robert J.Muller. Oracle: Developer/2000 Handbook

Scott Urman. Oracle: PL/SQL Programming

Oracle: Forms Reference Manual Vol. 1, Vol. 2

Release 4.5 Part No. A32509-2

Oracle: Building Reports Manual

Release 2.5 Part No. A32488-1

Working with Windows and Canvas Views

Chapter 11 (Forms Developer Guide)

Tutorial Chapter 2 (Graphics Developer Guide)

Built-in Subprograms Chapter 4 (Graphics Developer Guide)

Using Charts Chapter 8 (Graphics Developer Guide)

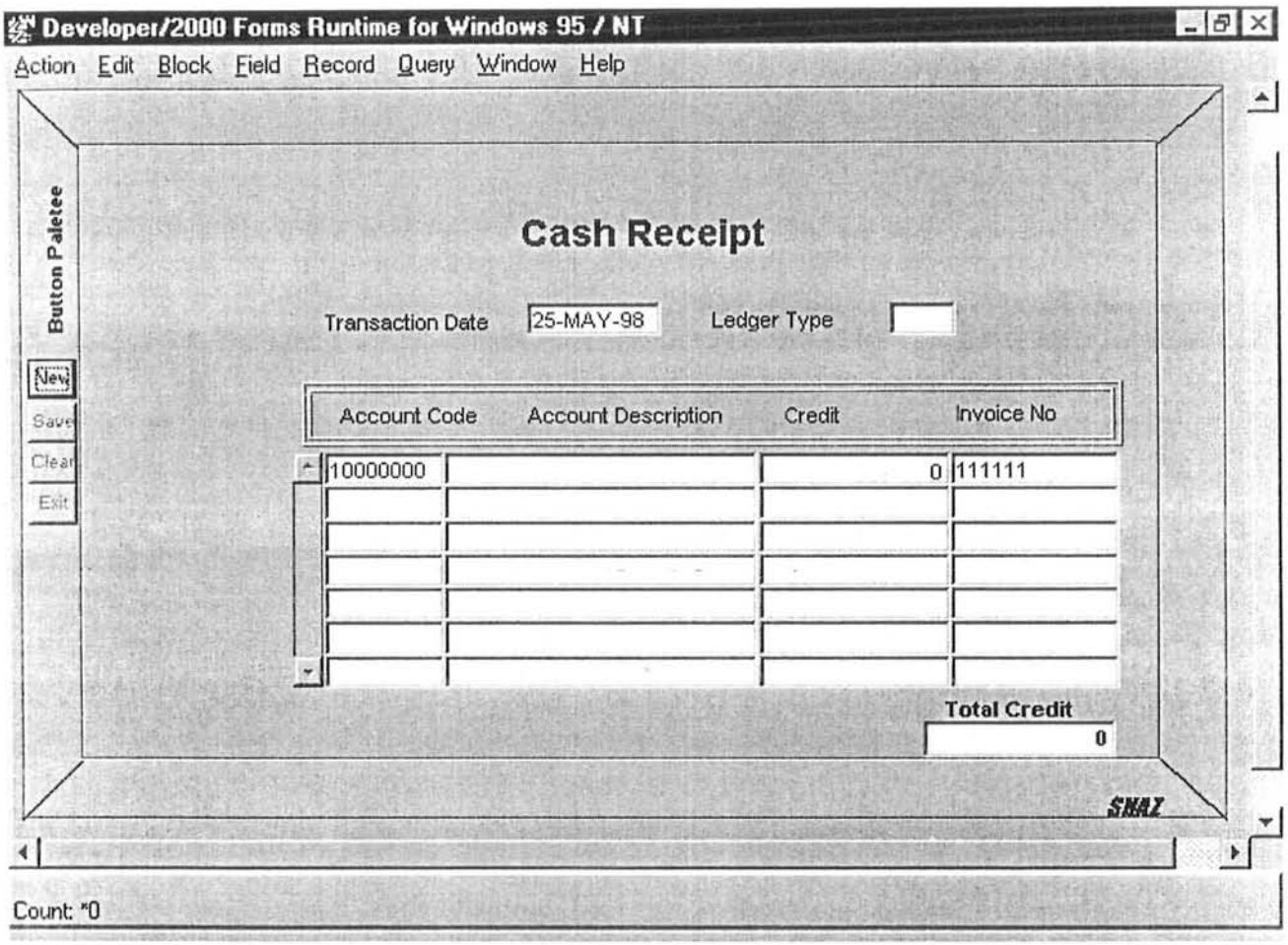
Using Parameters Chapter 10 (Graphics Developer Guide)

