INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®
UNIVERSITY OF OTTAWA

SYSTEM SCIENCE

THESIS


Student Name: Guowei Liang
Student No: 1763470

Thesis Director: Professor Daniel Lane

August 13, 2002

© Guowei Liang, Ottawa, Canada, 2002
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-76602-0
Abstract

The purpose of this thesis is to analyze the business management process for small to medium size enterprises (SMEs) and to develop an automated product management system for integrated business processing. Due to the relatively small scale of operation of these companies, they experience difficulties in applying readily available tools in the area of product management. The target of the analysis is to develop a prototype software system for automatic product management. Moreover, this software developed for this thesis is applied to a real business case — that of Inshore Fisheries, a small scale integrated fishing operation in Atlantic Canada. The system consists of six components that are grouped into two categories of input and output analysis. The Input category includes production operations and customer ordering. The Output category includes four performance reports: (1) order invoices, (2) production worksheets, (3) inventory tracking, and (4) periodic income statements. The entire system is implemented using Visual Basic 6.0. The database is designed using Microsoft Access. The four performance reports in the output are designed using Microsoft Excel spreadsheets. Furthermore, a GUI integrates all components together to form a complete business processing system.

Based on the results of the application of this system to Inshore Fisheries, this software solution can be applied to other SMEs by making specific modifications but without revising the basic principle and system design of the software system. This system possesses a great amount of flexibility and extendibility for practical business process decision-making.
Acknowledgements

Special thanks, with a grateful heart, go to Professor Daniel Lane both for his professional knowledge of business management and for fishery industry operation. His wisdom and his experience provide this thesis a uncountable contribution. His support and supervision are essential for the entire research. Special thanks also go to Inshore Fishries, especially go to Yvon d'Entremont, business office manager, and company co-owner Jean-Guy d'Entremont. Without their help, the APMS has nowhere to put into practice. Without the feedback from them, APMS would end up to be a windbaggary. Special thanks further go to Professor Jean Michel Thizy. He provided us a very useful PC environment.

The thesis defense committee members also provide this thesis many helpful comments. They are: Professor Daniel Lane (Supervisor), Professor Jean Michel Thizy (Chairman), Professor Doug Angus (Examiner), and Professor Gilles Paquet (Examiner).
Table of Contents

1 Introduction ......................................................................................................................... 1
  1.1 SMEs: The current situation ............................................................................................ 1
  1.2 Objective and Model overview ....................................................................................... 3
  1.3 Outline of the thesis ........................................................................................................ 4

2 Literature Review .................................................................................................................. 5
  2.1 Canadian SMEs .............................................................................................................. 5
  2.2 Business Process Management (BPM) Software solutions: Spreadsheet/Database
      approaches .................................................................................................................. 7
      2.2.1 Spreadsheet solutions ......................................................................................... 8
      2.2.2 Database solutions ............................................................................................ 9
      2.2.3 Excel vs. Database applications ......................................................................... 9
  2.3 BPM software solutions applications ........................................................................... 10
      2.3.1 Banco BBA Creditanstalt S.A. (Banco BBA 2000) ........................................... 10
      2.3.2 MemberWorks Canada Corporation (Rainier 2001) ......................................... 10
      2.3.3 Agris Corporation (Company Coach 2001) ..................................................... 11

3 Methodology ....................................................................................................................... 12
  3.1 The automatic product management system model ....................................................... 12
  3.2 How the model works .................................................................................................... 12
      3.2.1 Using VB to construct the user GUI .................................................................. 13
      3.2.2 VBA – Macro Control inside the Microsoft Excel spreadsheets ....................... 16
  3.3 Performance reporting .................................................................................................. 17
      3.3.1 Four output components .................................................................................... 18
      3.3.2 The integration of output reports ...................................................................... 20
  3.4 Access Database .......................................................................................................... 21

4 Case Analysis and Expected Results ................................................................................. 23
  4.1 Inshore Fishery Inc. ..................................................................................................... 23
  4.2 Motivation ..................................................................................................................... 23
4.3 Business Process Management Challenges .................................................... 24
4.3.1 The management requirements .................................................................... 25
4.3.2 Inshore Office software ............................................................................. 26

4.4 The database design of Inshore ...................................................................... 26

4.5 The Inshore APMS Model .............................................................................. 28
4.5.1 Input — Production .................................................................................... 31
4.5.2 Input — Order ........................................................................................... 36
4.5.3 Output — Invoice ...................................................................................... 39
4.5.4 Output — WorkSheet ............................................................................... 40
4.5.5 Output — Inventory .................................................................................. 42
4.5.6 Output — Income Statement .................................................................... 43

4.6 Use of APMS Model ...................................................................................... 44

4.7 Implementation in Inshore ............................................................................ 45

5 Research Issues ............................................................................................... 46
5.1 The selection of the spreadsheet software .................................................... 46
5.2 Other BPM model applications ..................................................................... 47
5.3 Model Scale – Local vs Internet .................................................................. 48
5.4 Model Implementation .................................................................................. 48

6 Conclusion ....................................................................................................... 49
6.1 Application to Other SMEs .......................................................................... 49
6.2 Software Requirement .................................................................................. 50
6.3 Extendibility ................................................................................................ 50
6.4 On-line order ............................................................................................... 51

7 Bibliography .................................................................................................... 52
Appendix A .......................................................................................................... 55
The Invoice Statement ....................................................................................... 55
Appendix B .......................................................................................................... 56
The Production Worksheet .............................................................................. 56
| Appendix C | 58 |
| The Inventory Report | 58 |
| Appendix D | 59 |
| The Income Statement | 59 |
| Appendix E | 60 |
| Source Code of the APMS | 60 |
List of Figures

Figure 1.1: The Outline of the Principal Model .......................................................... 3
Figure 3.1: The VB Graphic User Interface (GUI) ....................................................... 14
Figure 3.2: The Flow Chart for VBA Code ................................................................. 17
Figure 3.3: The Relationship among Four Components ............................................. 20
Figure 3.4: The Design of the Relationships ............................................................ 22
Figure 4.1: The Database Relationship of Inshore Fisheries ..................................... 28
Figure 4.2: The Opening window ............................................................................. 29
Figure 4.3: The VB Graphic User Interface (GUI) for Inshore Fisheries .................. 30
Figure 4.4: The Production Window for Choosing a Species ..................................... 31
Figure 4.5: Species Information Summary ............................................................... 32
Figure 4.6: Input Production Details ...................................................................... 33
Figure 4.7: Production Process for Fresh .................................................................. 34
Figure 4.8: The Production Process for Frozen ......................................................... 35
Figure 4.9: The Production Process for Slabs ............................................................ 36
Figure 4.10: Order Window for Choosing a Customer ............................................... 37
Figure 4.11: Customer Information ......................................................................... 38
Figure 4.12: Invoice Pop-up Window ....................................................................... 40
Figure 4.13: Work Sheet Pop-up Window ................................................................. 41
Figure 4.14: Inventory Pop-up Window ..................................................................... 42
Figure 4.15: Income Statement Pop-up Window ....................................................... 43
List of Tables

Table 1: Sample Production Report Design ................................................................. 18
Table 2: Sample Invoice Statement Design ............................................................... 18
Table 3: Sample Inventory Report Design ................................................................. 19
Table 4: Sample Income Statement Design ............................................................... 19
1 INTRODUCTION

The purpose of this research is to describe the development, design, testing and application of an automatic spreadsheet-based product management system to assist small and medium-sized business operations reporting and decision making.

1.1 SMEs: The current situation

SMEs (Small and Medium-sized Enterprises) are categorized by their firm size in terms of the number of employees in the organization—usually less than 250 employees in Europe or Canada, and less than 500 employees in the U.S.A. Most SMEs are independent and non-subsidiary operations. They may appear in the format of a sole proprietorship or a partnership and cover all the business sectors, e.g., production, finance, marketing, accounting, human resource management. The critical role in the development and success of SMEs is provided by progressive entrepreneurs who are willing to take more risks in order to get a better market opportunity.

Along with the growth of the economy, the market structure becomes more and more complex. As the first economist who studied successfully the dynamics of market structures, Schumpeter claimed that the growth of the economy is a result of entrepreneurial innovative behavior (Schumpeter 1962). Although many large corporations are specialized in formal R&D (research and development) and register many patents, they are not the sole source of innovation in the economy. The entrepreneurs in SMEs are also essential to economy development because they are more willing to take the risks in order to take advantage of the fleeting business opportunities. One research study found that small firms are more efficient than large ones in R&D (Acs and Audretsch 1990). Due to these innovative activities, SMEs drive market dynamics and stimulate the economic growth through continual birth, expansion, contraction, and death of firms (OECD 2000a).
Accompanying the globalization of the economy, SMEs are more and more exposed to international market competition. SMEs are "directly producing about 20 percent of OECD exports and about 35 percent of Asia’s exports" (OECD 1997, p. 7). Further, many SMEs have also established supply, sales, and promotion networks with some transnational corporations (TNCs). These outsourcing activities of the TNCs increase the market share of SMEs. Moreover, due to the dynamism of frequent entry and exit of SMEs, the international productivity and the overall economy will be developed continually.

In Canada, SMEs are a highly diversified group. Their activities are spread to every business sector, especially the service sector, e.g., wholesale and retail, restaurant and hotel, communications and the Internet. As the marketing plan of Corporations Directorate (Canada 2001) stated in 1995, SME businesses are the primary target markets in British Columbia, Alberta and Ontario (Kindra 1997). It is important to mention that 99% of firms in Canada are SMEs. Over 2.6 million SMEs account for 43% of private sector GDP and created 81% of net new jobs in last decade. The federal government in Canada has supported the growth of SMEs through many programs for SMEs that include tax reduction, human resource management, R&D, and the information highway.

Along with the continuous emergence of new technology and globalization of economy, the importance of economies of scale is declining. This fact provides more opportunity for smaller scale operations to be successful. But it also creates many business barriers to SMEs simultaneously. These include: low productivity, financial shortage, technology renewal, constrained managerial skillset, and new industry standards. The objective of most SMEs is to reap the maximum profit from the current market segmentation. They focus on promotion and sales of products and customer services. It is very easy to overlook the internal management. In order to extend their companies’ life cycle and to succeed opponents in competitive market situations, the successful SMEs should take advantage of existing technologies to improve the management skill and capabilities of internal business process.
1.2 Objective and Model overview

In this research, we explore and integrate up-to-date software technologies to assist an SME in production planning and standard reporting of business operations performance. This includes the capability of gathering and analyzing internal information in order to manage better.

The emphasis of this project is on the design of a performance-reporting package of the SME's business process management (BPM) system. From production to sales, the system records all production and customer entries and stores them in a database. Based on the production and the delivery of the orders, a supporting system is established to manage the inventory and report on aggregate economic performance on a periodic basis. Furthermore, all the information should be retrievable later under a simple request from the user and in an easy user select format that will enable users to query the database system. Based on these principles, the proposed research design of the business process management system is based on the following model overview:

![Graphic User Interface (GUI)](image)

**Figure 1.1: The Outline of the Principal Model**

The BPM model allows users to provide business process information (such as production dynamics, inventory characteristics, revenue and cost components) in a structural manner to a database. The database provides the information necessary to develop *pro forma* company reports. The user interface allows company members to query the database, to make additions,
deletions and changes, and to request *pro forma* company reports automatically generated by the report writing functionality. The model design caters to specific company requirements and results.

By implementing the automation strategy through the system, it reduces human error. The system manages regular business process activities automatically, e.g. issuing invoices, preparing customers’ orders, etc. These tasks release employees to carry out the operation more efficiently. By using the automation system, the employees can handle more business documents but use less time. Furthermore, due to using a "trustworthy system", employees don’t have to double-check the documents.

1.3 Outline of the thesis

This thesis describes the development, design, testing and application of an automatic spreadsheet-based product management system for a small and medium-sized business operation. It outlines the motivation, the technologies, and real world application relevant to this research. The following section contains a literature review that takes a broad look at the related issues published in the main areas of research. Next, the research methodology is presented. A cornerstone of the research is the case study for the application and implementation of the research output to a small-scale integrated fish harvesting and processing operation in southwest Nova Scotia. Finally, in the last chapter of the thesis the expected results of this work are described and the extensions of the research are discussed briefly.
2 LITERATURE REVIEW

The following presents a brief discussion on the development of product management systems in SMEs and an analysis of system components. The literature review is divided into the following areas:

1. Production management systems in Canadian SMEs: a brief history of SMEs and their use of automatic business process solutions to date.
2. Business Software solutions: an introduction to the business applications of spreadsheets and databases and the differences between them.
3. Examples of case studies: 3 cases that describe the actual use of the cutting-edge technology to deal with the internal business process management problem.

2.1 Canadian SMEs

Since the mid-1970s, entrepreneurial activities among small and medium-sized enterprises increased. In 1980s and 1990s, we perceive a further rapid growth of SMEs in Canada. Both the numbers of the businesses and numbers of employees have increased dramatically (Baldson 2000) over this period. Moreover, these entrepreneurs are coming from different ethnic cultures and the proportion of participating females is increasing. Most Canadian SMEs are active in the service industry. The following lists some aspects of current SMEs business in Canada (Canada 1994):

- 99% of firms are SMEs and 53% of the labor market is in SMEs.
- 60% of SMEs are in the private sector and they account for 43% GDP of the private sector.
- 32% business profit and 29% gross sales are generated by SMEs.

Implementing the proper strategies makes SMEs successful and competitive. These strategies are related to the firms' activities like financing and technology. We can summarize the key successful factors (KSFs) as follows (Baldwin, Chandler, Le and Papailiadis 1994):
1. Flexibility of quick response to customer requirements, quality of products and customer service.
2. Rigorous TQM (Total Quality Management), just-in-time inventory control.
3. Marketing and operation management.
4. Skilled employees.
5. Fast capital turnover and use of limited capital.
6. Adaptability to new technology, new products, and new materials.
7. Government programs.

Among all these KSFs, "technology", or in other words, R&D, is a measure that differentiates more successful from less successful SMEs. New technology appears in the market at an accelerating rate. In order to catch up to the pace of the information age and satisfy the dynamic market requirements, R&D is an ongoing issue. R&D provides SMEs with new cutting-edge technologies. Knowing the new technology is indispensable, for example, to entering into the foreign market.

There are four types of technology strategies for an SME:
- improving own existing technology,
- using other firms' technology,
- refining other firms' technology, and
- developing a new product (Baldwin, Chandler, Le and Papailiadis 1994).

Except for using others' technology, the other three types involve the R&D process of the development of technology to different extents. Thus, in order to increase competitive advantage, R&D is necessary. R&D contains product innovation and process innovation. Both of these are crucial for an SME to survive in the severe market competition it must face.

In this research, the major focus is on the business process innovation and development. It explores aspects of products management using cutting-edge technology in the form of available business software systems. According to this innovation, SMEs can upgrade their capability of
internal management efficiency and thus increase their competitive advantage in the marketplace.

As mentioned in the first part of this thesis, developing management skills and applying technology potential are the weaknesses attributed to failed Canadian SMEs. Thus, how to take the full advantage of the existing technologies and integrate them into management operations is the key problem for most SMEs. Compared with large firms, SMEs are still in a ‘sunrise stage’. A high entry rate of new SMEs is also associated with a high exit rate (WIPO 2001). This fact doesn’t necessarily imply that the SMEs which failed to survive don’t have a marketable product. Actually, it is a result of the incomplete and inexperienced management teams in these SMEs. On the other hand, the high exit rate is not an adverse indicator of the SME business. It is more like a learning process. The entrepreneurs obtain valuable management skills and experience from the failures. Further, as stated in Baldwin, Chandler, Le and Papailiadis (1994), R&D makes a difference for the more successful firms. R&D enables the SMEs to have the ability to adopt new technology. It enables the Canadian SMEs to grow continually.

2.2 Business Process Management (BPM) Software solutions: Spreadsheet/Database approaches

Along with the rapid popularization of the PC (Personal Computer), using computerized or electronic spreadsheet and databases to deal with data are already very common approaches to solving business process management problems. Especially in the SME business environment, entrepreneurs are familiar with computerized bookkeeping of employee information, inventory, customer invoices and so on.

A BPM software system is a solution for SMEs seeking a technology advantage. The basic requirement for BPM software is to automate the interaction of the customers, employees, business transactions, and the data (MetaStorm 2001). By using BPM software, it helps a SME to release resources from tedious paper work, and allows the business process to become more productive and more profitable. BPM software are business tools which enable users to automate the business transaction process by integrating some kinds of software that could be either customized software, or some common commercial business software such as spreadsheet
or database software designed for specific company needs. As a solution, users can obtain regular customized, professional business performance reports.

Spreadsheet solutions, such as those provided by the leaders in the field, Microsoft or Corel, have very powerful functionality. These spreadsheet and database software packages summarize and analyze information from different resources and then present them in user-friendly format to help decision makers (e.g., entrepreneurs) gain the business "big picture" for the company (Power 2000). Moreover, built-in templates or user-defined templates are capable of providing much convenience to users.

The following describes the basic characteristics of these software solutions using common spreadsheet and database platforms.

2.2.1 Spreadsheet solutions

Microsoft Excel is a spreadsheet program used to organize, manipulate, and chart data and to perform calculations. It is particularly good at managing tabular data. It also excels at creating, publishing, viewing and analyzing data for professional reporting purposes (Microsoft 1998). It has a set of rich analysis and formula tools that enable business users to do calculations much easier and simpler than before. For example, the pivot table and query functions (although not widely recognized and used in business applications) allow users to summarize and compare data quickly. The Excel worksheet-to-worksheet link, management and function wizards enable users to handle data applications more automatically.

After the introduction of Microsoft Excel 97 (JWalk & Associates 2001), Excel became a more intelligent business software package since it integrated the Visual Basic for Application (VBA) programming language inside the software allowing advanced users to develop automatic execution of multiple spreadsheet functions. Through VBA, and Object Linked and Embedded (OLE) functionality, other programming package applications can now communicate internally with other Microsoft software products, e.g., Excel passing information back and forth with the
Access database package. By using VBA, macros (short executable programs) can be manipulated by allowing the spreadsheet to perform some user-specific commands.

2.2.2 Database solutions

Microsoft Excel List and Microsoft Access both can serve as a PC-based relational Database Management System (DBMS). A DBMS is a program is a program that lets one or more computer users create and access data in a database (Brydon 1997). DBMS is challenged by the following:

- It should support two industry standards: Standard Query Language (SQL) and Query By Example (QBE).
- Take advantage of VB and macros.
- The new object-oriented extensions need to encapsulate the concepts of state and behavior information.
- Must provide user-friendly graphic users’ interface (GUI) such as the Access Database windows.

2.2.3 Excel vs. Database applications

Excel is not necessarily the perfect tool for complicated data architectures that require significant data input and validation. The following issues arise for considering development-using Excel (Powell 1995):

- Data size: if it is large (more than 10,000 records), database software like Access may be more appropriate.
- Data Security: Excel has a very low security level. Its security is only for an entire workbook, not for each record (row).
- Multiple user requirements: Excel is not designed for multi-user applications. For some business transactions processing involving many uses in different locations, it may be better to use multi-user database management software.
• Application objective: Excel is effective at a combination of data analysis, statistical analysis, diagram plotting, report generation while database software is more suitable for intensive data entry and data validation.

2.3 BPM software solutions applications

Following on the more formal descriptions above regarding business software, some examples are necessary for further understanding software solutions applied to business process management. The following lists three cases to illustrate potential advantages of deploying new technologies.

2.3.1 Banco BBA Creditanstalt S.A. (Banco BBA 2000)

Banco BBA is a Brazilian investment bank that provides a wide range of wholesale financial services. As the global competition increases continuously, BBA recognized the importance of IT. It was challenged to reduce its IT cost without compromising its market position and market share. By using Microsoft Office 2000, Banco BBA Creditanstalt S.A. reduced 17% of the end-user costs in distributed environment and 9% of overall business costs. Before taking advantage of Office 2000, it had deployed Office 97 for a few years to analyze and report on a range of its financial services business process work. The new features of Office 2000 allowed Banco BBA to further reduce its operation expenses by increasing efficiency. GartnerGroup conducted end-user and IT surveys both before and after the deployment of Office 2000 to evaluate the impact on end-user productivity and satisfaction. The survey showed that the time saving for the each GartnerGroup category is as follows: casual learning & self-support reduced by 43%, peer-to-peer support reduced by 43%, and file and data management reduced by 9%.

2.3.2 MemberWorks Canada Corporation (Rainier 2001)

MemberWorks Incorporation is a direct marketing company located in Montréal, Canada (MemberWorks 2001). It designs innovative membership programs for its customer companies.
These programs help customers get more value by providing them with necessary information not generally available to the public. As part of its business nature, MemberWorks has to respond to customers very quickly. The information for the customers must also be reliable. These challenges required MemberWorks to have an integration of the existing warehouse automation systems in order to customize the configuration of membership benefits. Thus, it uses clustered SQL 2000 and Windows 2000 as its business process management solution. Further, by taking advantage of the web-based application which uses a built-in function in Visual Basic, different departments in different locations can create and modify member benefit configurations. This BPM software solution provides much more membership flexibility for MemberWorks; and the customized reports are generated much faster and more conveniently.

2.3.3 Agris Corporation (Company Coach 2001)

Agris Corporation, headquartered in Atlanta, develops software solutions for agricultural communities in U.S., Canada, South Africa, and Australia. It has a sales support department which is responsible for generating proposals using Corel’s WordPerfect for the sales representatives. When it merged with Farmland Industries, they had many repeated “cut and paste” to be carried out because Farmland used the Excel spreadsheet package to produce the proposals. It took time to produce a complete proposal because the graphics and fonts did not match between these two pieces of software. As a consequence, it created friction between the sales department and administration. After a market search, they found a software, ProposalMaster, which is a sophisticated, automated, PC-based system for generating professional and customized proposals. ProposalMaster can match the software that Agris already had in place: Word, Excel, Access, and PowerPoint. ProposalMaster also interfaced with Agris' existing contact management software. After implementing ProposalMaster, Agris now produces a proposal much faster than before. It creates a better correlation between what sales representatives need and what the proposals provide. ProposalMaster has the flexibility to customize the proposal to meet the specific opportunity and address the particular client demand. As a result, Agris’ productivity has increased.
3 METHODOLOGY

The following sections describe the technical implementation of a general BPM system model for an SME. This model has a GUI (Graphic User Interface) for users to choose the preferred business process analysis and reporting functions. Each of these business process functions is represented by a model component. Basically, there are six components which are grouped into two categories. These include the general business process reports for Production and Customer Order input reports, and Invoice, Inventory, Production Worksheet, and Income output reports (Figure 3.1).

3.1 The automatic product management system model

Consider a small to medium sized enterprise, which has to develop its business dynamics in order to maintain market position and competitive advantage through its production and sales processes. Due to its relatively small scale of business, the company cannot afford to have its own R&D team. It is critical, however, for the company to have an effective automatic operation and production management system. Thus, it has to deploy a suitable software solution to meet the requirements of its internal business process management. These requirements may include the real-time reporting of sales and production processes which are the key sources of company benefits and expenditures. The business reports are necessary because the company is required to keep the records of company transactions. Furthermore, the production and the sales processes affect the company ongoing products inventory positions. Finally, periodic reporting formal reports, e.g., the income statements, require that an aggregate review of business process activities be conducted over a defined period. A generic look at this automatic product management system model is described in the following subsection.

3.2 How the model works

Inside the model, we deal with business process model components as objects. For example, production, sales, inventory, and income are different objects. All these objects are linked into
the database, which is designed and implemented using the software of Microsoft Access®. The connection between the user interfaces and the database is made by taking advantage of the new technology of ActiveX Data Objects (ADO). ADO is a data access technology which combines all the best features of Data Access Objects (DAO), Open DataBase Connectivity (ODBC), and Remote Data Objects (RDO) (Wright 1998). Moreover, VBA is applied in the spreadsheets that are designed with the software of Microsoft Excel®. The spreadsheet is linked into the database using macros. The information of all the tables in the database is manipulated with the sequenced query language (SQL). This design allows us to develop a user-friendly Graphic User Interface (GUI) that helps the user to carry out all the functionalities of the model in a much more straightforward and easier than otherwise. These functions of course may include the data input and the information retrieval.

3.2.1 Using VB to construct the user GUI

The VB-GUI provides an interface for users to input the raw data. On the interface, it describes itself by some text that tells the user what and how to use this software. The user simply clicks on the appropriate menu buttons on the interface in order to fulfill different functions. Each button will bring the user to another pop-up window in which the user can enter data and then insert them into the database or retrieve the requested information from the database in report format. This interface should also allow easier access to the spreadsheet reports so that the user can view or print out the reports according to their preference. A basic form of the GUI showing the business process model functionality looks as in Figure 3.1.

In Figure 3.1, there are two major parts for the user interface: INPUT and OUTPUT. Each part has several functionalities; and every function is treated as an object in this model.
Figure 3.1: The VB Graphic User Interface (GUI)

- **INPUT**: This component includes two functions; one is for production (product output "Production" button) and the other is for customer order ("Order" button).

  (1) "Production" activates another pop-up window to ask the user to select a specific product he/she wants to process. Users can update the information of that species from that pop-up window. After choosing the product, the user then can go step by step according to the instruction of the interface in order to enter all the production data of that particular production item. All these data are eventually stored in the database. Moreover, the number of the product item produced is added into the inventory.

  (2) "Order" works in a similar way as "Production". After clicking on the button of "Order", another pop-up window will be activated. The user can update the customer's personal information according to that window. In order to choose the correct customer among all entered in the database of the company's customers, the user moves through the customer list. On selection, the user then activates an Excel spreadsheet to process the customer's order order and to print the order report form. Again, all the data will be inserted into the database for future retrieval.

- **OUTPUT**: There are four reports included in this function. Each of these reports is designed according to a standard pro forma statement format. These statements include
company reports on the customer order invoices, the company's production worksheet reports, the current inventory position of the product portfolio, and aggregate cost and earnings (income) statements. These reports arise form the GUI buttons of "Invoice", "WorkSheet", "Inventory", and "Income Statement", respectively, as shown in Figure 3.1. The user can modify and update the information directly from these reports. These reports are developed based on the linkages to the company input and database by macro programs using VBA.

(1) "Invoice" activates a pop-up window that presents the user with a combo box asking the user to choose from a list of invoice numbers recorded in the database. According to this invoice number selected, the program retrieves all the related information on a particular customer order from the database. This information may be previewed as a Microsoft Excel spreadsheet report. It may also be printed out as a report and sent to the customer as the order invoice informing the customer of the sales agreement, products purchased and payment required for this order.

(2) "WorkSheet" activates another pop-up window which also asks the user to pick a production number from a combo box of recorded database information. Through this production number, the user can also examine all related information presented on the Microsoft Excel spreadsheet. The spreadsheet further calculates the profit and the cost of the production activity. This spreadsheet is a final report for the company on the production activity.

(3) "Inventory" activates a different pop-up window, which asks the user to identify a particular product. After the user selects a product from a combo box list of all products, this user can preview the current inventory status of that product that is presented via an Excel spreadsheet.

(4) "Income Statement" also activates a distinct pop-up window which will ask the user to provide input on fixed expenses. With this information, the system combines the results from the production and order activities over a specified time period to construct the
Income Statement for the period. The income statement is presented in pro forma format on a separate Excel spreadsheet report.

3.2.2 VBA – Macro Control inside the Microsoft Excel spreadsheets

The application of Visual Basic programming language built around Microsoft Office business applications increases the power of this popular software to a great extent. Visual Basic for Application (VBA) allows programmers to design spreadsheet and database macros that execute tasks specified by user preference. VBA is a useful programming tool for integrating several Microsoft Office solutions together, such as integrating Excel and Access in this model. The following provides a descriptive example of how the VBA Macro will be used to control the spreadsheet and the database.

For the “Order” executable, there is a defined Excel spreadsheet template that matches the company’s order report form. This template has a built-in macro. This macro is controlled by a command button. It can transfer all the data about the customer’s request to the database using Microsoft ActiveX Data Objects (ADO) Library. For example, assume three customers request orders today. The first orders 25 different products, the second orders 38 different products, and the third orders 42 different products. Each customer has to click the command button, (for example “SUBMIT/SAVE”) when they finish filling out the order form. This “SUBMIT/SAVE” button is already assigned to a macro programming function within VBA. It activates the database file and transfers the information about the customer order just input to that file. Information for the three customers goes to that file one by one, that is, 25 of the first, 38 of the second, and 42 of the third. All 105 product request records are now stored in that database file. The VBA code for the “SUBMIT/SAVE” button is designed as following control flow diagram:
3.3 Performance reporting

In this research, there are four output spreadsheet components and associated reports: (1) production worksheet, (2) invoice statement, (3) inventory reports, and (4) income statement. They are all designed using the spreadsheet methods of Microsoft Excel. Moreover, these reports have to be integrated together using the database and the GUI input components. They all can be retrieved according to the output part from the GUI which is described briefly in Figure 3.1. They are also designed using the built-in cell functions provided by Excel which calculate cell values automatically and consistently, and thus eliminating the possibility for human error.
3.3.1 Four output components

**Production.** The Production worksheet, as its name suggests, records the basic data and report form information about the production process. This information includes the wage rate, the quantity produced for each individual product item, the cost of production, the production area, the exchange rate, e.g., between the US dollar and the Canada dollar (as required), and so on, in accordance with company procedures. Table 1 provides a view of a design for a simple production report.

<table>
<thead>
<tr>
<th>Exchange rate (US vs. CAN): $1.5018</th>
<th>Wage rate (CAN/hr): $15.85</th>
<th>Production area Code: 4Xp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product items:</td>
<td>Item 1</td>
<td>Item 2</td>
</tr>
<tr>
<td>Quantity (units)</td>
<td>50</td>
<td>69</td>
</tr>
<tr>
<td>Expense (CAN)</td>
<td>$5,000</td>
<td>$6,900</td>
</tr>
</tbody>
</table>

*Table 1: Sample Production Report Design*

**Invoice.** The Invoice statement is a worksheet for billing the customer. It keeps track of the order made in the input segment by the customer. All the information from the customer is shown on this worksheet. The customer name, the products ordered, the total amount due, the invoice number, and the date of ordering and payment are all recorded on this invoice statement. Further, the invoice report checks the remaining quantity of each product item in inventory as an available quantity for completing sales (Table 2).

<table>
<thead>
<tr>
<th>Customer name: Minutes Food</th>
<th>Account No.: 50001</th>
<th>Invoice No.: 62011</th>
<th>Ship VIA: R&amp;R</th>
<th>Date: Aug. 21, 2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product items</td>
<td>Item 1 Avail.</td>
<td>Item 2 Avail.</td>
<td>Item 3 Avail.</td>
<td>Total</td>
</tr>
<tr>
<td>Quantity</td>
<td>80 100</td>
<td>67 200</td>
<td>53 169</td>
<td>200</td>
</tr>
<tr>
<td>Price ($US)</td>
<td>9,000</td>
<td>7,900</td>
<td>7,100</td>
<td>24,000</td>
</tr>
</tbody>
</table>

*Table 2: Sample Invoice Statement Design*

**Inventory.** Inventory reports are used for keeping track of the volume of all product items. Knowing the inventory levels helps manager make decisions about whether to manufacture more
or not and which product item should be produced. Along with the production and the customer order, the amount of inventory moves up with new production and down with customer orders over time. Therefore, the business process management systems needs to include a report to keep track of inventory levels for all the product items in the firm’s product portfolio. Further, the inventory status is used to check and verify the final invoice statement as to how much customers can order at maximum by product. If the customer requires more than the inventory, the system sends back a warning message to the user to inform about the inventory shortfall (Table 3).

<table>
<thead>
<tr>
<th>Inventory Statement</th>
<th>Date: Apr 12, 2002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Species</td>
<td>Item No.</td>
</tr>
<tr>
<td>Product 1</td>
<td>CA2B1</td>
</tr>
<tr>
<td>Product 2</td>
<td>FL1S2</td>
</tr>
<tr>
<td>Product 3</td>
<td>TU1S4</td>
</tr>
</tbody>
</table>

Table 3: Sample Inventory Report Design

Income Statement. The last of the four output reports is the income statement. This report provides a pro forma statement for a given period of the profit or the loss for the company. The definition of the period is flexible but normally is done monthly, quarterly or yearly. The income statement is a consolidated performance measure of a collection of production activities (i.e., operating costs of the firm) and customer orders (i.e., operating revenues to the firm). The following table presents a simple example of the income statement report.

<table>
<thead>
<tr>
<th>Income Statement</th>
<th>Year to Date Jan-Apr (in CANS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Revenue</td>
<td>$900,000</td>
</tr>
<tr>
<td>Operating Cost</td>
<td>500,000</td>
</tr>
<tr>
<td>Other Expense</td>
<td>100,000</td>
</tr>
<tr>
<td>Profit before Tax</td>
<td>400,000</td>
</tr>
<tr>
<td>Tax (50 %)</td>
<td>200,000</td>
</tr>
<tr>
<td>Net Profit (Loss)</td>
<td>200,000</td>
</tr>
</tbody>
</table>

Table 4: Sample Income Statement Design
3.3.2 The integration of output reports

All business process spreadsheets whether input or output is related to each other to provide an integrated system of business reports. The production worksheets add the amount of every product into the inventory worksheet. The invoice statements take out the same amount from the inventory. The inventory also tells the invoice statement how much is remaining for future sales and triggers production decisions. The cost comes from the production, and the revenues from the invoice orders. Cost and revenues go together to calculate the final profit position of the firm.

The production worksheet has all the information of the production process of every product item. The quantity (Table1) of each item produced adds to the inventory entries. The expense data items from the production worksheet (Table 1) also are linked to the income statement (Table 4). The Invoice statement asks how much is available (Table 2) for every product in advance of new orders by establishing a link to actual inventory. The total balance (Table 3) of the inventory is reduced by the quantity for each item (Table 2) in the invoice statement. The prices (Table 2) for each item that the customer orders contribute to the firm’s revenue. These amounts go to the income statement as the revenue (Table 4). Figure 3.3 illustrates the component linkages of the system.

![Diagram showing the relationship among four components: Inventory, Production, Invoice Statement, and Income Statement.](image)

**Figure 3.3: The Relationship among Four Components**
3.4 Access Database

The database is designed and developed using with the popular Microsoft Access software as part of the Microsoft Office suite of business software. According to the information required by the six model components (2 input and 4 output modules), the database basically contains 9 tables which store the relevant data. These database tables are described below.