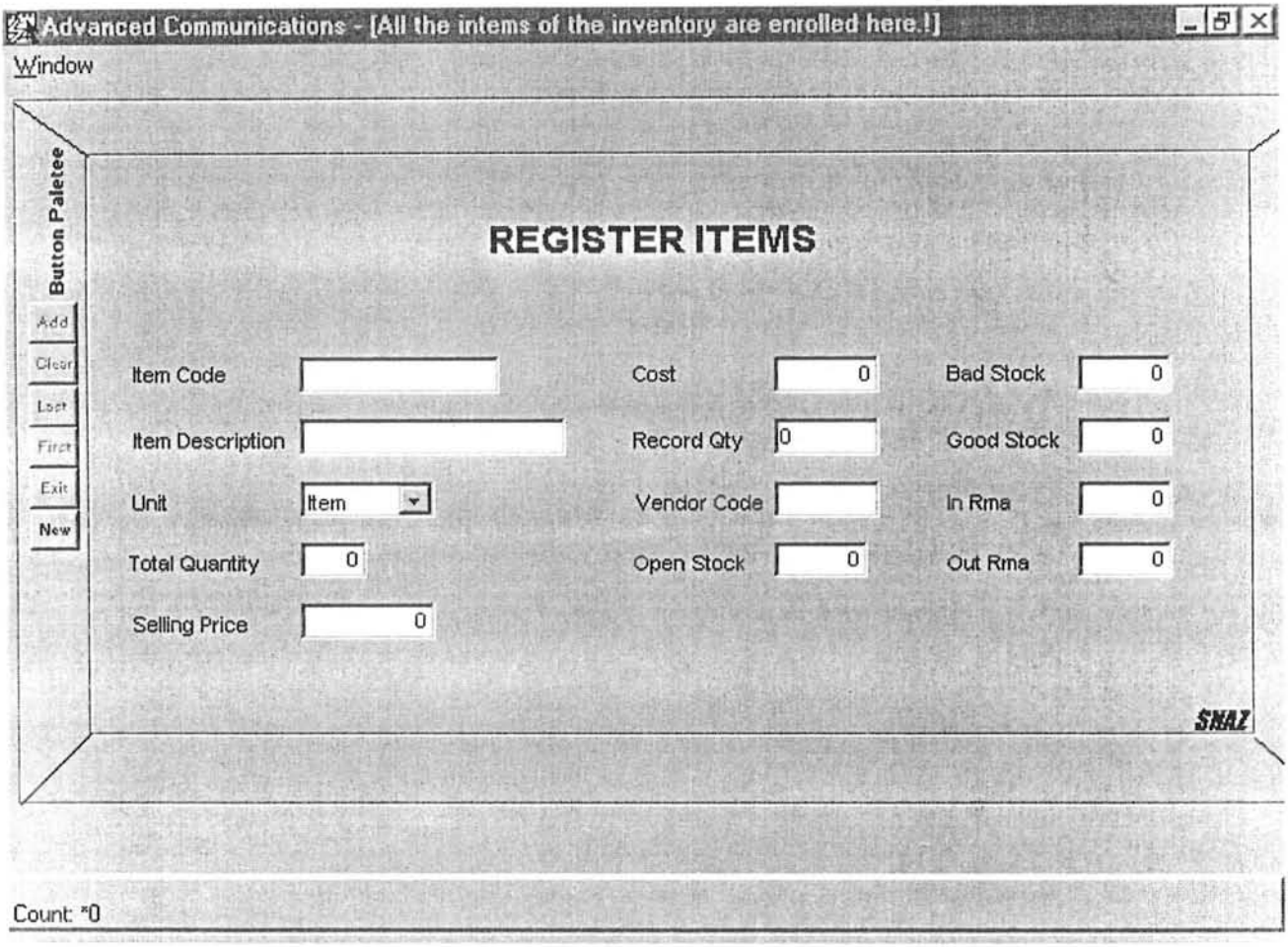
Integrating Displays Chapter 16 (Graphics Developer Guide)

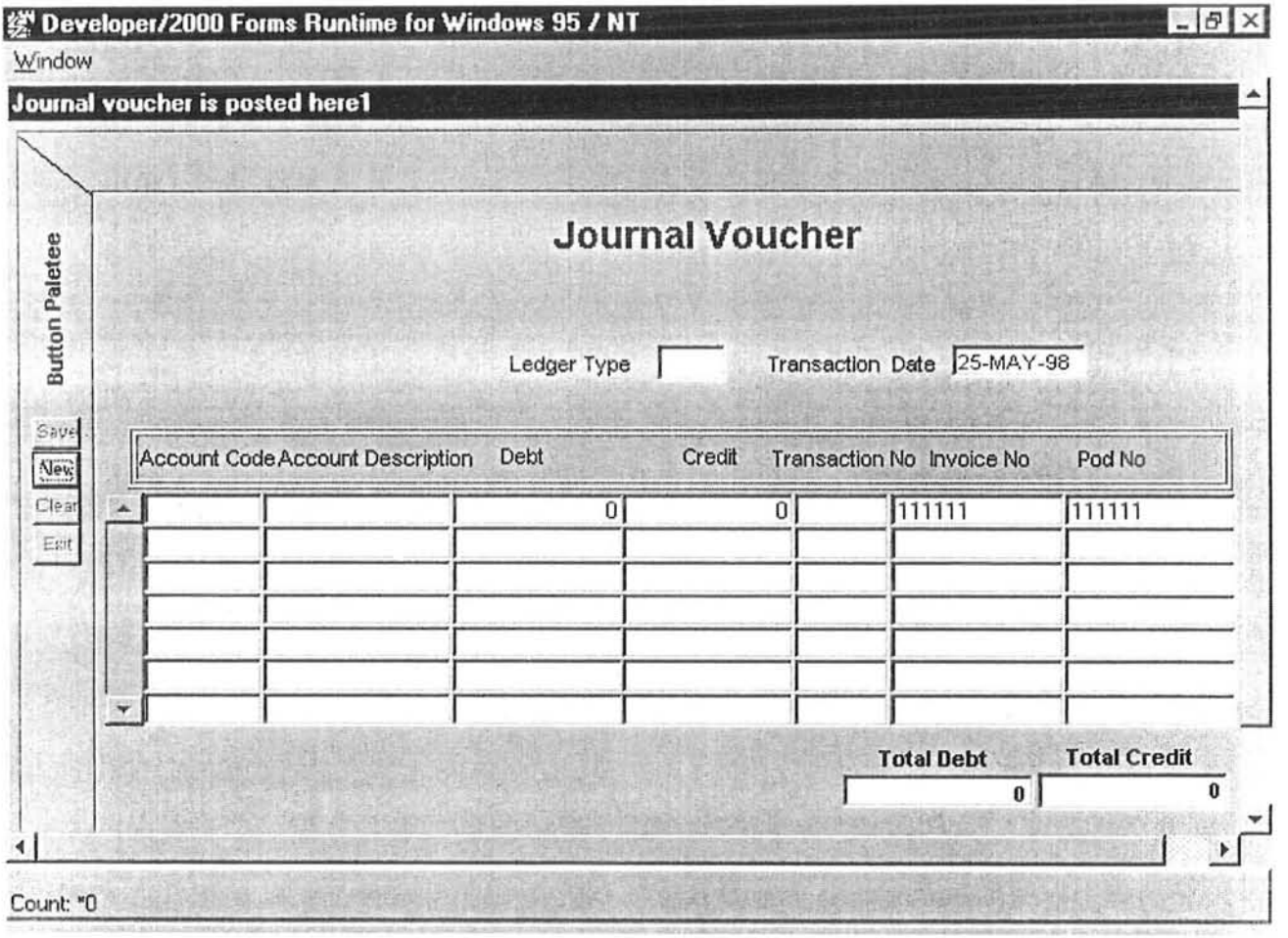
Walter B.Meigs, Robert F. Meigs . Accounting The Basis for