1. Invoice Table: it contains the records of customer orders.
2. Production Table: it stores the information for every production process such as date, exchange rate, and production area, etc.
3. Inventory Table: it records the amount of each frozen product item.
4. Customer Information Table: it keeps the information on the customers.
5. Product information Table: it stores the data for processing each product including workers’ salary rates, packing costs, and overtime, etc.
6. Frozen Products Table: it stores the records of the frozen product for each production process.
7. Fresh Products Table: it stores the records of the fresh product for each production process.
8. Other Products Table: it stores the records of other products produced in the production process including by-products.
9. Invoice and Customer Table: it has the information which has the invoice number and the corresponding customer name.

These 9 tables are linked with defined relationships among them. The following diagram describes these general relationships (detailed explanations of the relationships between each table are referred to in section 4.3 of this thesis related to the case study). Figure 4.1 below summarizes the implied relationships among the various database tables.

21
Figure 3.4: The Design of the Relationships

The following section presents details of the system design and use for a specific case study for a small vertically integrated fish harvest and processing firm in southwest Nova Scotia.
4 CASE ANALYSIS AND EXPECTED RESULTS

Chapter 3 described the model developed in the course of this research in general terms. This chapter presents a business case that applies this model practically for a real application. This case is used to test the validity and functionality of the automatic business processing model. The expected result is also presented in this case study.

4.1 Inshore Fishery Inc.

The fishing sector in Atlantic Canada is a significant economic activity for the region, especially for small to medium-sized enterprises (SMEs) that dominate the sector throughout the provinces of Nova Scotia, Newfoundland, New Brunswick, and Prince Edward Island. The SME business environment with its heavy reliance on nearby American markets is considerably dynamic. Inshore Fisheries of Pubnico, Nova Scotia is a third generation family-owned and operated fish processing and harvesting company. It has diversified into various areas in the fishing sector of southwestern Nova Scotia, including groundfish, invertebrates, and shellfish. Inshore Fisheries produces fresh and frozen fish products primarily for a strong customer base in the eastern seaboard of the United States. The company specializes in ground fish species (e.g., cod, haddock, flounder), but also shrimp and silver hake (Inshore 2001). Inshore Fisheries has five mobile gear vessels that vary between the sizes of 45 and 63 feet in length. The crew in each vessel is usually 3 to 5 people. Inshore Fisheries has a complete fish products line for fresh and frozen seafood products. They can deliver the packages of fresh fillets in groundfish of 10lb, 20lb, and 25lb.

4.2 Motivation

Inshore Fisheries, a small, family-owned and operated diversified fishing and processing company located in Nova Scotia's southwestern area, produces fresh and frozen fish products primarily for a customer base in the eastern seaboard of the United States. Improved business
performance in this competitive marketplace requires companies to be on top of every business transaction. For example, Inshore needs to know very clearly about product inventory, especially for frozen products, and to be prepared to respond quickly to customer demands for fresh fish products. The APMS model is designed to assist Inshore in its ability to respond quickly based on the potential customer sales and inventory and production possibilities.

At present, Inshore Fisheries operations are handled by a limited number of well-versed managers who have effective control over buying, selling, and production. During the peak business periods, decisions are made with rapid-fire urgency and very little documented information. Business reporting is done after the fact and as a result is a cumbersome task that requires automation.

The APMS system links productions reports from handling and processing raw fish product to meet customer demands arising from product invoice lists, to frozen inventory products, and finally to reports on net operating income over a series of production, inventory, and customer invoices.

The APMS allows the production, inventory, invoice, and income statement reports to be linked. This means that changes in one business report (e.g., production worksheets) are immediately reflected and updated in related reports (e.g., product inventories). This system provides an easy means of providing work reports data and then automatically uses these data to generate business reports, e.g., an invoice report for a customer, a worksheet production statement, and an operating income statement report for managers.

The major motivation of developing this APMS system is to save time and minimize the possibility of human error through a standard reporting design.

4.3 Business Process Management Challenges

In order to improve the business performance in this competitive marketplace (US – Canada), it requires the company to be on top of every business transaction. During the active
fishing season (usually between June and December) *Inshore Fisheries* requires an efficient and effective way to deal with the daily barrage of information on product management from its customers, from its vessels, and from other vessels from whom it may be advantageous to purchase supplies of raw fish products to keep up inventories in the face of customer demand. For example, *Inshore Fisheries* needs to know very clearly about product inventory, especially for frozen products, and to be prepared to respond quickly to customer demands for fresh fish products. The APMS solution represents a very effective tool for what *Inshore Fisheries* needs to improve its production and customer order processes. The following section describes the management requirements and selection of particular business office software as well as the development of the APMS model specifically in response to the needs of the company.

### 4.3.1 The management requirements

Thus, *Inshore Fisheries* are required to develop their internal business process management and recording systems in order to meet the continuous business change in markets, exchange rates, supply and demand for its fish products. Accordingly, company managers are at an advantage if they find new and efficient ways to keep track of products inventory, client business lists, production control, and operating incomes and bottom line profits. This model can assist *Inshore Fisheries* in its ability to respond quickly based on the potential customer sales and inventory and production possibilities.

*Inshore Fisheries*’ requirements for a business process management system include the functionality that allows the production, inventory, invoice, and income statement reports to be linked together. This means that changes in one business report (e.g., production worksheets) are immediately reflected and updated in related reports (e.g., product inventories). The model proposed here provides an easy means of providing work reports data and then automatically uses these data to generate business reports, e.g., an invoice report for a customer, a worksheet production statement, and an operating income statement report for managers. The user can use the GUI to retrieve these reports.
4.3.2 Inshore Office software

Currently for Inshore’s office work, the staff uses the Corel Suite software package primarily to produce basic office reports, letters, etc. Driven by the staff and the awareness that much more of their analysis and report writing could be automated, they were seeking to develop more advanced business process systems.

Driven by its clientele and the rise of electronic communications, Inshore has experienced some difficulties in its compatibility and use of its office software systems. Consequently, they have been gradually shifting from Corel software to the more widely used Microsoft Office software.

Given current capabilities of Inshore Fisheries, it wanted to develop a system which has more business process management functionalities. The challenge for this research was to provide an easy to use software solution adapted to their specific needs in the fishing industry to match the way in which they currently do their work. As part of this discussion, the selection of an appropriate software platform took place as a precursor to development and implementation of the proposed model. After some deliberation, it was decided jointly that the Microsoft Office would be the most convenient platform on which to develop the system. Moreover, stimulated by the discussion, the company was enticed to move up its move to Microsoft Office.

4.4 The database design of Inshore

Before designing the full model and after the software issue was resolved, the database setup was required to be developed first. In order to maintain the compatibility of the different software applications, Microsoft Access was adopted primarily in order to make the database compatible with the other parts of this software that would also be using the business office products from Microsoft, as well as to take advantage of the software capabilities associated with the suite of Microsoft software.
As described in section 3.4 above, the database is composed of nine interrelated tables. Thus, *Inshore's* database consists of 9 related tables designed specifically for this case problem as follows. Details of the following text description is also illustrated in the Microsoft Access Figure 4.1 found below.

1. ComInfo Table: contains the customer information. It has nine fields including the customer name, contact information, and the address of this customer.

2. Inv_Com Table: contains the information about matching the customer with the invoice. It has invoice No., date, customer, and ship 4 fields, where "customer" refers to the name in the table of ComInfo and the field ship means the shipping company name here.

3. Invoice Table: contains four fields, invoice No., item No., quantity, and unit price, where invoice No. relates to the invoice No. in the table of Inv_Com.

4. Inventory Table: consist of seven fields, species name, item No, size, skin on or boneless, package amount, quantity, and index. The field item No. relates to the field item No. in the table of Invoice. The field index here is used as a key of sorting all the items.

5. Species_Info Table: contains the basic production information for each of the species, such as the wage, cost, packing expense, and freight, where the field species relates to field species in the table of Inventory.

6. Production Table: contains 10 fields here, which includes production No., date, species, vessel name, processing area, rate of return, and three kinds of exchange rate. The field species relates to the field species in the table of Species_Info.

7. Frozen Products Table: contains only three field which are production No., item No., and weight. Item No. relates to the item No. in the table of Inventory. Production No. relates to the production No. in the table of Production.

8. Fresh Products Table: contains three field also which are production No, size and weight. The production No. relates to the field of production No. in the table of Production.

9. Slab Products Table: contains 3 fields; they production No., the amount of slab in frozen, and in fresh. The production No. relates to the field of production No. in the table of Production.
4.5 The Inshore APMS Model

In order to handle the problems mentioned above, a customized design software solution is required. Thus, the automatic product management system (APMS) was developed for this purpose. Since this model is being applied for Inshore Fisheries in this case, it needed to have an opening window which introduces the company name:
The Figure 4.2 above tells the user what this software is for and what it does generally.

As described in part 3.2.1, the system required a user-friendly GUI. This GUI must fulfill all the necessary business functions referred to in 3.2.1. Based on the idea illustrated in Figure 3.1, a customized GUI was developed using Visual Basic (Figure 4.3). Clicking on the command button “Start” from Figure 4.2 brings the user to this GUI. It includes two main parts: one is the input functions and the other is the output functions. Inside the input frame, it consists of two functions, both production and customer order. Users can choose either one by just clicking on the appropriate command button. Inside the output frame, it is composed of four functions which can generate four different operation performance reports. Further, this GUI remains on the user’s desktop of the computer unless the user clicks on the “EXIT” button at the end of the session. The functions for this application are described in detail in the following six subsections: two for input and four for output.
Figure 4.3: The VB Graphic User Interface (GUI) for Inshore Fisheries

There are seven command buttons in the above figure. The functions for these point and click buttons are defined as follows:

1. Production: a button for processing the production procedure.
2. Order: a button for customer order.
3. Invoice: a button for generating the performance report of customer order based on recorded customer invoice numbers.
4. WorkSheet: a button for generating the performance report of production based on recorded numbered production worksheets.
5. Inventory: a button for generating the performance report of inventory status by product.
6. Income Statement: a button for generating the \textit{pro forma} performance report of the operation situation of the firm over a given time period.
7. Exit: close the application.
The following section itemizes the functionality of the Inshore Fisheries APMS.

4.5.1 Input — Production

According to the GUI shown in Figure 4.3, users can select either “Production” or “Order”. Then a series of pop-up windows will come up to ask the user to fill in the related information. After going through these windows, the data are stored in the linked database tables.

For illustration purposes, consider the process of using the Inshore model for the “Production” function. Figure 4.4 shows the pop-up window when clicking on “Production”. This window asks the user to choose the species of interest that will be produced. Each species (product) will have some basic required information such as freight cost, packing cost, and other processing costs as noted in the inout frame of Figure 4.5 below. Users can modify this information if necessary. In order to view or update the data, the user needs simply to click on the button “View/Update Species Production Data” shown in Figure 4.4. This command button can trigger another pop-up window which shows all the fundamental information related to this species (Figure 4.5). Closing the window of Figure 4.5 will return the user back to the window of production (Figure 4.4).

![Figure 4.4: The Production Window for Choosing a Species](image-url)
From Figure 4.4, there are three command buttons and a drop down combo box. These are:

- Combo Box: contains 7 species for users to choose. These species are restricted to: Catfish, Cod, Flounder, Haddock, Ocean Perch, Pollock, and Turbot.
- View/Update Species Production Data - allows the user to confirm that the data of the species information is correct, otherwise, it can modify or update the data.
- Input Production Details - it brings users to the Figure 4.5 window where they can process the production data.
- Exit: closes this window.

![Species Information Summary](image)

**Figure 4.5: Species Information Summary**

For Figure 4.5 below, there are two command buttons and a arrow button. These are:

- OK: it means all data in this window are valid; the user is returned to the production window (Figure 4.4)
• Exit: if there are no modifications or any updates on the window, it brings the user back to Figure 4.4.
• Arrow Button: it allows the user to go through all the species one by one.

After the user checks and/or updates all the information of the species that is ready to be processed, the user can proceed to the next step by clicking on "OK". At this point, the user is returned main window “Production”, Figure 4.4. Now, the user must click on button “Input Production Details” (Figure 4.4) in order to process this species. Figure 4.6 shows the processing window for the chosen species. Users can then type the related information into these input text boxes. In order to maintain software integrity, the buttons of “Fresh”, “Frozen”, and “Slabs” in Figure 4.6 are initially “faded out”, i.e., no entries are permitted until a production number is provided from the outset.

Figure 4.6: Input Production Details

The Inout Production Details menu contains five command buttons and three combo boxes (Figure 4.6). These are:
Production Weight data by product for the various categories of harvested fish including:

- Fresh: process fresh products.
- Frozen: process frozen products.
- Slab: process slab products.

In order to complete the entire production process, the user has to click on these three buttons one by one. They are illustrated below in Figure 4.7, Figure 4.8, and Figure 4.9, respectively.

- Choose Another Species: process another species other than the one which is currently identified at the top of the menu or go back to the previous window after the user finishes processing the current species.
- Exit: close this window without processing.

Provide fixed information about the company’s production definition, i.e., vessel, area, source:

- Vessel Combo Box: list all the company’s vessel names for single selection.
- Area Combo Box: list all the production areas for single selection.
- Source Combo Box: list all sources available for single selection.

Figure 4.7: Production Process for Fresh
There are two command buttons in the Fresh window.

- Submit: submit all the amounts and insert into database.
- Exit: close this window without enter any information.

![Image of Fresh window with data entry fields]

**Figure 4.8: The Production Process for Frozen**

There are also two buttons in the Frozen window; they have the same function as the corresponding buttons in the Fresh window.

- Submit: submit and insert all the values into the database.
- Exit: close the window without saving any data.

For the Slabs window (shown in Figure 4.9), its two command buttons function in the same manner as the Frozen and Fresh windows above.

- Submit: submit and insert all the values into the database.
- Exit: close the window without saving any data.
Figure 4.9: The Production Process for Slabs

After finishing the execution of the Fresh, Frozen, and Slabs production input tables, these the production process is complete and the user can return to the main menu, e.g., Figure 4.3.

4.5.2 Input — Order

Next we review the input function “Order”. When the user clicks on the button “Order” from the GUI (Figure 4.3), the following window (Figure 4.10) pops up. This software can deal with three kinds of customers:

1. If the customer has previously ordered seafood products from Inshore Fisheries, then the customer is an “existing customer” and the user can retrieve the customer name and information in the combo box by clicking on the down arrow.
2. If the customer is a new customer who has never ordered any products from Inshore Fisheries before, this customer and tombstone information must be added to the contact information of Inshore’s customer database.
3. For an existing client, one may also want to update the contact information in the Inshore’s database because this customer has perhaps already changed its telephone number or even moved to another address.
Figure 4.10: Order Window for Choosing a Customer

From the figure above, there are three command buttons in this Order window.

- **New Customer/Update Existing Customer**: for a new customer to enter some basic contact information or for an existing customer to update the contact information.
- **Process Order**: orders can be placed directly by clicking this button.
- **Exit**: close the window without doing anything.

For the customer belonged to the first category, “existing customer”, one can proceed directly to click the button “Process Order” after finding the name from the combo box. But for the other two kinds of customers (category 2 and 3 above), they need to click on the button “New Customer/Update Existing Customer”. This activates a new pop-up window (Figure 4.11) which shows the information requirements for the customer database. New customers can enter the contact information by clicking on the button “Add New Record”. On this pop-up window, the user can further review the other existing customers’ information using the arrow button.
Figure 4.11: Customer Information

There are command buttons and an arrow button in this Customer information window.

- **Add New Record**: All the text boxes will become blank by clicking on this button. A new customer can thus be entered with the related information into these empty fields.
- **Submit**: After the customer data has been entered all the data for the customer is updated with the information in this window. The Submit button records these data into the corresponding tables in the database.
- **Exit**: Close this window without any modification.
- **Arrow button**: Allow the user to view all the customers one by one through the database table.

Clicking “Process Order” (Figure 4.10) activates a Microsoft Excel spreadsheet. This is a pre-defined spreadsheet template. The spreadsheet contains a company order form and a built-in button “SUBMIT / SAVE” which turns on the spreadsheet macro. This macro transfers all the
data from the Excel spreadsheet to the related tables in the database file. On this Excel spreadsheet, the user is required to fill in the cells that are highlighted. These include such data as: INVOICE NUMBER, INVOICE DATE, and Ship VIA fields. Further, before the user enters the quantity of any particular product item; one should look in advance at the Stock-on-Hand column which shows how many units of the product are remaining in the inventory. The quantity of the order must not exceed the storage amount in the inventory. After filling out this order, the user can simply click on the macro button “SUBMIT/SAVE” to complete the entire order process and transfer data to the database.

4.5.3 Output — Invoice

After the customer places the order, the customer can anticipate receiving the bill. Inshore Fisheries has to send this customer the invoice to demand payment. As a basic customer requirement, the software APMS has developed a function to generate a customized company invoice. Every invoice requires an invoice number corresponding to the invoice number in the order sheet. According to this invoice number, the APMS can retrieve all the related information about the customer order including the total amount of money that that customer should pay.

When Inshore Fisheries finishes the order for the customer, it needs to send the invoice to the customer as soon as possible in order to collect the bill. Thus, the operator of APMS clicks on the button of “Invoice” from the GUI (please refer to Figure 4.3). This activates a pop-up window which asks the user to pick up an invoice number from a drop-down combo box (Figure 4.12). Originally, the combo box of Invoice No and all three text boxes of Date, Customer, and Ship VIA are empty until the operator chooses the invoice number (e.g. 112211 in Figure 4.12) when the text boxes are filled with the corresponding data from that invoice. These data are retrieved from the database and are consistent with the invoice number.
Before sending the invoice to the customer, the operator of APMS needs to take a look at the report in advance. This operator also wants to make sure everything on the invoice report is correct, especially the amount of payment. By clicking Preview shown on Figure 4.12, it brings the operator to a Microsoft Excel Spreadsheet which has all the information for the customer who just placed an order with an invoice number of 112211 (see also Appendix A for a full example of this output report). Moreover, the button of “Preview” is in nonclickable (“gray”) at the beginning. It becomes enabled only by choosing a invoice number from the combo box.