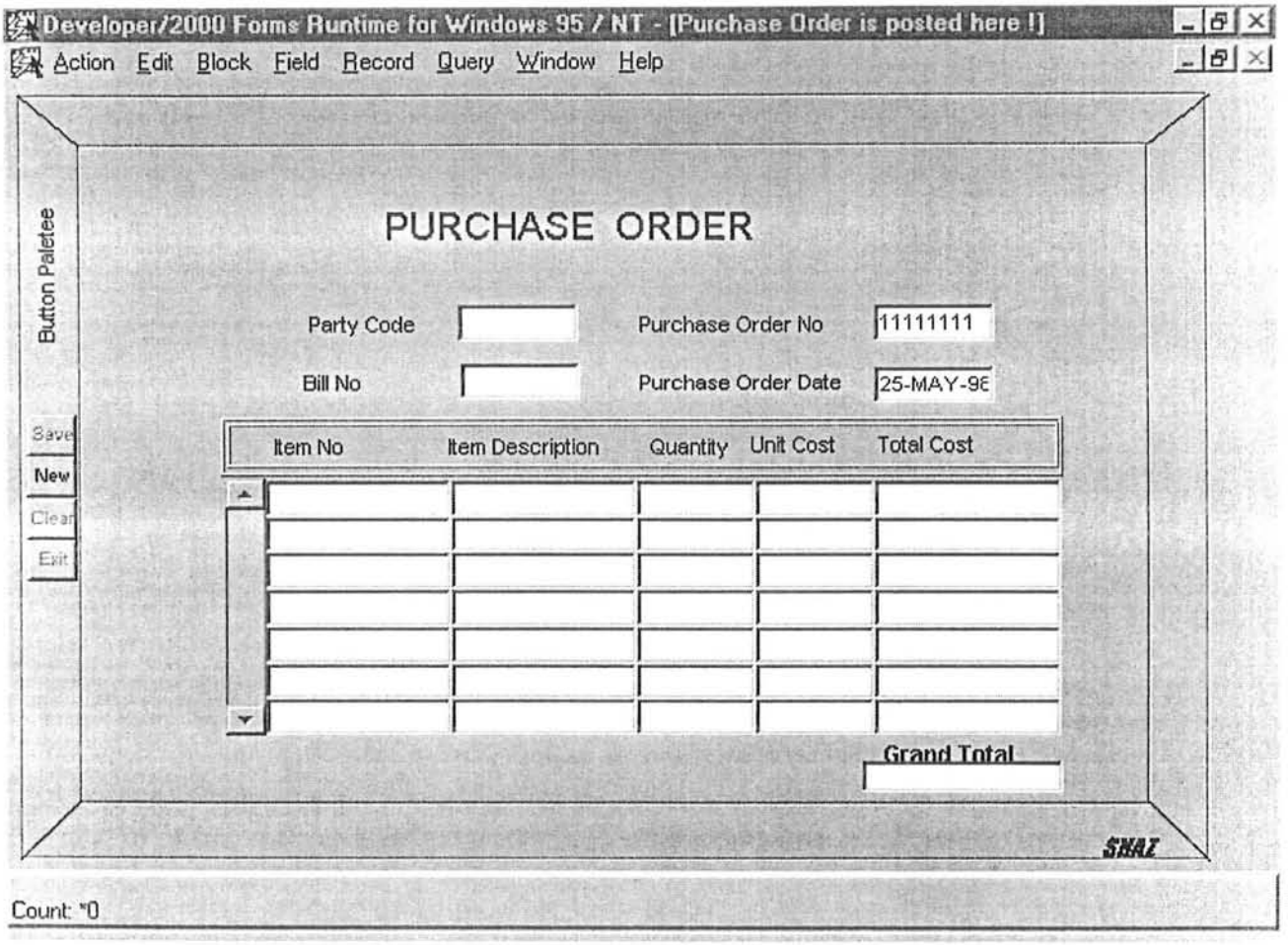
Business Decisions 7th edition

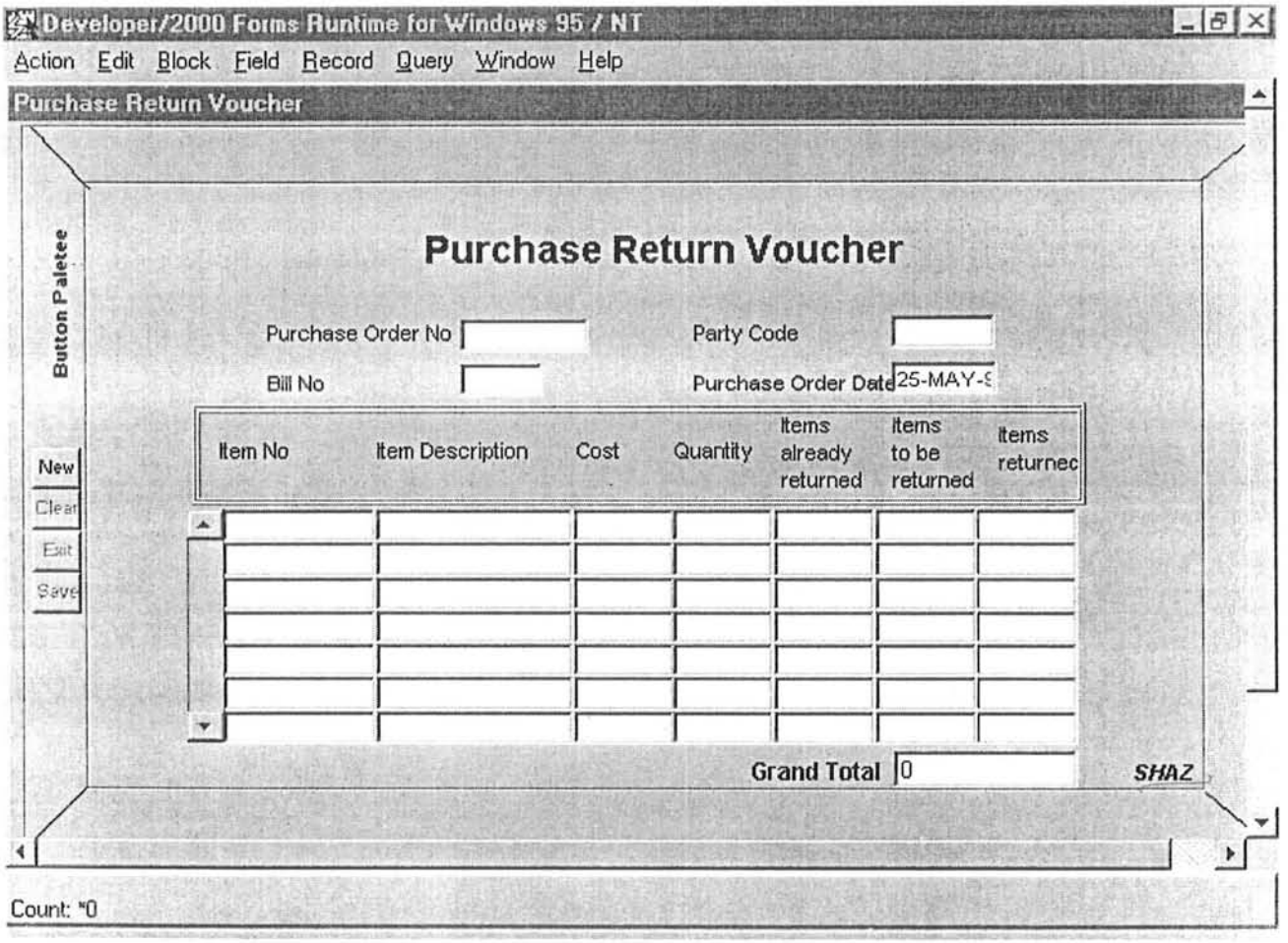
DecEasy Accounting User's Guide version 4.2

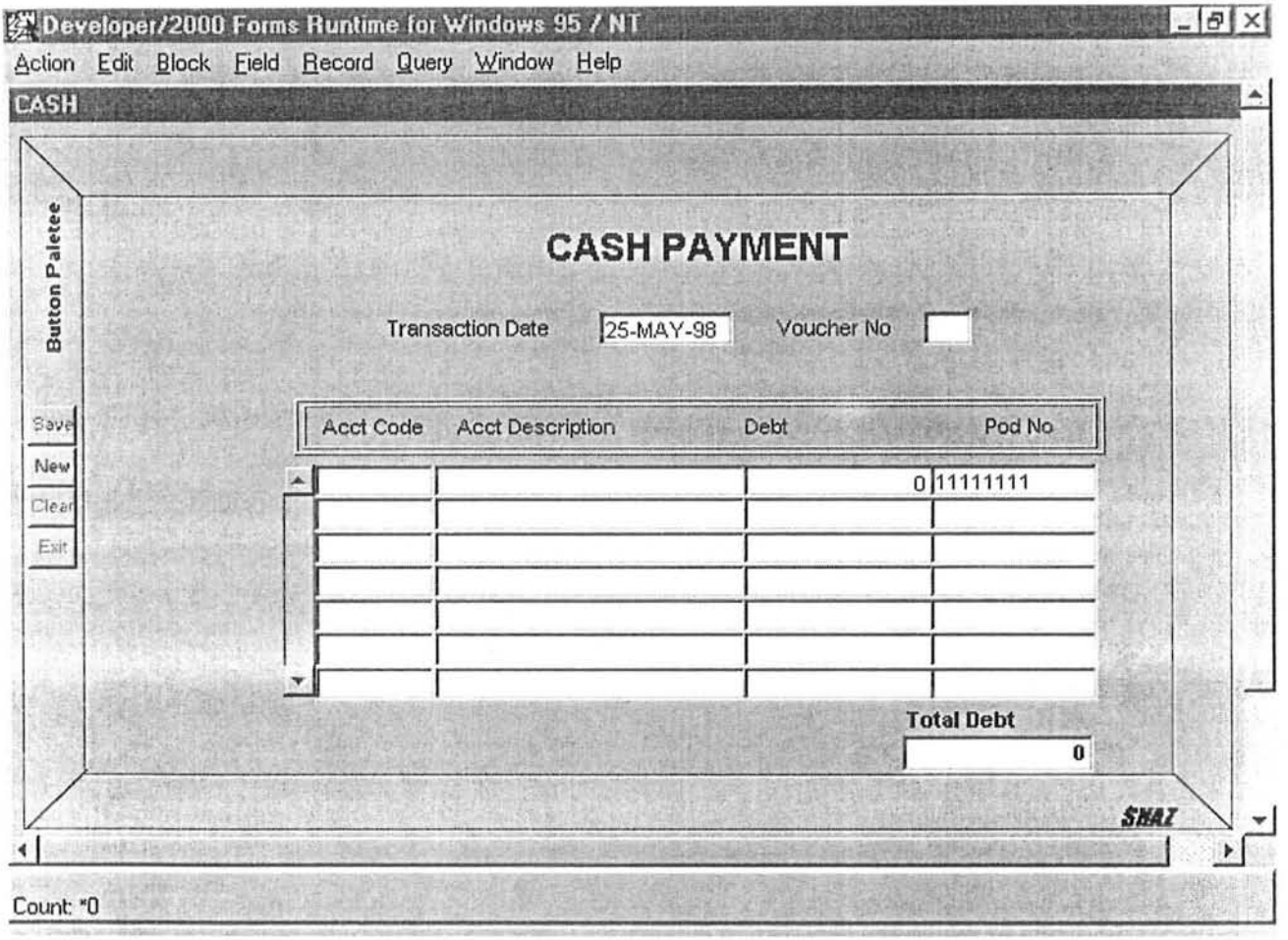

Developer/2000 Forms Runtime for Windows 95 / NT
Action Edit Block Field Record Query Window Help
Button Paletee
New
Save
Clear
Exit
Cash Receipt
Transaction Date
25-MAY-98
Ledger Type
Account Code
Account Description
Credit
Invoice No
10000000
0
111111
Total Credit
0
SHAZ
Count: *0