4.5.4 Output — WorkSheet

When a production process is completed, a report is required to keep track of the company internal management. There is a production number for each production process that is entered by the user. In order to generate the production report, or “WorkSheet”, the production number is very important. Based on this requirement of the users of the software solution, APMS has this function already built-in. The user of the APMS can retrieve all the information related to any particular production number and generate a WorkSheet report for that particular production run.

When Inshore Fisheries finishes the production process, the operator can then click on the button “WorkSheet” from the GUI (Figure 4.3). This activates another pop-up window (Figure
4.13). Similar to the pop-up window of the invoice (Figure 4.12), all the text boxes there are initially empty. This Work Sheet window asks the operator to select a production number from the drop-down combo box. After picking up a production number, e.g., 512268 in the case of Figure 4.13, the fields for Date, Vessel, Source, and Area are filled in automatically. These data are retrieved from the database which is built up when processing the product items. All these data are consistent with the unique production number.

![Work Sheet Pop-up Window](image)

**Figure 4.13: Work Sheet Pop-up Window**

After choosing the production number, the operator can view the report by clicking on the Preview button. Note that the “Preview” button is disabled when opening the Work Sheet Pop-up window. Choosing a production number will enable this button. Clicking on the “Preview” button activates a Microsoft Excel spreadsheet which contains all the related information about the production number 512268 (see also Appendix B for a full example of this output report). The operator can also further modify the values for any cell on this spreadsheet. When all the values are correct, this spreadsheet can be saved and printed as a final report for the production process.
4.5.5 Output — Inventory

Inventory is a critical factor for the product management inside a SME due to its small operation scale. A SME must keep its inventory at a proper level which is not too high or too low. If the inventory level were too high, it would occupy lots of space and thus take a great amount of cost to maintain all the product items. If the inventory level were too low, it would not have enough product items for the customer demand and thus reduce the profit of the company. Therefore, this requires that the product management software has a function to monitor the inventory level.

Inside the APMS, the inventory level goes down after the customer places an order and the inventory level goes up right after the production process has occurred. Thus, Inshore Fisheries has to monitor the inventory closely. It can do this by clicking on the button of “Inventory” on the GUI (Figure 4.3). This action brings the operator an inventory pop-up window (Figure 4.14). The inventory in the database is organized by different species. There are seven species in total. The operator of the software APMS can choose any species of interest from the drop-down combo box.

![Inventory Pop-up Window](image)

**Figure 4.14: Inventory Pop-up Window**

The selection of a species also enables the “Preview” button. The operator thus can go to preview the inventory level (see also Appendix C for a full example of this output report). This report shows the current amount of every product item for that species. The operator can make a decision on whether to produce more or not. And this operator can know how much to produce
for each item. Further, the inventory level can be input directly to the customer order spreadsheet which tells the operator how much is remaining in the inventory.

4.5.6 Output — Income Statement

Economic performance analysis is a very essential step for the product management of an SME. SMEs have tight budgets relative to larger companies. Inshore Fisheries, must have a short payment collection period and a short cash flow period in order to increase its competition capability. A strong financial performance is required for Inshore Fisheries in the ravaged fisheries sector. Thus, it needs a tool to monitor its economic performance situation very closely.

To meet this requirement, the APMS contains a built-in function to evaluate economic performance inside the software. The operator can observe a button called “Income Statement” on the software GUI (Figure 4.3). After clicking on this button, a pop-up window appears (Figure 4.15). There are a few text boxes on this window. Each of these text boxes represents an exogenous fixed input item related to the computation of the company’s income statement.

![Income Statement](image)

**Figure 4.15: Income Statement Pop-up Window**
After the operator confirms all these monetary values are correct by clicking "OK", the "Preview" button is enabled. This button will again activate the Microsoft Excel spreadsheet where the report of income statement is generated. All the numbers on the income statement pop-up window (Figure 4.15) are imported into this spreadsheet and calculated automatically. The final report is illustrated in Appendix D.

4.6 Use of APMS Model

The development of the APMS for Inshore Fisheries was made possible with the assistance of Yvon d’Entremont, Business Office manager, and company co-owner Jean-Guy d’Entremont of Pubnico, Nova Scotia. Their assistance and feedback throughout the period of software design and testing provided a valuable opportunity to observe the basis requirements of the company’s operations.

The company, through its officials demonstrated throughout a desire to become more efficient. Opportunities for increased efficiency presented themselves in the carrying out of basic business tasks, including data input and generation of summary and consolidated report writing, and the monitoring and tracking of regular business transactions including customer purchase requests, and production processing activities. While the company was very familiar with electronic processing to speed communications (e.g., FAX, email, cellular telephone and other wireless communication devices were in common usage around the office well as on board company vessels and down on the wharf), there was less familiarity with the possibilities to enhance business office performance and customer relations via enhanced use of business software.

This project has not yet been fully accepted by the company as a complete replacement for the routine, manual exercises that is has been designed to replace. However, as familiarity with the software grows, it is anticipated that the company will seek out still further opportunities to automate its business processing activities using similar, custom-designed software.
4.7 Implementation in Inshore

Considering the fact that the APMS system has not yet been accepted by Inshore Fishries, it is important to think about the possible barriers to implementation.

Under the current system, the business operation manager has the complete control of all transactions and often must make many decisions quickly. While the APMS could assist in these decisions, the implementation of having to consult the computer to define transactions may be felt to be too disructive.

As well, potential users may have difficulty understanding the software and therefore are immediately skeptical about its benefits. These factors taken into account suggest the need for a “hands-on” training workshop be implemented to assist Inshore fisheries users in answering questions and dealing with potential contingencies that may arise during software use.
5 RESEARCH ISSUES

This section describes several issues that need to be addressed during the course of this research project. These issues include the implementation of the BPM system for Inshore at the local office level, and the future direction of research including wide area, Intranet and Internet applications of the BPM toward a practical e-business system.

The following sub-sections discuss these issues in terms of: (1) the practice of Quattro Pro and Microsoft Excel 2000, (2) the application to other BPM models, (3) the potential Internet application, and (4) model implementation.

5.1 The selection of the spreadsheet software

Currently in the commercial market, we may find two major spreadsheet business solutions: Quattro Pro® and Microsoft Excel®. They both specialize on the data analysis and have a great deal of built-in formulas. For the spreadsheet solution in this case, we may have to make a trade-off between Quattro Pro 9 and Microsoft Office 2000.

Quattro Pro allows us to create professional spreadsheets for data management and analysis. It provides all the tools to produce tables, financial forms, lists, databases, charts, reports, or any other type of data-oriented document. Using Quattro Pro, we can create a notebook that contains a single spreadsheet or an entire project, complete with spreadsheets, reports, charts, and a slide show presentation.

Quattro Pro does have lots of advantages for data handling on any one particular spreadsheet. Unfortunately, in the course of this work, problems had been discovered in the use of Quattro Pro to handle multiple linking tasks that are required. Under the operating system (OS) of Windows (as used at Inshore), these multiple linking activities create some fatal problems, e.g.,

- Inability to save files after data modification or link update;
- Inability to open files with links to other notebooks;
• Computer crashes;
• Tedium updating of links when transferring the spreadsheet system between two machines.

Similar to Quattro Pro 9 described above, Microsoft Office 2000 is also a spreadsheet software with very strong business-oriented functionalities. Again, we use Excel in one of the most popular operating systems - Windows. Basically, all the problems in Quattro Pro that are listed above do not appear in Excel. Excel provides us with an easier linking environment.

Compared to Quattro Pro 9, Excel is much more compatible to Windows OS. Moreover, Excel provides a good working environment for customized spreadsheet model design as required for the Inshore project. It is easier to update all the references that connect to other cells and other sheets. It has a strong error and type checking function. Further, the latest version of Visual Basic for Applications is included with Excel 2000 although we did not use any VBA in this project. Visual Basic 6 provides Excel with features such as model input user forms and support for additional ActiveX controls than Quattro Pro.

All in all, given the problems that were experienced with Quattro Pro, Excel is much easier to expand and manipulate.

5.2 Other BPM model applications

Although the BPM model described here is a tailor-made model for Inshore Fisheries, the principle still applies to other companies, and other business sectors of the economy, especially where SMEs are involved.

As described in a generic BPM system descriptions in the third part of this thesis, this model does not necessarily focus on any particular business case. Actually, any SME company that has problems similar to those of Inshore Fisheries can take advantage of this software solution. In other words, if a company, or an independent subunit of the company which is going
to automate its business transaction process and willing to have the customized print-out of some performance reports can really think about this Automatic Product Management System (APMS).

5.3 Model Scale – Local vs Internet

The current APMS model presented here focuses only on Intranet-type applications, i.e., a private network established within an enterprise. It is not open to the public. It is only used by the employees of the enterprise on proprietary company data. Thus, it does have its constraints. The customers have no access to this network.

An intranet only pays attention to the information communication among its own company network. It can be set up by connecting several local area networks (LANs) together or taking advantage of leased line in the wide area network (WAN). All the employees share the same resources inside this company. In the current information age, most Intranets are connecting to the outside Internet through one or more gateway computers. The people outside this company can obtain access to its network anywhere and anytime. This creates convenience for the customers because they can do the business by just clicking at their homes or offices through the Internet. As a consequence, this may increase the efficiency and thus bring in more business to the SME. Further, it saves lots of paperwork due to electronic business transaction processing.

5.4 Model Implementation

This model generally is suitable for the use in SMEs because it does not take up too many resources to implement and use. It only requires a PC which has Microsoft Excel and Access (contained as part of the Microsoft Office suite) installed. Further, it does not require too much computer skill or professional knowledge for the person who will ultimately control this software because it has a neat and straightforward interface. What the operator needs is to follow the software instruction step by step. Also because of its simplicity, it does have some constraints. For example, it is only compatible with Windows operating system. It does not fit into Unix or Macintosh.
6 CONCLUSION

For an SME, the product management is a very critical business procedure. An efficient product management system provides the dynamics for a company. It increases the vitality for the company in such a severe competitive market. It also can minimize the product maintenance cost and improve the profit position of the company’s stakeholders.

This research focused on how to automate this business procedure. It also built a software prototype APMS (Automatic Product Management System). Aside from applying it to Inshore Fisheries, it was also shown that it could be applied more broadly into other SMEs. The functionality of the APMS is extendable and concise. According to the different requirements of different customer, the APMS can be further developed to a much more complex software by adding some other functions.

All in all, we believe that APMS is a simple software solution for SMEs that need assistance in business process management. We also believe the commercialization of APMS is a challenging but meaningful job when the software design takes into account the specific needs of the company.

6.1 Application to Other SMEs

The nature of SMEs is small-to-medium size business ventures. They do not have a large operational scale. They do not have a large amount of budget. They do not have too many resources. Usually, they cannot afford their own R&D team. On the other hand, just because they are not large, they are faced with market dynamics. They can reach a management decision quickly. They can change their operation very easily in order to meet the different economic situation. Based on these common characteristics of SMEs, APMS can easily be modified for use in each company. According to the different requirements of each company, APMS can make the consistent modifications which fit into this company’s environment. For example, if a company is producing toys, APMS can just change the product items what are currently fish into toys. As
well, the database is modified based on the data that this toy company will generate. Although there are many subtle modifications for the software, the fundamental principal is still the same.

6.2 Software Requirement

The entire software was developed using the tools from Microsoft. The database was designed using Microsoft Access. The spreadsheet solution was Microsoft Excel. The macros inside the spreadsheet were developed using VBA. The coding language for the entire program was Visual Basic. The integrated development environment was VB 6.0.

This software is a standalone version so far which means someone has to install it on his/her own machine if he/she wants to use it. It is not a software which can be running on a server and providing service to all the client machines because it is not a network version. Furthermore, the current system has been designed and will run only on the Windows operating system, including Windows 98, 2000, ME, and XP.

6.3 Extendibility

APMS right now is a very simple software concept. It only fulfills some basic functionalities of a company: production process, order, invoice report, production report (worksheet), inventory report, and income statement. These functions may not satisfy a company’s needs in a more globalized and competitive market. Due to the simplicity of APMS, it can be extended very easily. According to the further requirements of the customer, it can add functions to the APMS. For instance, APMS may satisfy the current economic environment when a company develops this software. Due to the uncertainty of the market, the economic situation may require that the product management reflect some other aspects of the company (e.g. cash flow statement) in the future. It is unreasonable to ask this SME to develop another software in order to meet this new requirement because it involves lots of investment of human power and cost. It can just add this function to the GUI and develop it using Visual Basic. Moreover, the database has to add some tables and fields which are consistent with the new functions.
6.4 On-line order

So far, the entire research has been focused on the internal usage of the company. We assumed that this software solution, APMS, would be used only by the employees inside this company. The operator of the APMS will take the order for the customer.

Considering the popularity of World Wide Web, it is appealing to place order forms on-line which would allow the customer to fill it out directly on the company’s web page. This is also a competitive strategy which can increase the sales volume. Customers can easily access the web page of the company. It is convenient for them if they can just order on-line.
7 BIBLIOGRAPHY


Canada. 2001. Corporations Directorate. Industry Canada,


Appendix A

The Invoice Statement

This is a printed-screen of the invoice statement that is designed for Inshore Fisheries. It contains all information described in Table 2: Sample Invoice Statement Design. After making every transaction, Inshore must record all the customer information, including the customer name, account number, date, ordered product items and the total balance and so on and so forth. This is also the bill that will be sent to this customer in order to charge for the payment.

<table>
<thead>
<tr>
<th>SPICES</th>
<th>SIZE</th>
<th>50/PK</th>
<th>PRC.</th>
<th>Item</th>
<th>QTY</th>
<th>Unit Price</th>
<th>Total Price</th>
<th>BALANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATFISH</td>
<td>12</td>
<td>5</td>
<td>15</td>
<td>CA55</td>
<td>36</td>
<td>$180.00</td>
<td>$6,120.00</td>
<td>$6,120.00</td>
</tr>
<tr>
<td>COD</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>CO21</td>
<td>15</td>
<td>$180.00</td>
<td>$8,100.00</td>
<td>$14,220.00</td>
</tr>
<tr>
<td>COD</td>
<td>8</td>
<td>5</td>
<td>12</td>
<td>CO21</td>
<td>25</td>
<td>$180.00</td>
<td>$4,500.00</td>
<td>$18,720.00</td>
</tr>
<tr>
<td>COD</td>
<td>12</td>
<td>10</td>
<td>5</td>
<td>CO52</td>
<td>12</td>
<td>$180.00</td>
<td>$2,160.00</td>
<td>$20,880.00</td>
</tr>
<tr>
<td>FLOUNDER</td>
<td>4</td>
<td>20</td>
<td>20</td>
<td>FLB4</td>
<td>11</td>
<td>$180.00</td>
<td>$1,980.00</td>
<td>$22,850.00</td>
</tr>
<tr>
<td>FLOUNDER</td>
<td>12</td>
<td>5</td>
<td>15</td>
<td>FL51</td>
<td>15</td>
<td>$180.00</td>
<td>$2,700.00</td>
<td>$25,550.00</td>
</tr>
<tr>
<td>HERRING</td>
<td>4</td>
<td>10</td>
<td>10</td>
<td>HRS5</td>
<td>26</td>
<td>$180.00</td>
<td>$4,680.00</td>
<td>$50,240.00</td>
</tr>
<tr>
<td>HERRING</td>
<td>8</td>
<td>10</td>
<td>15</td>
<td>HRS5</td>
<td>22</td>
<td>$180.00</td>
<td>$5,960.00</td>
<td>$34,200.00</td>
</tr>
<tr>
<td>HERRING</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>HRS5</td>
<td>32</td>
<td>$180.00</td>
<td>$5,760.00</td>
<td>$39,960.00</td>
</tr>
<tr>
<td>PRAWN</td>
<td>12</td>
<td>5</td>
<td>20</td>
<td>PR05</td>
<td>111</td>
<td>$180.00</td>
<td>$19,980.00</td>
<td>$59,940.00</td>
</tr>
<tr>
<td>PRAWN</td>
<td>12</td>
<td>5</td>
<td>15</td>
<td>PR51</td>
<td>54</td>
<td>$180.00</td>
<td>$8,120.00</td>
<td>$66,060.00</td>
</tr>
<tr>
<td>PRAWN</td>
<td>12</td>
<td>15</td>
<td>20</td>
<td>PR55</td>
<td>22</td>
<td>$180.00</td>
<td>$3,960.00</td>
<td>$70,020.00</td>
</tr>
<tr>
<td>PRAWN</td>
<td>12</td>
<td>15</td>
<td>10</td>
<td>PR55</td>
<td>23</td>
<td>$180.00</td>
<td>$4,140.00</td>
<td>$74,160.00</td>
</tr>
<tr>
<td>PRAWN</td>
<td>4</td>
<td>15</td>
<td>15</td>
<td>PR55</td>
<td>11</td>
<td>$180.00</td>
<td>$1,980.00</td>
<td>$76,180.00</td>
</tr>
</tbody>
</table>

Total $76,140.00

Credit $76,140.00

Please detach portion below and send with your remittance.