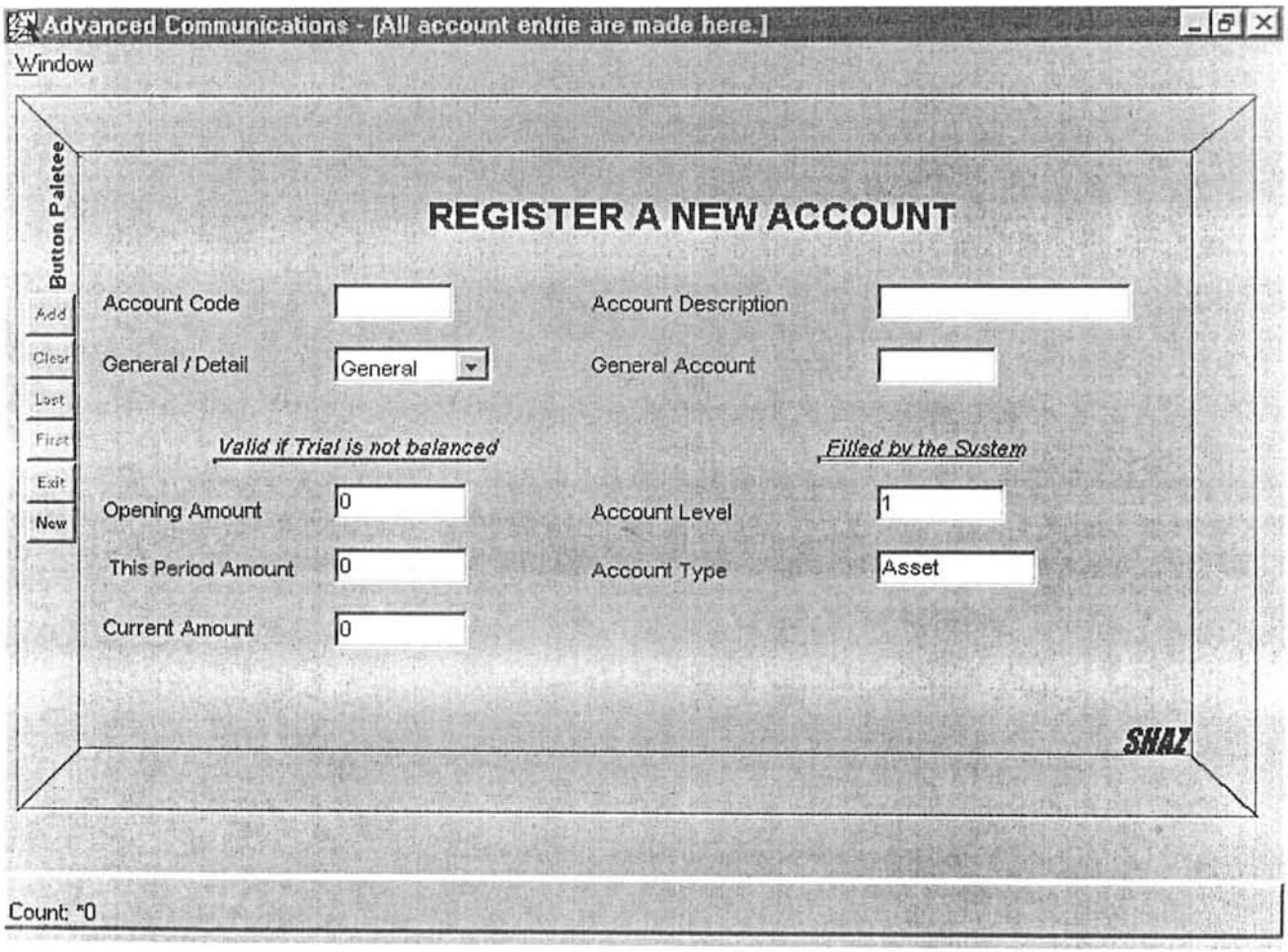

Advanced Communications - [All the items of the inventory are enrolled here.!]
Window
Button Paletee
Add
Clear
Last
First
Exit
New
REGISTER ITEMS
Item Code
Item Description
Unit
Item
Total Quantity
0
Selling Price
0
Cost
0
Record Qty
0
Vendor Code
Open Stock
0
Bad Stock
0
Good Stock
0
In Rma
0
Out Rma
0
SNAZ
Count: *0