FROM: INSHORE FISHERIES LTD.

Middle East Palnico,
Harbours Brothers, Nova Scotia

INVOICE NUMBER 112211
INVOICE DATE 01/22/99
PAYMENT DUE DATE 04/22/99
TOTAL AMOUNT DUE $76,140.00

THANK YOU FOR YOUR BUSINESS!

Note: All data presented in this Appendix have been disguised and do not reflect the actual business performance of Inshore Fisheries.
Appendix B

The Production Worksheet

This is printed-screen of the production worksheet which is designed for Inshore Fishery. It contains all information described in Table 1: Sample Production Report Design. After Inshore harvests the products, it sells fresh fish on direct order from customers and flash freezes other product and puts in inventory. During the transactions, especially the foreign (US primarily) business, the exchange rate is very important because it affects the profit in Canadian dollar directly. Therefore, Inshore sets a bottom line for the exchange rate before calculating expected profit. If the actual exchange rate goes under this bottom line, then they may not make a profit on the transaction.

(to be continued...)
Note: All data presented in this Appendix have been disguised and do not reflect the actual business performance of *Inshore Fisheries*. 
Appendix C

The Inventory Report

This is a printed-screen of the inventory report that is designed for Inshore Fisheries. It contains all information described in Table 3: Sample Inventory Report Design. Inshore keeps the inventory of all the product items in order to keep track of the availability for next customer order. It can serve as a measure which tells when it is the time to produce which product item. It also records the production date, ordered date and the customer name.

INSHORE FISHERIES LTD.

<table>
<thead>
<tr>
<th>Inventory Statement</th>
<th>Date: 4/26/2002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Species</td>
<td>ItemNo</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1B1</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1B2</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1B3</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1B4</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1S1</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1S2</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1S3</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA1S4</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2B1</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2B2</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2B3</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2B4</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2S1</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2S2</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2S3</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA2S4</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3B1</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3B2</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3B3</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3B4</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3S1</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3S2</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3S3</td>
</tr>
<tr>
<td>Haddock</td>
<td>HA3S4</td>
</tr>
</tbody>
</table>

INSHORE FISHERIES LTD.

Middle West Pubnico,
Yarmouth County, Nova Scotia
B0W 2M0

ADDRESS QUESTIONS TO
Yvon d’Entremont
(902) 782-2522

Note: All data presented in this Appendix has been disguised and does not reflect the actual business performance of Inshore Fisheries.
Appendix D

The Income Statement

This is a printed-screen of the income statement that is designed for Inshore Fishery. It contains all information described in Table 4: Sample Income Statement Design. This statement measures the profitability of the business after a certain period. It shows an overall performance situation of the company.

<table>
<thead>
<tr>
<th>Operating Revenue</th>
<th>Year to Date Amount</th>
<th>% of Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fresh</td>
<td>3,192.30</td>
<td>2.27%</td>
</tr>
<tr>
<td>Frozen</td>
<td>157,417.25</td>
<td>98.62%</td>
</tr>
<tr>
<td>Slab</td>
<td>1,500.25</td>
<td>0.38%</td>
</tr>
<tr>
<td>Other</td>
<td>211.02</td>
<td>0.17%</td>
</tr>
<tr>
<td>Total Operating Revenue</td>
<td>162,119.83</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operating Costs</th>
<th>Year to Date Amount</th>
<th>% of Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost of Fish</td>
<td>27,610.60</td>
<td>16.88%</td>
</tr>
<tr>
<td>Overhead</td>
<td>5,640.00</td>
<td>3.44%</td>
</tr>
<tr>
<td>Freight</td>
<td>5,604.56</td>
<td>3.38%</td>
</tr>
<tr>
<td>Paying</td>
<td>5,606.71</td>
<td>3.46%</td>
</tr>
<tr>
<td>Subtotal</td>
<td>40,052.88</td>
<td>24.60%</td>
</tr>
<tr>
<td>Miscellaneous Costs</td>
<td>145.00</td>
<td>0.09%</td>
</tr>
<tr>
<td>Labour Costs</td>
<td>20,130.64</td>
<td>12.93%</td>
</tr>
<tr>
<td>Total Operating Cost</td>
<td>60,339.42</td>
<td>37.46%</td>
</tr>
<tr>
<td>Net Operating Income</td>
<td>91,779.43</td>
<td>57.44%</td>
</tr>
</tbody>
</table>

Other Expenses

<table>
<thead>
<tr>
<th>License &amp; Other Fixed</th>
<th>Year to Date Amount</th>
<th>% of Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depreciation</td>
<td>1,340.00</td>
<td>0.09%</td>
</tr>
<tr>
<td>Repairs</td>
<td>251.00</td>
<td>0.14%</td>
</tr>
<tr>
<td>Insurance</td>
<td>451.00</td>
<td>0.28%</td>
</tr>
<tr>
<td>Interest Expense</td>
<td>100.00</td>
<td>0.06%</td>
</tr>
<tr>
<td>Total Other Expenses</td>
<td>1,942.00</td>
<td>0.12%</td>
</tr>
<tr>
<td>Total Cost</td>
<td>82,309.02</td>
<td>51.27%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Profit (before income taxes)</th>
<th>Year to Date Amount</th>
<th>% of Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Income Tax Payable (50%)</td>
<td>45,799.25</td>
<td>28.81%</td>
</tr>
</tbody>
</table>

NET PROFIT (LOSS) | 45,799.25 | 28.81%

Note: All data presented in this Appendix have been disguised and do not reflect the actual business performance of Inshore Fisheries.
Appendix E

Source Code of the APMS

This appendix presents the entire source code of the APMS model developed and applied for the Inshore Fisheries case study. The code is written in Visual Basic Version 6.0 from Microsoft.

The code contains approximately 2600 lines of code. Code functionality includes tasks for the user interface GUI and the linkages of input and output of the system. The executable file has size of 320KByte. The code runs on a Windows 98, ME, 2000, or XP operating systems.

The code also contains documentation throughout to describe the functionality of each frame and window. Further details and access to this course code are available from the author.
'Code for start frame
Option Explicit

Private Sub cmdStart_Click()
  'frmDescription.Show vbCrLf, Me
  Load frmDescription
  frmDescription.Show
  Unload Me
End Sub

Private Sub Form_Load()
  lblToday.Caption = "Today is: " & Date
End Sub

'Code for the frame of Description
Option Explicit
Public cn As ADOConnection

Private Sub cmdExit_Click()
  cn.Close
  Set cn = Nothing
End
End Sub

Private Sub cmdIncome_Click()
  frmIncome.Show vbCrLf, Me
End Sub

Private Sub cmdInven_Click()
  frmInvent.Show vbCrLf, Me
End Sub

Private Sub cmdInvoice_Click()
  frmInvoice.Show vbCrLf, Me
End Sub

Private Sub cmdOrder_Click()
  frmOrder.Show vbCrLf, Me
End Sub

Private Sub cmdProd_Click()
  frmProd.Show vbCrLf, Me
End Sub

Private Sub cmdWS_Click()
  frmWS.Show vbModal, Me
End Sub

Private Sub Form_Load()
  Set cn = New ADOConnection
  'cn.Open "Provider =
  Microsoft.Jet.OLEDB.4.0;Data
  Source=C:\Documents and
  Settings\Guowei\Desktop\ADMS\ADMS.mdb
  ;Persist Security Info=False"
  cn.Open "Provider =
  Microsoft.Jet.OLEDB.4.0;Data
  Source=ADMS.mdb;Persist Security
  Info=False"
End Sub

'Code for frame of Production
Option Explicit
Public sSpecies As String
'sSpecies
'Set sSpecies = "hello wei"

'Private Sub cmbSpecies_Change()
  'If cmbSpecies.DataChanged = True Then
  '  sSpecies = cmbSpecies.Index
  '  End If
'  'End Sub

Public bChange As Boolean

Private Sub cmdSpecies_Click()
  If cmbSpecies.DataChanged = True Then
    sSpecies = cmbSpecies.Text
    End If
End Sub

Private Sub cmdExit_Click()
  Unload Me
End Sub

Private Sub cmdProcess_Click()
  'Dim iProdTemp As Variant
iProdTemp = Shell("c:\Program Files\Microsoft Office\Office\EXCEL " & "D:\\EE\EE\Sys\Thesis\ADMS\WorkSheetTempl ate.exe", 1)
'ActivateProd
Unload Me
frmProdInput.Show \vbModal, Me
End Sub

Private Sub cmdSpecies_Click()
Unload Me
frmSpecies.Show \vbModal, Me
End Sub

Private Sub Form_Load()
bChange = False
cmbSpecies.AddItem ("Catfish")
cmbSpecies.AddItem ("Cod")
cmbSpecies.AddItem ("Flounder")
cmbSpecies.AddItem ("Haddock")
cmbSpecies.AddItem ("OceanPerch")
cmbSpecies.AddItem ("Pollock")
cmbSpecies.AddItem ("Turbot")

If bChange = True Then
cmbSpecies.Text = sSpecies
Else
sSpecies = cmbSpecies.Text
End If
End Sub

'Code for frame of Species Production Data
Option Explicit

Private Sub cmdExit_Click()
Unload Me
End Sub

Private Sub cmdOk_Click()
dataSpecies.Recordset.MoveLast
dataSpecies.Refresh
Unload Me
frmProd.Show \vbModal, Me

frmProd.cmbSpecies.Text =
frmProd.sSpecies =
End Sub

Private Sub Form_Load()
dataSpecies.ToolTipText = "Use the arrow key to switch species"
txtSpecies.ToolTipText = "Name of the Species"
txtArea.ToolTipText = "Name of that Area"
txtVessel.ToolTipText = "Name of the Vessel"
txtSource.ToolTipText = "Where the fish come from"
txtManWage.ToolTipText = "Canadian dollars per pound"
txtManOT.ToolTipText = "Canadian dollars per pound"
txtWomanOT.ToolTipText = "Canadian dollars per pound"
txtFishCost.ToolTipText = "Canadian dollars per pound"
txtOHCost.ToolTipText = "Canadian dollars per pound"
txtLaborCost.ToolTipText = "Canadian dollars per pound"
txtFreshFreight.ToolTipText = "Canadian dollars per pound"
txtFrozenFreight.ToolTipText = "Canadian dollars per pound"
txtFreshPack.ToolTipText = "Canadian dollars per pound"
txtFrozenPack.ToolTipText = "Canadian dollars per pound"
txtSlabPack.ToolTipText = "Canadian dollars per pound"

Select Case frmProd.sSpecies
Case "Catfish" dataSpecies.Recordset.MoveFirst
Case "Cod" dataSpecies.Recordset.MoveNext

Select Case frmProd.sSpecies
Case "Catfish" dataSpecies.Recordset.MoveFirst
Case "Cod" dataSpecies.Recordset.MoveNext

Case "Flounder"
dataSpecies.Recordset.MoveNext
Case "Haddock"
dataSpecies.Recordset.MoveNext
Case "OceanPerch"
dataSpecies.Recordset.MoveNext
Case "Pollock"
dataSpecies.Recordset.MoveNext
Case "Turbot"
dataSpecies.Recordset.MoveNextLast
End Select
frmProd.bChange = True
End Sub

' Code for frame of Input Production Details
Option Explicit
Public exist As Boolean
Private Sub cmdBack_Click()
Unload Me
frmProd.Show vbModal, Me
End Sub

Private Sub cmdExit_Click()
Unload Me
End Sub

Private Sub cmdFresh_Click()
Unload Me
frmFresh.Show vbModal, Me
End Sub

Private Sub cmdFrozen_Click()
Unload Me
frmFrozen.Show vbModal, Me
End Sub

Private Sub cmdSlabs_Click()
Unload Me
frmSlab.Show vbModal, Me
End Sub

Private Sub Form_Load()
'cn.Close

Set rs1 = Nothing
Set rs2 = Nothing
Set rs3 = Nothing
'Set cn = Nothing

cmdFresh.Enabled = False
cmdFrozen.Enabled = False
cmdSlabs.Enabled = False
'lblSpecies.Caption = "Species: " 
frmProd.sSpecies
val_species.Caption = frmProd.sSpecies
exist = False
End Sub

Private Sub txtProduction_KeyDown(KeyCode
As Integer, Shift As Integer)
cmdFresh.Enabled = True
cmdFrozen.Enabled = True
cmdSlabs.Enabled = True
End Sub

'Code for frame of Fresh
Option Explicit

Private Sub cmdExit_Click()
Unload Me
End Sub

Private Sub cmdSubmit_Click()

Dim cmd As String
Dim fconnection As Connection
Dim tmp As Boolean

tmp = False

Set fconnection = New Connection
fconnection.Open
"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=ADMS.mdb;Persist Security
Info=False"
If Text1.Text <> "" Then

If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + "," + "Date()"
cmd = cmd + "," +
frmProdInput.txtProdution.Text + ""
frmProdInput.val_species.Caption + ""
cmd = cmd + "," +
frmProdInput.cboVessel.Text + ""
cmd = cmd + "," +
frmProdInput.cboSource.Text + ""
cmd = cmd + "," +
frmProdInput.cboArea.Text + ""
cmd = cmd + "," + frmProdInput.txtRate.Text
cmd = cmd + "," +
frmProdInput.txtDayX.Text
cmd = cmd + "," +
frmProdInput.txtActualX.Text
cmd = cmd + "," +
frmProdInput.txtBaseX.Text
cmd = cmd + ")"
fconnection.Execute cmd
frmProdInput.exist = True
End If

cmd = "insert into Fresh values ("
cmd = cmd + "," + val_prod_no.Caption + ""
cmd = cmd + "," + Label1.Caption + ""
cmd = cmd + "," +
cmd = cmd + Text1.Text
cmd = cmd + "")"
fconnection.Execute cmd
End If

If Text2.Text <> "" Then
If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + "," + "Date()"
cmd = cmd + "," +
frmProdInput.txtProdution.Text + ""
cmd = cmd + "," +
frmProdInput.val_species.Caption + ""

```
cmd = cmd + "," + frmProdInput.cboVessel.Text + ""
352 cmd = cmd + "," + frmProdInput.cboSource.Text + ""
353 cmd = cmd + "," + frmProdInput.cboArea.Text + ""
354 cmd = cmd + "," + frmProdInput.cboRate.Text
355 cmd = cmd + "," + frmProdInput.txtDayX.Text
356 cmd = cmd + "," + frmProdInput.valSource.Caption + ""
357 cmd = cmd + "," + Label2.Caption + ""
358 cmd = cmd + "," + Label3.Caption + ""
359 cmd = cmd + "," + tmp
360 connection.Execute cmd
361 frmProdInput.exist = True
362 End If
363 If Text4.Text <> "" Then
364 If frmProdInput.exist = False Then
365 cmd = "insert into Production values ("
366 cmd = cmd + "" + frmProdInput.txtProduction.Text + ""
367 cmd = cmd + "" + frmProdInput.val_species.Caption + ""
368 cmd = cmd + "," + "Date()"
369 cmd = cmd + "," + ""
370 cmd = cmd + "," + frmProdInput.cboVessel.Text + ""
371 cmd = cmd + "," + frmProdInput.cboSource.Text + ""
372 cmd = cmd + "" + frmProdInput.cboArea.Text + ""
373 cmd = cmd + "," + frmProdInput.cboRate.Text
374 cmd = cmd + "," + frmProdInput.txtDayX.Text
375 End If
376 End If
377 If Text3.Text <> "" Then
378 If frmProdInput.exist = False Then
379 cmd = "insert into Production values ("
380 cmd = cmd + "" + frmProdInput.txtProduction.Text + ""
381 cmd = cmd + "" + frmProdInput.val_species.Caption + ""
382 cmd = cmd + "" + "Date()"
383 cmd = cmd + "," + ""
384 cmd = cmd + "," + ""
385 cmd = cmd + "," + frmProdInput.cboVessel.Text + ""
386 cmd = cmd + "," + frmProdInput.cboSource.Text + ""
387 cmd = cmd + "," + frmProdInput.cboArea.Text + ""
388 cmd = cmd + "" + frmProdInput.cboRate.Text
389 cmd = cmd + "," + frmProdInput.txtDayX.Text
390 cmd = cmd + "" + frmProdInput.txtActualX.Text
391 cmd = cmd + "" + frmProdInput.txtBaseX.Text
392 cmd = cmd + "," + frmProdInput.txtBaseX.Text
393 cmd = cmd + "" + frmProdInput.txtDayX.Text
394 cmd = cmd + "" + frmProdInput.txtActualX.Text
395 cmd = cmd + "" + frmProdInput.txtBaseX.Text
396 cmd = cmd + "")"
397 connection.Execute cmd
398 frmProdInput.exist = True
399 End If
400 cmd = "insert into Fresh values ("
401 cmd = cmd + "" + val_prod_no.Caption + ""
402 cmd = cmd + "" + Label3.Caption + ""
403 cmd = cmd + "" + tmp
404 cmd = cmd + "")"
405 connection.Execute cmd
406 tmp = True
407 End If
408
409 If Text4.Text <> "" Then
410 If frmProdInput.exist = False Then
411 cmd = "insert into Production values ("
412 cmd = cmd + "" + frmProdInput.txtProduction.Text + ""
413 cmd = cmd + "" + frmProdInput.val_species.Caption + ""
414 cmd = cmd + "" + "Date()"
415 cmd = cmd + "," + ""
416 cmd = cmd + "" + ""
417 cmd = cmd + "" + frmProdInput.cboVessel.Text + ""
418 cmd = cmd + "" + frmProdInput.cboSource.Text + ""
419 cmd = cmd + "" + frmProdInput.cboArea.Text + ""
420 cmd = cmd + "" + frmProdInput.cboRate.Text
421 cmd = cmd + "" + frmProdInput.txtDayX.Text
422 cmd = cmd + "" + frmProdInput.txtActualX.Text
423 cmd = cmd + "" + frmProdInput.txtBaseX.Text
424 cmd = cmd + "" + frmProdInput.txtBaseX.Text
425 cmd = cmd + "" + frmProdInput.txtDayX.Text
426 cmd = cmd + "" + frmProdInput.txtActualX.Text
427 cmd = cmd + "" + frmProdInput.txtBaseX.Text
428 cmd = cmd + "")"
429 connection.Execute cmd
430 frmProdInput.exist = True
431 End If
432 If Text3.Text <> "" Then
433 If frmProdInput.exist = False Then
434 cmd = "insert into Production values ("
435 cmd = cmd + "" + val_prod_no.Caption + ""
436 cmd = cmd + "" + Label4.Caption + ""
437 cmd = cmd + "" + tmp
438 cmd = cmd + "")"
65```
cmd = cmd + Text4.Text
440 cmd = cmd + ")"
441 fconnection.Execute cmd
442 tmp = True
443 End If
444
445 If Text5.Text <> "" Then
446 If frmProdInput.exist = False Then
447 cmd = "insert into Production values ("
448 cmd = cmd + "" +
449 frmProdInput.txtProduction.Text + ""
450 cmd = cmd + ", " + "Date()"
451 cmd = cmd + ", " +
452 frmProdInput.val_species.Caption + ""
453 cmd = cmd + ", " +
454 frmProdInput.cboVessel.Text + ""
455 cmd = cmd + ", " +
456 frmProdInput.cboSource.Text + ""
457 cmd = cmd + ", " +
458 frmProdInput.cboArea.Text + ""
459 cmd = cmd + ", " + frmProdInput.txtRate.Text
460 cmd = cmd + ", " + frmProdInput.txtDayX.Text
461 cmd = cmd + ", " + frmProdInput.txtActualX.Text
462 cmd = cmd + ", " + frmProdInput.txtBaseX.Text
463 cmd = cmd + ")"
464 fconnection.Execute cmd
465 frmProdInput.exist = True
466 End If
467 cmd = "insert into Production values ("
468 cmd = cmd + "+ val_prod_no.Caption + ""
469 cmd = cmd + ", " + Label5.Caption + ""
470 cmd = cmd + ", " +
471 cmd = cmd + Text5.Text
472 cmd = cmd + ")"
473 fconnection.Execute cmd
474 tmp = True
475 End If
476
477 If Text6.Text <> "" Then
478 If frmProdInput.exist = False Then
479 cmd = "insert into Production values ("
480 cmd = cmd + "" + frmProdInput.txtProduction.Text
481 + ""
482 cmd = cmd + ", " + "Date()"
483 cmd = cmd + ", " +
484 frmProdInput.val_species.Caption + ""
485 cmd = cmd + ", " + frmProdInput.cboVessel.Text
486 + ""
487 cmd = cmd + ", " + frmProdInput.cboSource.Text
488 + ""
489 cmd = cmd + ", " + frmProdInput.cboArea.Text + ""
490 cmd = cmd + "" + frmProdInput.txtRate.Text
491 cmd = cmd + "" + frmProdInput.txtDayX.Text
492 cmd = cmd + "" + frmProdInput.txtActualX.Text
493 cmd = cmd + "" + frmProdInput.txtBaseX.Text
494 cmd = cmd + ")"
495 fconnection.Execute cmd
496 frmProdInput.exist = True
497 End If
498 cmd = "insert into Fresh values ("
500 cmd = cmd + "" + val_prod_no.Caption + ""
501 cmd = cmd + ", " + Label6.Caption + ""
502 cmd = cmd + "" +
503 cmd = cmd + Text6.Text
504 cmd = cmd + ")"
505 fconnection.Execute cmd
506 tmp = True
507 End If
508
509 If Text7.Text <> "" Then
510 If frmProdInput.exist = False Then
511 cmd = "insert into Production values ("
512 cmd = cmd + "" +
513 frmProdInput.txtProduction.Text + ""
514 cmd = cmd + ", " + "Date()"
515 cmd = cmd + "" +
516 frmProdInput.val_species.Caption + ""
517 cmd = cmd + ", " +
518 frmProdInput.cboVessel.Text + ""
519 cmd = cmd + "" +
520 frmProdInput.cboSource.Text + ""
521 cmd = cmd + "" +
522 frmProdInput.cboArea.Text + ""
523 cmd = cmd + "" + frmProdInput.txtRate.Text
524 cmd = cmd + "" +
525 frmProdInput.txtDayX.Text
526  cmd = cmd + "," +
527  frmProdInput.txtActualX.Text
528  cmd = cmd + "," +
529  frmProdInput.txtBaseX.Text
530  cmd = cmd + ")"
531  fconnection.Execute cmd
532  frmProdInput.exist = True
533  End If
534  cmd = "insert into Fresh values ("
535  cmd = cmd + ";" + val_prod_no.Caption + ";"
536  cmd = cmd + "," + Label7.Caption + ";"
537  cmd = cmd + "," +
538  cmd = cmd + Text7.Text
539  cmd = cmd + ");"
540  fconnection.Execute cmd
541  tmp = True
542  End If
543  If tmp = True Then
544  frmProdInput.exist = True
545  frmProdInput.txtProduction.Enabled = False
546  frmProdInput.cboArea.Enabled = False
547  frmProdInput.cboSource.Enabled = False
548  frmProdInput.cboVessel.Enabled = False
549  frmProdInput.txtActualX.Enabled = False
550  frmProdInput.txtBaseX.Enabled = False
551  frmProdInput.txtDayX.Enabled = False
552  frmProdInput.cmdLoad.Enabled = False
553  frmProdInput.cmdRate.Enabled = False
554  frmProdInput.txtRate.Enabled = False
555  End If
556  Unload Me
557  End Sub
558  Private Sub Form_Load()
559  val_prod_no.Caption =
560  frmProdInput.txtProduction.Text
561  val_species.Caption = frmProdsSpecies
562  "cmdSubmit.Enabled = False
563  End Sub
564  Private Sub Text1_Change()
565  "cmdSubmit.Enabled = True
566  End Sub
567
570  'Code for frame Frozen
571  Option Explicit
572  Dim lab_array(24) As Label
573  Dim tmp2 As Integer
574  Dim cmd As String
575  Dim fconnection As Connection
576  Dim rs As Recordset
577  Dim pconnection As Connection
578  Dim tmp As Boolean
579  tmp = False
580  Set fconnection = New Connection
581  fconnection.Open
582  "Provider=Microsoft.Jet.OLEDB.4.0;Data
583  Source=ADMS.mdb;Persist Security
584  Info=False"
585  If Text1.Text <> "" Then
586  If frmProdInput.exist = False Then
587  cmd = "insert into Production values ("
588  cmd = cmd + ";" +
589  cmd = cmd + ";" +
590  cmd = cmd + ";" +
591  cmd = cmd + ";" +
592  cmd = cmd + "" +
593  cmd = cmd + "" +
594  cmd = cmd + "" +
595  cmd = cmd + "" +
596  cmd = cmd + "" +
597  cmd = cmd + "" +
598  cmd = cmd + "" +
599  cmd = cmd + ";" +
600  cmd = cmd + "" +
601  cmd = cmd + "" +
602  cmd = cmd + "" +
603  cmd = cmd + "" +
604  cmd = cmd + "" +
605  cmd = cmd + "" +
606  cmd = cmd + "" +
607  cmd = cmd + ");"
608  frmProdInput.val_species.Caption + ""
609  cmd = cmd + ";" +
610  frmProdInput.cboVessel.Text + ""
611  cmd = cmd + ";" +
612  frmProdInput.txtActualX.Text
613  cmd = cmd + ");"
614
67
fconnection.Execute cmd
615 frmProdInput.exist = True
616 End If
617 cmd = "insert into Frozen values ("
618 cmd = cmd + "" + val_prod_no.Caption + ""
619 cmd = cmd + "," + Label1.Caption + ""
620 cmd = cmd + ","
621 cmd = cmd + Text1.Text
622 cmd = cmd + ")"
623 fconnection.Execute cmd
624 Set rs = New Recordset
625 cmd = "select * from Inventory where spieces ="
626 = ""
627 cmd = cmd + frmProd.sSpecies + "" and "
628 cmd = cmd + "ItemNo = "
629 cmd = cmd + lab_array(0).Caption + ""
630 rs.Open cmd, fconnection,
631 adOpenForwardOnly, adLockReadOnly
632 rs.MoveFirst
633 tmp2 = rs("Quantity")
634 rs.Close
635 Set rs = Nothing
636 Text25.Text = 0
637 Text25.Text = Text1.Text + tmp2
638 cmd = "update Inventory set Quantity ="
639 cmd = cmd + Text25.Text
640 cmd = cmd + " where Spieces ="
641 cmd = cmd + frmProd.sSpecies + "" and "
642 cmd = cmd + "ItemNo = "
643 cmd = cmd + lab_array(0).Caption + ""
644 fconnection.Execute cmd
645 tmp = True
646 End If
647
648 If Text2.Text <> "" Then
649 If frmProdInput.exist = False Then
650 cmd = "insert into Production values ("
651 cmd = cmd + ""
652 frmProdInput.txtProduction.Text + ""
653 cmd = cmd + "," + "Date()"
654 cmd = cmd + "," + " +
655 frmProdInput.val_species.Caption + ""
656 cmd = cmd + "," + " +
657 frmProdInput.cboVessel.Text + ""
658 cmd = cmd + "," +
659 frmProdInput.cboSource.Text + ""
660 cmd = cmd + "," +
661 frmProdInput.cboArea.Text + ""
662 cmd = cmd + "," + frmProdInput.txtRate.Text
663 cmd = cmd + "," +
664 frmProdInput.txtDayX.Text
665 cmd = cmd + "," +
666 frmProdInput.txtActualX.Text
667 cmd = cmd + "," +
668 frmProdInput.txtBaseX.Text
669 cmd = cmd + ")"
670 fconnection.Execute cmd
671 frmProdInput.exist = True
672 End If
673 cmd = "insert into Frozen values ("
674 cmd = cmd + "" + val_prod_no.Caption + ""
675 cmd = cmd + "," + Label2.Caption + ""
676 cmd = cmd + ","
677 cmd = cmd + Text2.Text
678 cmd = cmd + ")"
679 fconnection.Execute cmd
680 Set rs = New Recordset
681 cmd = "select * from Inventory where spieces ="
682 = ""
683 cmd = cmd + frmProd.sSpecies + "" and "
684 cmd = cmd + "ItemNo = "
685 cmd = cmd + lab_array(1).Caption + ""
686 rs.Open cmd, fconnection,
687 adOpenForwardOnly, adLockReadOnly
688 rs.MoveFirst
689 tmp2 = rs("Quantity")
690 rs.Close
691 Set rs = Nothing
692 Text25.Text = 0
693 Text25.Text = Text2.Text + tmp2
694 cmd = "update Inventory set Quantity ="
695 cmd = cmd + Text25.Text
696 cmd = cmd + " where Spieces ="
697 cmd = cmd + frmProd.sSpecies + "" and "
698 cmd = cmd + "ItemNo = "
699 cmd = cmd + lab_array(1).Caption + ""
700 fconnection.Execute cmd
701 tmp = True
End If

If Text3.Text <> "" Then
    If frmProdInput.exist = False Then
        cmd = "insert into Production values ("
        cmd = cmd + "" + "" + "" + "" + ""
        frmProdInput.txtProduction.Text + ""
        cmd = cmd + ", " + "" + "" Date()
        cmd = cmd + ", " + ""
        frmProdInput.val_species.Caption + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboVessel.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboSource.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboArea.Text + ""
        cmd = cmd + ", " + frmProdInput.txtRate.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtDayX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtActualX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtBaseX.Text
        cmd = cmd + ", " + ""
        frmProdInput.val_species.Caption + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboVessel.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboSource.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboArea.Text + ""
        cmd = cmd + ", " + frmProdInput.txtRate.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtDayX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtActualX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtBaseX.Text
        cmd = cmd + ", " + ""
        frmProdInput.val_species.Caption + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboVessel.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboSource.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboArea.Text + ""
        cmd = cmd + ", " + frmProdInput.txtRate.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtDayX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtActualX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtBaseX.Text
        cmd = cmd + "")"
        fconnection.Execute cmd
        frmProdInput.exist = True
        Set rs = New Recordset
        cmd = "select * from Inventory where spieces = "
        cmd = cmd + "" + val_prod_no.Caption + ""
        cmd = cmd + "" + Label3.Caption + ""
        cmd = cmd + ", " + ""
        cmd = cmd + "" + Text3.Text
        cmd = cmd + "")"
        fconnection.Execute cmd
        Set rs = Nothing
        cmd = "update Inventory set Quantity ="
        cmd = cmd + ", " + ""
        cmd = cmd + "" where Spieces = "
        cmd = cmd + "" + frm_Pb_sSpecies + "" and "
        cmd = cmd + "" + ""
        cmd = cmd + "" + ""
        cmd = cmd + "" + ""
        cmd = cmd + "")"
        fconnection.Execute cmd
        tmp = True
        End If

If Text4.Text <> "" Then
    If frmProdInput.exist = False Then
        cmd = "insert into Production values ("
        cmd = cmd + "" + "" + "" + "" + ""
        frmProdInput.txtProduction.Text + ""
        cmd = cmd + ", " + "" + "" Date()
        cmd = cmd + ", " + ""
        frmProdInput.val_species.Caption + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboVessel.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboSource.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboArea.Text + ""
        cmd = cmd + ", " + frmProdInput.txtRate.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtDayX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtActualX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtBaseX.Text
        cmd = cmd + ", " + ""