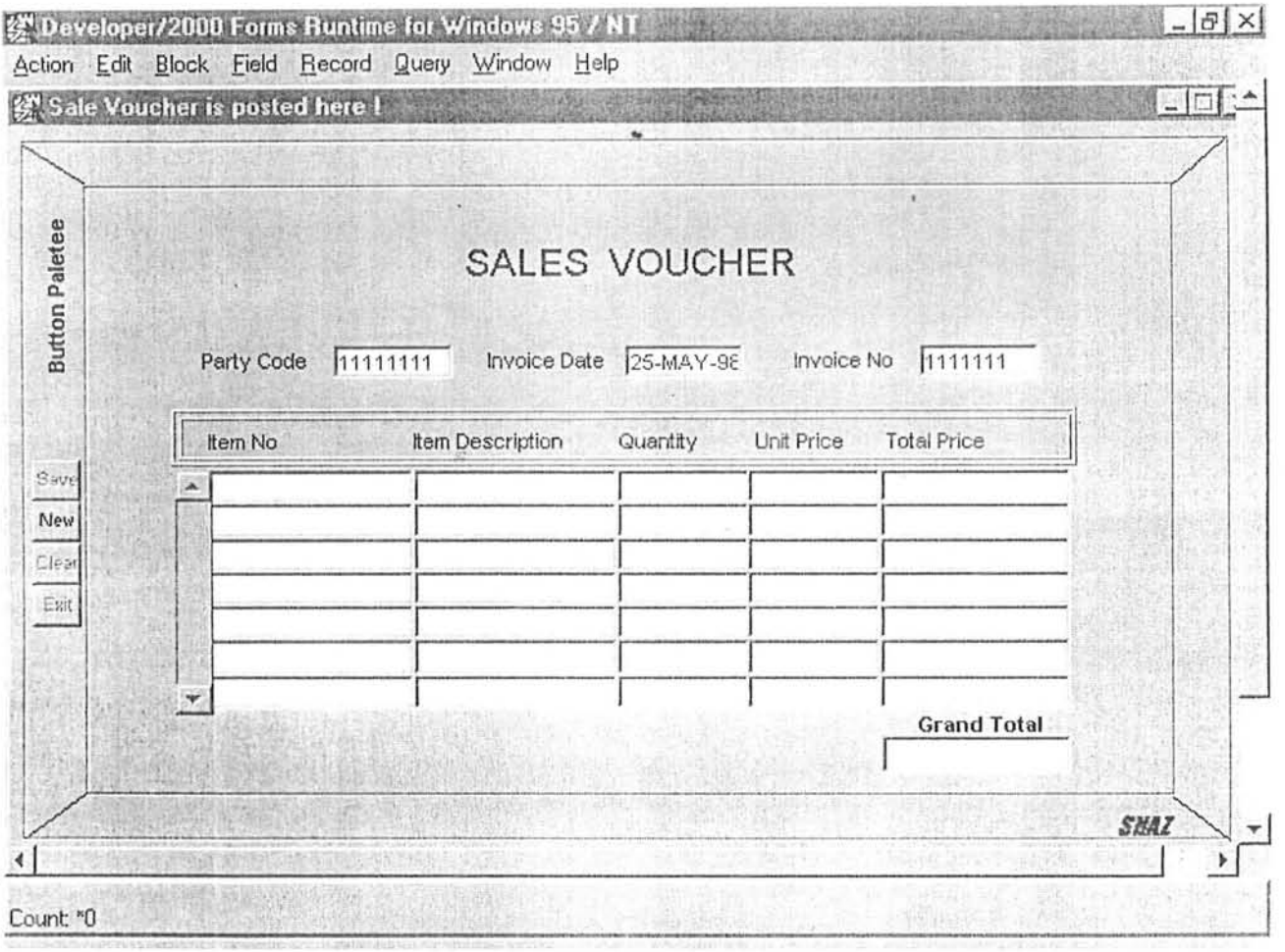

Developer/2000 Forms Runtime for Windows 95 / NT
Window
Journal voucher is posted here1
Button Paletee
Save
New
Clear
Exit
Journal Voucher
Ledger Type
Transaction Date
25-MAY-98
Account Code
Account Description
Debt
Credit
Transaction No
Invoice No
Pod No
0
0
111111
111111
Total Debt
Total Credit
0
0
Count: *0


Developer/2000 Forms Runtime for Windows 95 / NT - [Purchase Order is posted here !]
Action Edit Block Field Record Query Window Help
Button Paletee
Save
New
Clear
Exit
PURCHASE ORDER
Party Code
Purchase Order No
11111111
Bill No
Purchase Order Date
25-MAY-9
Item No
Item Description
Quantity
Unit Cost
Total Cost
Grand Total
SHAZ
Count: *0


Developer/2000 Forms Runtime for Windows 95 / NT
Action Edit Block Field Record Query Window Help
Purchase Return Voucher
Button Paletee
New
Clear
Exit
Save
Purchase Return Voucher
Purchase Order No
Party Code
Bill No
Purchase Order Date
25-MAY-
Item No
Item Description
Cost
Quantity
Items already returned
Items to be returned
Items returnec
Grand Total
0
SHAZ
Count: *0


Developer/2000 Forms Runtime for Windows 95 / NT
Action Edit Block Field Record Query Window Help
CASH
Button Paletee
Save
New
Clear
Exit
CASH PAYMENT
Transaction Date
25-MAY-98
Voucher No
Acct Code
Acct Description
Debt
Pod No
0
11111111
Total Debt
0
SHAZ
Count: *0


Advanced Communications - [All account entrie are made here.]
Window
Button Paletee
Add
Clear
Last
First
Exit
New
REGISTER A NEW ACCOUNT
Account Code
Account Description
General / Detail
General
General Account
Valid if Trial is not balanced
Filled by the System
Opening Amount
0
Account Level
1
This Period Amount
0
Account Type
Asset
Current Amount
0
SHAZ
Count: *0


Developer/2000 Forms Runtime for Windows 95 / NT
Action Edit Block Field Record Query Window Help
Sale Voucher is posted here !
Button Paletee
Save
New
Clear
Exit
SALES VOUCHER
Party Code 11111111
Invoice Date 25-MAY-98
Invoice No 1111111
Item No
Item Description
Quantity
Unit Price
Total Price
Grand Total
SHAZ
Count: *0

Trial Balance 1

Time : 09:10:09 PM
Dated : 25-MAY-98

Acct Code	Account Description	Opening amount		This period amount		Current Amount	
		Debt	Credit	Debt	Credit	Debt	Credit
10000000	Capital Account	.00	.00	.00	3,000.00	.00	3,000.00
20000000	Asset Account	.00	.00	4610.00	.00	4,610.00	.00
30000000	Liability Account	.00	.00	.00	2,000.00	.00	2,000.00
40000000	Revenue Account	.00	.00	.00	1,150.00	.00	1,150.00
50000000	Expense Account	.00	.00	1540.00	.00	1,540.00	.00
	Summary :	.00	.00	6,150.00	6,150.00	6,150.00	6,150.00