        frmProdInput.val_species.Caption + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboVessel.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboSource.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboArea.Text + ""
        cmd = cmd + ", " + frmProdInput.txtRate.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtDayX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtActualX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtBaseX.Text
        cmd = cmd + ", " + ""
        frmProdInput.val_species.Caption + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboVessel.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboSource.Text + ""
        cmd = cmd + ", " + ""
        frmProdInput.cboArea.Text + ""
        cmd = cmd + ", " + frmProdInput.txtRate.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtDayX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtActualX.Text
        cmd = cmd + ", " + ""
        frmProdInput.txtBaseX.Text
        cmd = cmd + "")"
        fconnection.Execute cmd
        frmProdInput.exist = True
        rs.Open cmd, fconnection, adOpenForwardOnly, adLockReadOnly
        rs.MoveNext
        tmp2 = rs("Quantity")
cmd = cmd + ")"
fcnconnection.Execute cmd
rs = New Recordset
cmd = "select * from Inventory where Spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(3).Caption + ""
rs.Open cmd, fconnection, adOpenForwardOnly, adLockReadOnly
tmp2 = rst("Quantity")
rs.Close
Set rs = Nothing
Text25.Text = 0
cmd = "update Inventory set Quantity ="
cmd = cmd + " where Spieces = "
Text25.Text = Text25.Text + cmd + frmProd.sSpecies + " and "
Text25.Text = Text25.Text + cmd + "ItemNo = "
fcnconnection.Execute cmd
tmp = True
End If
If Text5.Text <> "" Then
If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + ""
frmProdInput.txtProduction.Text = ""
cmd = cmd + ", " + "Date()"
cmd = cmd + ", " + frmProdInput.val_species.Caption + ""
cmd = cmd + ", " + frmProdInput.cboVessel.Text + ""
cmd = cmd + ", " + frmProdInput.cboSource.Text + ""
cmd = cmd + ", " + frmProdInput.cboArea.Text + ""
cmd = cmd + ", " + frmProdInput.txtRate.Text
cmd = cmd + ", " + frmProdInput.txtDayX.Text
frmProdInput.txtActualX.Text = cmd + "", " +
frmProdInput.txtBaseX.Text = cmd + "", " +
frmProdInput.exist = True
End If
cmd = "insert into Frozen values ("
cmd = cmd + ""
valProdNo.Caption = ""
cmd = cmd + ", " + Label5.Caption + ""
cmd = cmd + ", "
cmd = cmd + Text5.Text
cmd = cmd + ")"
fcnconnection.Execute cmd
Set rs = New Recordset
cmd = "select * from Inventory where Spieces = "
Text25.Text = Text25.Text + cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(4).Caption + ""
rn.Open cmd, fconnection, adOpenForwardOnly, adLockReadOnly
rs.MoveFirst
tmp2 = rst("Quantity")
rs.Close
Set rs = Nothing
Text25.Text = 0
Text25.Text = Text5.Text + tmp2
cmd = "update Inventory set Quantity ="
Text25.Text = Text25.Text + cmd + " where Spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(4).Caption + ""
fcnconnection.Execute cmd
tmp = True
End If
If Text6.Text <> "" Then
If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + ""
frmProdInput.txtProduction.Text = ""
End If
877  cmd = cmd + "," + "Date()"
878  cmd = cmd + "," + " +
879  frmProdInput.val_species.Caption + ""
880  cmd = cmd + "," + " +
881  frmProdInput.cboVessel.Text + ""
882  cmd = cmd + "," + " +
883  frmProdInput.cboSource.Text + ""
884  cmd = cmd + "," + " +
885  frmProdInput.cboArea.Text + ""
886  cmd = cmd + "," + frmProdInput.txtRate.Text
887  cmd = cmd + "," + " +
888  frmProdInput.txtDayX.Text
889  cmd = cmd + "," + " +
890  frmProdInput.txtActualX.Text
891  cmd = cmd + "," + " +
892  frmProdInput.txtBaseX.Text
893  cmd = cmd + ")"
894  fconnection.Execute cmd
895  frmProdInput.exist = True
896  End If
897  cmd = "insert into Frozen values ("
898  cmd = cmd + "" + val_prod_no.Caption + ""
899  cmd = cmd + "," + Label1.Caption + ""
900  cmd = cmd + "," + " +
901  cmd = cmd + Text6.Text
902  cmd = cmd + ")"
903  fconnection.Execute cmd
904  Set rs = New Recordset
905  cmd = "select * from Inventory where species = "
906  = ""
907  cmd = cmd + frmProd.sSpecies + " and "
908  cmd = cmd + "itemNo = "
909  cmd = cmd + lab_array(5).Caption + ""
910  rs.Open cmd, fconnection,
911  adOpenForwardOnly, adLockReadOnly
912  rs.MoveFirst
913  tmp2 = rs("Quantity")
914  rs.Close
915  Set rs = Nothing
916  Text25.Text = 0
917  Text25.Text = Text6.Text + tmp2
918  cmd = "update Inventory set Quantity ="
919  cmd = cmd + Text25.Text
920  cmd = cmd + ", where Species = "
921  cmd = cmd + frmProd.sSpecies + " and "
922  cmd = cmd + "itemNo = "
923  cmd = cmd + lab_array(5).Caption + ""
924  fconnection.Execute cmd
925  tmp = True
926  End If
927  If Text7.Text <> "" Then
928  If frmProdInput.exist = False Then
929  cmd = "insert into Production values ("
930  cmd = cmd + ""
931  cmd = cmd + ""
932  cmd = cmd + ""
933  cmd = cmd + ""
934  cmd = cmd + ""
935  cmd = cmd + ""
936  cmd = cmd + ""
937  cmd = cmd + ""
938  cmd = cmd + ""
939  cmd = cmd + ""
940  cmd = cmd + ""
941  cmd = cmd + ""
942  cmd = cmd + ""
943  cmd = cmd + ""
944  cmd = cmd + ""
945  cmd = cmd + ""
946  cmd = cmd + ""
947  cmd = cmd + ""
948  cmd = cmd + ""
949  cmd = cmd + "")"
950  fconnection.Execute cmd
951  frmProdInput.exist = True
952  End If
953  cmd = "insert into Frozen values ("
954  cmd = cmd + "" + val_prod_no.Caption + ""
955  cmd = cmd + "" + Label7.Caption + ""
956  cmd = cmd + ""
957  cmd = cmd + Text7.Text
958  cmd = cmd + ")"
959  fconnection.Execute cmd
960  Set rs = New Recordset
961  cmd = "select * from Inventory where species = "
962  = ""
963  cmd = cmd + frmProd.sSpecies + " and "
964  cmd = cmd + "itemNo = "
cmd = cmd + "insert into Frozen values ("
1009 cmd = cmd + "," + val_prod_no.Caption + ""
1010 cmd = cmd + "," + Lab18.Caption + ""
1011 cmd = cmd + "," + Label8.Caption + ""
1012 cmd = cmd + ","
1013 cmd = cmd + Text8.Text
1014 cmd = cmd + ")"
1015 fconnection.Execute cmd
1016 Set rs = New Recordset
1017 cmd = "select * from Inventory where spieces = "
1018 cmd = cmd + frmProd.sSpecies + " and "
1019 cmd = cmd + "itemNo = "
1020 cmd = cmd + lab_array(7).Caption + ""
1021 rs.Open cmd, fconnection, adOpenForwardOnly,
1022 adLockReadOnly
1023 rs.MoveFirst
1024 tmp2 = rs("Quantity")
1025 rs.Close
1026 Set rs = Nothing
1027 Text25.Text = 0
1028 Text25.Text = Text8.Text + tmp2
1029 cmd = "update Inventory set Quantity ="
1030 cmd = cmd + Text25.Text
1031 cmd = cmd + " where Spieces = "
1032 cmd = cmd + frmProd.sSpecies + " and "
1033 cmd = cmd + "ItemNo = "
1034 cmd = cmd + lab_array(7).Caption + ""
1035 fconnection.Execute cmd
1036 tmp = True
1037 End If
1038 If Text8.Text <> "" Then
1039 If frmProdInput.exist = False Then
1040 If frmProdInput.exist = False Then
1041 cmd = "insert into Production values ("
1042 cmd = cmd + Text8.Text + ""
1043 cmd = cmd + "," + "Date()"
1044 cmd = cmd + "," + "Date()"
1045 cmd = cmd + "," + 
1046 cmd = cmd + "," + 
1047 cmd = cmd + "," + frmProdInput.cboVessel.Text + ""
1048 cmd = cmd + "," + frmProdInput.cboVessel.Text + ""
1049 cmd = cmd + "," + frmProdInput.cboSource.Text + ""
1050 cmd = cmd + "," + frmProdInput.cboArea.Text + ""
1051 cmd = cmd + "," + frmProdInput.txtBaseX.Text
1052 cmd = cmd + ")"
cmd = cmd + "," + frmProdInput.txtDayX.Text
1054 cmd = cmd + "," + frmProdInput.txtActualX.Text
1055 cmd = cmd + "," + frmProdInput.txtBaseX.Text
1056 cmd = cmd + ")"
1057 fconnection.Execute cmd
1058 frmProdInput.exist = True
1059 End If
1060 cmd = "insert into Frozen values ("
1061 cmd = cmd + "," + val_prod_no.Caption + ""
1062 cmd = cmd + "," + Label9.Caption + ""
1063 cmd = cmd + ","
1064 cmd = cmd + Text9.Text
1065 cmd = cmd + ")"
1066 fconnection.Execute cmd
1067 Set rs = New Recordset
1068 cmd = "select * from Inventory where species = "
1069 cmd = cmd + frmProd.sSpecies + " and "
1070 cmd = cmd + "ItemNo = "
1071 cmd = cmd + lab_array(8).Caption + ""
1072 rs.Open cmd, fconnection, True, adOpenForwardOnly, adLockReadOnly
1073 rs.MoveFirst
1074 cmd = cmd + Text10.Text
1075 cmd = cmd + ")"
1076 fconnection.Execute cmd
1077 rs.Close
1078 Text25.Text = 0
1079 cmd = "update Inventory set Quantity ="
1080 cmd = cmd + Text25.Text
1081 cmd = cmd + "," where Spices = "
1082 cmd = cmd + frmProd.sSpecies + " and "
1083 cmd = cmd + "ItemNo = "
1084 cmd = cmd + lab_array(9).Caption + ""
1085 cmd = cmd + lab_array(8).Caption + ""
1086 fconnection.Execute cmd
1087 tmp = True
1088 End If
1089
1090 If Text10.Text <> "" Then
1091 If frmProdInput.exist = False Then
1092 cmd = "insert into Production values ("
1093 cmd = cmd + ""
1094 cmd = cmd + ""
1095 cmd = cmd + ""
1096 cmd = cmd + "," + "Date()"
1097 cmd = cmd + "," + "+
1098 frmProdInput.val_species.Caption + ""
1099 cmd = cmd + "," + "+
1100 frmProdInput.cboVessel.Text + ""
1101 cmd = cmd + "," + "+
1102 frmProdInput.cboSource.Text + ""
1103 cmd = cmd + "," + "+
1104 frmProdInput.cboArea.Text + ""
1105 cmd = cmd + "," + frmProdInput.txtRate.Text
1106 cmd = cmd + "," + "+
1107 frmProdInput.txtDayX.Text
1108 cmd = cmd + "," + "+
1109 frmProdInput.txtActualX.Text
1110 cmd = cmd + "," + "+
1111 frmProdInput.txtBaseX.Text
1112 cmd = cmd + ")"
1113 fconnection.Execute cmd
1114 frmProdInput.exist = True
1115 End If
1116 cmd = "insert into Frozen values ("
1117 cmd = cmd + ""
1118 cmd = cmd + ""
1119 cmd = cmd + ""
1120 cmd = cmd + Text10.Text
1121 cmd = cmd + ")"
1122 fconnection.Execute cmd
1123 Set rs = New Recordset
1124 cmd = "select * from Inventory where species = "
1125 cmd = cmd + ""
1126 cmd = cmd + ""
1127 cmd = cmd + ""
1128 cmd = cmd + ""
1129 rs.Open cmd, fconnection, True, adOpenForwardOnly, adLockReadOnly
1130 rs.MoveFirst
1131 cmd = cmd + Text10.Text
1132 cmd = cmd + ""
1133 rs.Close
1134 Set rs = Nothing
1135 Text25.Text = 0
1136 Text25.Text = Text10.Text + tmp2
1137 cmd = "update Inventory set Quantity ="
1138 cmd = cmd + Text25.Text
1139 cmd = cmd + ""
1140 cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(9).Caption + ""
fcollection.Execute cmd
tmp = True
End If

If Text11.Text <> "" Then
If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + "" +
frmProdInput.txtProduction.Text + ""
cmd = cmd + ", " + "Date()"
cmd = cmd + ", " +
frmProdInput.val_species.Caption + ""
cmd = cmd + "" +
frmProdInput.cboVessel.Text + ""
cmd = cmd + ", " +
frmProdInput.cboSource.Text + ""
cmd = cmd + "" +
frmProdInput.cboArea.Text + ""
cmd = cmd + ", " + frmProdInput.txtRate.Text
cmd = cmd + ", " +
frmProdInput.txtDayX.Text
cmd = cmd + ""
cmd = cmd + ", " +
frmProdInput.txtActualX.Text
cmd = cmd + ""
cmd = cmd + ""
fcollection.Execute cmd
frmProdInput.exist = True
End If

cmd = "insert into Frozen values ("
cmd = cmd + "" + val_prod_no.Caption + ""
cmd = cmd + "" + Label11.Caption + ""
cmd = cmd + ", " +
cmd = cmd + ""
cmd = cmd + ""
fcollection.Execute cmd
Set rs = New Recordset

cmd = "select * from Inventory where Spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(10).Caption + ""
rconnection.Execute cmd
tmp = True
End If
1229  cmd = cmd + "" + val_prod_no.Caption + ""
1230  cmd = cmd + "," + Label12.Caption + ""
1231  cmd = cmd + ""
1232  cmd = cmd + Text12.Text
1233  cmd = cmd + ")"
1234  fconnection.Execute cmd
1235  Set rs = New Recordset
1236  cmd = "select * from Inventory where species ="
1237  = ""
1238  cmd = cmd + frmProd.sSpecies + ";" and "." + ""
1239  cmd = cmd + "," + "ItemNo = ""
1240  cmd = cmd + lab_array(11).Caption + ""
1241  rs.Open cmd, fconnection,
1242  adOpenForwardOnly, adLockReadOnly
1243  rs.MoveFirst
1244  tmp2 = rs("Quantity")
1245  rs.Close
1246  Set rs = Nothing
1247  Text25.Text = 0
1248  Text25.Text = Text12.Text + tmp2
1249  cmd = "update Inventory set Quantity ="
1250  cmd = cmd + Text25.Text
1251  cmd = cmd + "," where Species ="
1252  cmd = cmd + frmProd.sSpecies + ";" and "." + ""
1253  cmd = cmd + "ItemNo = ""
1254  cmd = cmd + lab_array(11).Caption + ""
1255  fconnection.Execute cmd
1256  tmp = True
1257  End If
1258  If Text13.Text <> "" Then
1259  If frmProdInput.exist = False Then
1260  cmd = "insert into Production values ("
1261  cmd = cmd + Text25.Text
1262  cmd = cmd + "," + "Prod.
1263  cmd = cmd + "" + "" + "," + "," + "Date()"
1264  cmd = cmd + "," + "species.Caption + ""
1265  cmd = cmd + "," + ""
1266  cmd = cmd + "" + "" + "" + "" + ""]Caption + ""
1267  cmd = cmd + "" + "" + ""
1268  cmd = cmd + "" + "" + ""
1269  cmd = cmd + "" + "" + ""
1270  cmd = cmd + "" + "" + ""
1271  cmd = cmd + "" + "" + ""
1272  cmd = cmd + "" + "frmProdInput.txtRate.Text
1273  cmd = cmd + "," + "frmProdInput.txtRate.Text
1274  cmd = cmd + "," + ""
1275  frmProdInput.txtDayX.Text
1276  cmd = cmd + "," + ""
1277  frmProdInput.txtActualX.Text
1278  cmd = cmd + "," + ""
1279  frmProdInput.txtBaseX.Text
1280  cmd = cmd + ")"
1281  fconnection.Execute cmd
1282  frmProdInput.exist = True
1283  End If
1284  cmd = "insert into Frozen values ("
1285  cmd = cmd + "," + val_prod_no.Caption + ""
1286  cmd = cmd + "," + Label12.Caption + ""
1287  cmd = cmd + "," + ""
1288  cmd = cmd + Text13.Text
1289  cmd = cmd + ")"
1290  fconnection.Execute cmd
1291  Set rs = New Recordset
1292  cmd = "select * from Inventory where species ="
1293  = ""
1294  cmd = cmd + frmProd.sSpecies + ";" and "." + ""
1295  cmd = cmd + "ItemNo = ""
1296  cmd = cmd + lab_array(12).Caption + ""
1297  rs.Open cmd, fconnection,
1298  adOpenForwardOnly, adLockReadOnly
1299  rs.MoveFirst
1300  tmp2 = rs("Quantity")
1301  rs.Close
1302  Set rs = Nothing
1303  Text25.Text = 0
1305  cmd = "update Inventory set Quantity ="
1306  cmd = cmd + Text25.Text
1307  cmd = cmd + "," where Species ="
1308  cmd = cmd + frmProd.sSpecies + ";" and "." + ""
1309  cmd = cmd + "ItemNo = ""
1310  cmd = cmd + lab_array(12).Caption + ""
1311  fconnection.Execute cmd
1312  tmp = True
1313  End If
1314  If Text14.Text <> "" Then
1315  If frmProdInput.exist = False Then
1316  If frmProdInput.exist = False Then
1317  If frmProdInput.exist = False Then
cmd = "insert into Production values ("
frmProdInput.txtProduction.Text + ":"
frmProdInput.val_species.Caption + ""
frmProdInput.cboVessel.Text +"
frmProdInput.cboSource.Text +"
frmProdInput.cboArea.Text +"
frmProdInput.txtRate.Text
frmProdInput.txtDayX.Text
frmProdInput.txtActualX.Text
frmProdInput.txtBaseX.Text
frmProdInput.cboSource.Text +")"
connection.Execute cmd
frmProdInput.exist = True
End If

If Text15.Text <> "" Then
If frmProdInput.exist = False Then

cmd = "insert into Production values ("
cmd = cmd + ":" + frmProdInput.txtRate.Text
frmProdInput.txtDayX.Text
frmProdInput.txtActualX.Text
frmProdInput.txtBaseX.Text
cmd = cmd + ":" + Text14.Text
cmd = cmd + ")"
connection.Execute cmd
Set rs = New Recordset

cmd = "select * from Inventory where spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(13).Caption + ""