Trial Balance

Dated : 25 MAY 1998

Acct Type	Account Code	Account Description		Debit	Credit
Asset	21100000	Opening Stock		2,500.00	.00
	21200000	Inventory COGS		40.00	.00
	21300000	Purchase Account		7,130.15	.00
	22222222	sdfsd		.00	4,980.15
			Total :	**9,670.15**	**4,980.15**
Capital	11000000	Profit and Loss Account		.00	.00
	11111111	tertert		.00	1,500.00
	12222222	tererter		.00	1,500.00
	13333333	test		.00	.00
			Total :	**.00**	**3,000.00**
Expense	55555555	werwer		1,500.00	.00
			Total :	**1,500.00**	**.00**
Liability	31111111	test		.00	1,000.00
	33333333	terter		.00	1,000.00
			Total :	**.00**	**2,000.00**
Revenue	44444444	test		.00	1,000.00
			Total :	**.00**	**1,000.00**
				Total Debit	**Total Credit**
			Summary :	**11,170.15**	**10,980.15**

Balance Sheet

Capital

11000000	Profit and Loss Account	.00
11111111	tertert	1,500.00
12222222	tererter	1,500.00
13333333	test	.00
Total Capital		**3,000.00**

Liability

33333333	terter	1,000.00
31111111	test	1,000.00
Total Liability		**2,000.00**

Profit & Loss account .00

Asset

21100000	Opening Stock		2,500.00
21200000	Inventory COGS		40.00
21300000	Purchase Account		7,130.15
22222222	sdfsd	-	4,980.15
Total Asset			**4,690.00**

Profit = Asset - Capital - Liability - **310.00**

Inventory Value

Time : 07:59:32 PM

Dated : 25 MAY 1998

Item No	Item Description	Cost	Total Qty	Total Price
1111111111111	werwe	110.31	33.00	3,640.23
2222222222222	werwer	36.56	32.00	1,169.92
3333333333333	tewrtert	20.00	9.00	180.00
			Grand Total	4,990.15

P R O F I T I T E M W I S E

Time : 08:12:11 PM
Dated : 25 MAY 1998

Item No	Quantity	Price	Cost	Profit	% Profit	Profit / Unit	Unit Selling Price	Unit Purchase Price
1111111111111111	8.00	300.00	330.93	- 30.93	.00	- 3.87	37.50	41.37
2222222222222222	2.00	20.00	36.56	- 16.56	.00	- 8.28	10.00	18.28
3333333333333333	2.00	10.00	20.00	- 10.00	.00	- 5.00	5.00	10.00

Chart ofAccounts

Dated : 25 MAY 1998

Time 09:40:40 PM

Account Code	Account Description	Account Type	Account Level	General /Detail	General Account
10000000	Capital Account	Capital	1	G	0
11000000	Profit and Loss Account	Capital	2	D	10000000
11111111	tertert	Capital	2	D	10000000
12222222	tererter	Capital	2	D	10000000
13333333	test	Capital	2	D	10000000
20000000	Asset Account	Asset	1	G	0
21000000	Total Inventory Account	Asset	2	G	20000000
21100000	Opening Stock	Asset	3	D	21000000
21200000	Inventory COGS	Asset	3	D	21000000
21300000	Purchase Account	Asset	3	D	21000000
22000000	Cash Account	Asset	2	G	20000000
22222222	sdfsd	Asset	2	D	20000000
30000000	Liability Account	Liability	1	G	0
31111111	test	Liability	2	D	30000000
33333333	terter	Liability	2	D	30000000
40000000	Revenue Account	Revenue	1	G	0
41000000	Sales Account	Revenue	2	G	40000000
42000000	Sales Return Account	Revenue	2	G	40000000
44444444	test	Revenue	2	D	40000000
50000000	Expense Account	Expense	1	G	0
51000000	Cost of goods sold	Expense	2	G	50000000
53000000	Purchase Retuen Account	Expense	2	G	50000000
55555555	werwer	Expense	2	D	50000000

Income Statement

ed : 25 MAY 1998
e : 07:39:20 PM

	Account Code	Account Description	Debt	Credit
venue				
	42000000	Sales Return Account		710.00
	41000000	Sales Account		860.00
	44444444	test		1,000.00
	Total Revenue			**1,150.00**
	Cost Of Goods Sold			**40.00**
	Gross Profit		**1,110.00**	
penses				
	53000000	Purchase Retuen Account	.00	
	51000000	Cost of goods sold	40.00	
	55555555	werwer	1,500.00	
	Total Expense		**1,540.00**	
	Net Income		**430.00**	

Inventory Movement

Item No 111111111111111

Item Description werwe

Dated : 25 MAY 1998

Opening Stock 20

Time : 07:51:02 PM

Transaction Date	Sales	Purchases	Sales Return	Purchase Return	Balance
05 MAY 1998	0	2	0	0	22
06 MAY 1998	3	0	0	0	19
07 MAY 1998	0	6	0	0	25
08 MAY 1998	3	0	0	0	22
10 MAY 1998	2	0	0	0	20
11 MAY 1998	0	5	0	0	25
12 MAY 1998	0	0	2	0	27
18 MAY 1998	0	0	3	0	30
20 MAY 1998	0	0	2	0	32