rs.Open cmd, connection,
rs.MoveFirst
tmp2 = rs("Quantity")
rs.Close
Set rs = Nothing
Text25.Text = 0

update Inventory set Quantity ="
cmd = cmd + Text25.Text
cmd = cmd + " where Spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(13).Caption + ""
connection.Execute cmd
tmp = True
End If

If Text15.Text <> "" Then
If frmProdInput.exist = False Then

cmd = "insert into Production values ("
cmd = cmd + ":" + frmProdInput.txtRate.Text
frmProdInput.txtDayX.Text
frmProdInput.txtActualX.Text
frmProdInput.txtBaseX.Text
cmd = cmd + ":" + Text14.Text
cmd = cmd + ")"
connection.Execute cmd
frmProdInput.exist = True
End If

connection.Execute cmd
Set rs = New Recordset

cmd = "select * from Inventory where spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(13).Caption + ""
rOpen cmd, connection,
rMoveFirst

tmp2 = rs("Quantity")
rClose
Set rs = Nothing
Text25.Text = 0

update Inventory set Quantity ="
cmd = cmd + Text25.Text
cmd = cmd + " where Spieces = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(13).Caption + ""
connection.Execute cmd
tmp = True
End If
cmd = "select * from Inventory where species = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(14).Caption + ""
rs.Open cmd, fconnection, adOpenForwardOnly, adLockReadOnly
rs.MoveFirst
tmp2 = rs("Quantity")
rs.Close
Set rs = Nothing
Text25.Text = 0
Text25.Text = Text15.Text + tmp2
cmd = "update Inventory set Quantity ="
cmd = cmd + Text25.Text
cmd = cmd + " where Species = "
cmd = cmd + frmProd.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(14).Caption + ""
fconnection.Execute cmd
tmp = True
End If
If Text16.Text <> "" Then
If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + ""
frmProdInput.txtProduction.Text + ""
cmd = cmd + ", " + "Date()"
cmd = cmd + ", " + "Date()"
cmd = cmd + ", "
cmd = cmd + ""
frmProdInput.cboVessel.Text + ""
cmd = cmd + ", "
cmd = cmd + ""
frmProdInput.cboSource.Text + ""
cmd = cmd + ", "
cmd = cmd + ""
frmProdInput.cboArea.Text + ""
cmd = cmd + ", " + frmProdInput.txtRate.Text
cmd = cmd + ""
frmProdInput.txtDayX.Text
cmd = cmd + ""
frmProdInput.txtActualX.Text
cmd = cmd + ""
frmProdInput.txtBaseX.Text
cmd = cmd + "")"
fconnection.Execute cmd
frmProdInput.exist = True
End If
cmd = "insert into Frozen values ("
cmd = cmd + ""
val_prod_no.Caption + ""
cmd = cmd + "", " + Label16.Caption + ""
cmd = cmd + ", "
cmd = cmd + Text16.Text
cmd = cmd + "")"
fconnection.Execute cmd
Set rs = New Recordset
cmd = "select * from Inventory where species ="
cmd = cmd + ""
cmd = cmd + frmProdInput.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(15).Caption + ""
rn.Open cmd, fconnection, adOpenForwardOnly, adLockReadOnly
rs.MoveFirst
tmp2 = rs("Quantity")
rs.Close
Set rs = Nothing
Text25.Text = 0
cmd = "update Inventory set Quantity ="
cmd = cmd + Text25.Text
cmd = cmd + " where Species = "
cmd = cmd + frmProdInput.sSpecies + " and "
cmd = cmd + "ItemNo = "
cmd = cmd + lab_array(15).Caption + ""
fconnection.Execute cmd
tmp = True
End If
If Text17.Text <> "" Then
If frmProdInput.exist = False Then
cmd = "insert into Production values ("
cmd = cmd + ""
frmProdInput.txtProduction.Text + ""
cmd = cmd + ", " + frmProdInput.txtRate.Text
cmd = cmd + ""
frmProdInput.txtDayX.Text
cmd = cmd + ""
frmProdInput.txtActualX.Text
cmd = cmd + ""
frmProdInput.txtBaseX.Text
cmd = cmd + ", " +
1492 frmProdInput.cboVessel.Text + ""
1493 cmd = cmd + ", " +
1494 frmProdInput.cboSource.Text + ""
1495 cmd = cmd + ", " +
1496 frmProdInput.cboArea.Text + ""
1497 cmd = cmd + ", " + frmProdInput.txtRate.Text
1498 cmd = cmd + ", " +
1499 frmProdInput.txtDayX.Text
1500 cmd = cmd + ", " +
1501 frmProdInput.txtActualX.Text
1502 cmd = cmd + ", " +
1503 frmProdInput.txtBaseX.Text
1504 cmd = cmd + ")"
1505 fconnection.Execute cmd
1506 frmProdInput.exist = True
1507 End If
1508 cmd = "insert into Frozen values ("
1509 cmd = cmd + "" + val_prod_no.Caption + ""
1510 cmd = cmd + ", " + Label17.Caption + ""
1511 cmd = cmd + ", " +
1512 cmd = cmd + Text17.Text
1513 cmd = cmd + ")"
1514 fconnection.Execute cmd
1515 Set rs = New Recordset
1516 cmd = "select * from Inventory where spieces = "
1517 = ""
1518 cmd = cmd + frmProd.sSpecies + "" and "
1519 cmd = cmd + "ItemNo = "
1520 cmd = cmd + lab_array(16).Caption + ""
1521 rs.Open cmd, fconnection,
1522 adOpenForwardOnly, adLockReadOnly
1523 rs.MoveNextFirst
1524 tmp2 = rs("Quantity")
1525 rs.Close
1526 Set rs = Nothing
1527 Text25.Text = 0
1528 Text25.Text = Text17.Text + tmp2
1529 cmd = "update Inventory set Quantity ="
1530 cmd = cmd + Text25.Text
1531 cmd = cmd + " where Spieces = "
1532 cmd = cmd + frmProd.sSpecies + "" and "
1533 cmd = cmd + "ItemNo = "
1534 cmd = cmd + lab_array(16).Caption + ""
1535 fconnection.Execute cmd
1536 tmp = True
1537 End If
1538
1539 If Text18.Text <> "" Then
1540 If frmProdInput.exist = False Then
1541 cmd = "insert into Production values ("
1542 cmd = cmd + ""
1543 frmProdInput.txtProduction.Text + ""
1544 cmd = cmd + ", " + "Date()"
1545 cmd = cmd + ", " +
1546 frmProdInput.val_species.Caption + ""
1547 cmd = cmd + ", " +
1548 frmProdInput.cboVessel.Text + ""
1549 cmd = cmd + ", " +
1550 frmProdInput.cboSource.Text + ""
1551 cmd = cmd + ", " +
1552 frmProdInput.cboArea.Text + ""
1553 cmd = cmd + ", " + frmProdInput.txtRate.Text
1554 cmd = cmd + ", " +
1555 frmProdInput.txtDayX.Text
1556 cmd = cmd + ", " +
1557 frmProdInput.txtActualX.Text
1558 cmd = cmd + ", " +
1559 frmProdInput.txtBaseX.Text
1560 cmd = cmd + ")"
1561 fconnection.Execute cmd
1562 frmProdInput.exist = True
1563 End If
1564 cmd = "insert into Frozen values ("
1565 cmd = cmd + "" + val_prod_no.Caption + ""
1566 cmd = cmd + ", " + Label18.Caption + ""
1567 cmd = cmd + ", " +
1568 cmd = cmd + Text18.Text
1569 cmd = cmd + ")"
1570 fconnection.Execute cmd
1571 Set rs = New Recordset
1572 cmd = "select * from Inventory where spieces = "
1573 = ""
1574 cmd = cmd + frmProd.sSpecies + "" and "
1575 cmd = cmd + "ItemNo = "
1576 cmd = cmd + lab_array(17).Caption + ""
1577 rs.Open cmd, fconnection,
1578 adOpenForwardOnly, adLockReadOnly
1579    rs.MoveFirst
1580    tmp2 = rs("Quantity")
1581    rs.Close
1582    Set rs = Nothing
1583    Text25.Text = 0
1584    Text25.Text = Text18.Text + tmp2
1585    cmd = "update Inventory set Quantity ="
1586    cmd = cmd + Text25.Text
1587    cmd = cmd + " where Spieces = "
1588    cmd = cmd + frmProd.sSpecies + " and "
1589    cmd = cmd + "ItemNo = "
1590    cmd = cmd + lab_array(17).Caption + ""
1591    fconnection.Execute cmd
1592    tmp = True
1593    End If
1594
1595    If Text19.Text <> "" Then
1596    If frmProdInput.exist = False Then
1597    cmd = "insert into Production values ("
1598    cmd = cmd + ""
1599    frmProdInput.txtProduction.Text + ""
1600    cmd = cmd + ", " + "Date()"
1601    cmd = cmd + ", " + ""
1602    frmProdInput.val_species.Caption + ""
1603    cmd = cmd + ", " + ""
1604    frmProdInput.cboVessel.Text + ""
1605    cmd = cmd + "", " + ""
1606    frmProdInput.cboSource.Text + ""
1607    cmd = cmd + "", " + ""
1608    frmProdInput.cboArea.Text + ""
1609    cmd = cmd + "", " + frmProdInput.txtRate.Text
1610    cmd = cmd + "", " + ""
1611    frmProdInput.txtDayX.Text
1612    cmd = cmd + ", " + ""
1613    frmProdInput.txtActualX.Text
1614    cmd = cmd + ", " + ""
1615    frmProdInput.txtBaseX.Text
1616    cmd = cmd + ")"
1617    fconnection.Execute cmd
1618    frmProdInput.exist = True
1619    End If
1620    cmd = "insert into Frozen values ("
1621    cmd = cmd + ""
1622    cmd = cmd + "val_prod_no.Caption + ""
1623    cmd = cmd + ", " + Label19.Caption + ""
1624    cmd = cmd + Text19.Text
1625    cmd = cmd + "")"
1626    fconnection.Execute cmd
1627    Set rs = New Recordset
1628    cmd = "select * from Inventory where Spieces ="
1629    cmd = cmd + frmProd.sSpecies + " and "
1630    cmd = cmd + "ItemNo = "
1631    cmd = cmd + lab_array(18).Caption + ""
1632    rs.Open cmd, fconnection, adOpenForwardOnly, adLockReadOnly
1633    rs.MoveNext
1634    tmp2 = rs("Quantity")
1635    rs.Close
1636    Set rs = Nothing
1637    Text25.Text = 0
1639    cmd = "update Inventory set Quantity ="
1640    cmd = cmd + Text25.Text
1641    cmd = cmd + " where Spieces = "
1642    cmd = cmd + frmProd.sSpecies + " and "
1643    cmd = cmd + "ItemNo = "
1644    cmd = cmd + lab_array(18).Caption + ""
1645    fconnection.Execute cmd
1646    tmp = True
1647    End If
1648
1649    If Text20.Text <> "" Then
1650    If frmProdInput.exist = False Then
1651    cmd = "insert into Production values ("
1652    cmd = cmd + ""
1653    frmProdInput.txtProduction.Text + ""
1654    cmd = cmd + ", " + frmProdInput.txtRate.Text
1655    cmd = cmd + ", " + "Date()"
1656    cmd = cmd + "", " + ""
1657    cmd = cmd + ", " + ""
1658    frmProdInput.val_species.Caption + ""
1659    cmd = cmd + "", " + ""
1660    frmProdInput.cboVessel.Text + ""
1661    cmd = cmd + "", " + ""
1662    frmProdInput.cboSource.Text + ""
1663    cmd = cmd + "", " + ""
1664    frmProdInput.cboArea.Text + ""
1665    cmd = cmd + ", " + frmProdInput.txtRate.Text
cmd = cmd + "," +
1667  frmProdInput.txtDayX.Text
1668  cmd = cmd + "," +
1669  frmProdInput.txtActualX.Text
1670  cmd = cmd + "," +
1671  frmProdInput.txtBaseX.Text
1672  cmd = cmd + ")"
1673  fconnection.Execute cmd
1674  frmProdInput.exist = True
1675  End If
1676  cmd = "insert into Frozen values ("
1677  cmd = cmd + "" + val_prod_no.Caption + ""
1678  cmd = cmd + "," + Label20.Caption + ""
1679  cmd = cmd + ","
1680  cmd = cmd + Text20.Text
1681  cmd = cmd + ")"
1682  fconnection.Execute cmd
1683  Set rs = New Recordset
1684  cmd = "select * from Inventory where spieces ="
1685  ""
1686  cmd = cmd + frmProd.sSpecies + " and "
1687  cmd = cmd + "ItemNo = "
1688  cmd = cmd + lab_array(19).Caption + ""
1689  rs.Open cmd, fconnection,
1690  adOpenForwardOnly, adLockReadOnly
1691  rs.MoveFirst
1692  tmp2 = rs("Quantity")
1693  rs.Close
1694  Set rs = Nothing
1695  Text25.Text = 0
1696  Text25.Text = Text20.Text + tmp2
1697  cmd = "update Inventory set Quantity ="
1698  cmd = cmd + Text25.Text
1699  cmd = cmd + "," where Spieces = ""
1700  cmd = cmd + frmProd.sSpecies + " and "
1701  cmd = cmd + "ItemNo = "
1702  cmd = cmd + lab_array(19).Caption + ""
1703  fconnection.Execute cmd
1704  tmp = True
1705  End If
1706  
1707  If Text21.Text <> "" Then
1708  If frmProdInput.exist = False Then
1709  cmd = "insert into Production values ("
1710  cmd = cmd + ""
1711  frmProdInput.txtProdution.Text + ""
1712  cmd = cmd + "," + "Date()"
1713  cmd = cmd + "," +
1714  frmProdInput.val_species.Caption + ""
1715  cmd = cmd + "," +
1716  frmProdInput.cboVessel.Text + ""
1717  cmd = cmd + "," +
1718  frmProdInput.cboSource.Text + ""
1719  cmd = cmd + "," +
1720  frmProdInput.cboArea.Text + ""
1721  cmd = cmd + "," + frmProdInput.txtRate.Text
1722  cmd = cmd + "," +
1723  frmProdInput.txtDayX.Text
1724  cmd = cmd + "," +
1725  frmProdInput.txtActualX.Text
1726  cmd = cmd + "," +
1727  frmProdInput.txtBaseX.Text
1728  cmd = cmd + ")"
1729  fconnection.Execute cmd
1730  frmProdInput.exist = True
1731  End If
1732  cmd = "insert into Frozen values ("
1733  cmd = cmd + "" + val_prod_no.Caption + ""
1734  cmd = cmd + "," + Label21.Caption + ""
1735  cmd = cmd + ","
1736  cmd = cmd + Text21.Text
1737  cmd = cmd + ")"
1738  fconnection.Execute cmd
1739  Set rs = New Recordset
1740  cmd = "select * from Inventory where spieces ="
1741  ""
1742  cmd = cmd + frmProd.sSpecies + " and "
1743  cmd = cmd + "ItemNo = "
1744  cmd = cmd + lab_array(20).Caption + ""
1745  rs.Open cmd, fconnection,
1746  adOpenForwardOnly, adLockReadOnly
1747  rs.MoveFirst
1748  tmp2 = rs("Quantity")
1749  rs.Close
1750  Set rs = Nothing
1751  Text25.Text = 0
1753  cmd = "update Inventory set Quantity ="
cmd = cmd + Text22.Text
1755 cmd = cmd + " where Species = "
1756 cmd = cmd + frmProd.sSpecies + " and "
1757 cmd = cmd + "ItemNo = "
1758 cmd = cmd + lab_array(20).Caption + ""
1759 fconnection.Execute cmd
1760 tmp = True
1761 End If
1762
1763 If Text22.Text <> "" Then
1764 If frmProdInput.exist = False Then
1765 cmd = "insert into Production values (" 
1766 cmd = cmd + ""
1767 frmProdInput.txtProduction.Text + ""
1768 cmd = cmd + ", ", " + "Date()
1769 cmd = cmd + ", ", " + 
1770 frmProdInput.val_species.Caption + ""
1771 cmd = cmd + ", ", " + 
1772 frmProdInput.cboVessel.Text + ""
1773 cmd = cmd + ", ", " + 
1774 frmProdInput.cboSource.Text + ""
1775 cmd = cmd + ", ", " + 
1776 frmProdInput.cboArea.Text + ""
1777 cmd = cmd + ", ", " + frmProdInput.txtRate.Text
1778 cmd = cmd + ", ", " + 
1779 frmProdInput.txtDayX.Text
1780 cmd = cmd + ", ", " + 
1781 frmProdInput.txtActualX.Text
1782 cmd = cmd + ", ", " + 
1783 frmProdInput.txtBaseX.Text
1784 cmd = cmd + ")"
1785 fconnection.Execute cmd
1786 frmProdInput.exist = True
1787 End If
1788 cmd = "insert into Frozen values ("
1789 cmd = cmd + ""
1790 cmd = cmd + " val_prod_no.Caption + ""
1791 cmd = cmd + ", ", " + Label22.Caption + ""
1792 cmd = cmd + ", ", " + 
1793 cmd = cmd + ")"
1794 fconnection.Execute cmd
1795 Set rs = New Recordset
1796 cmd = "select * from Inventory where species = "
1797 = ""
1798 cmd = cmd + frmProd.sSpecies + "" and "
1799 cmd = cmd + "ItemNo = "
1800 cmd = cmd + lab_array(21).Caption + ""
1801 rs.Open cmd, fconnection,
1802 adOpenForwardOnly, adLockReadOnly
1803 rs.MoveFirst
1804 tmp2 = rs("Quantity")
1805 rs.Close
1806 Set rs = Nothing
1807 Text25.Text = 0
1808 Text25.Text = Text22.Text + tmp2
1809 cmd = "update Inventory set Quantity ="
1810 cmd = cmd + Text25.Text
1811 cmd = cmd + " where Species = "
1812 cmd = cmd + frmProd.sSpecies + "" and "
1813 cmd = cmd + "ItemNo = "
1814 cmd = cmd + lab_array(21).Caption + ""
1815 fconnection.Execute cmd
1816 tmp = True
1817 End If
1818
1819 If Text23.Text <> "" Then
1820 If frmProdInput.exist = False Then
1821 cmd = "insert into Production values ("
1822 cmd = cmd + ""
1823 frmProdInput.txtProduction.Text + ""
1824 cmd = cmd + ", ", " + "Date()"
1825 cmd = cmd + ", ", " + 
1826 frmProdInput.val_species.Caption + ""
1827 cmd = cmd + ", ", " + 
1828 frmProdInput.cboVessel.Text + ""
1829 cmd = cmd + ", ", " + 
1830 frmProdInput.cboSource.Text + ""
1831 cmd = cmd + ", ", " + 
1832 frmProdInput.cboArea.Text + ""
1833 cmd = cmd + ", ", " + frmProdInput.txtRate.Text
1834 cmd = cmd + ", ", " + 
1835 frmProdInput.txtDayX.Text
1836 cmd = cmd + ", ", " + 
1837 frmProdInput.txtActualX.Text
1838 cmd = cmd + ", ", " + 
1839 frmProdInput.txtBaseX.Text
1840 cmd = cmd + ")"
1841 fconnection.Execute cmd
frmProdInput.exist = True
End If

cmd = "insert into Frozen values ("

cmd = cmd + " + val_prod_no.Caption + ")"

cmd = cmd + " + Label23.Caption + ""

cmd = cmd + ", "

cmd = cmd + "Text23.Text"

cmd = cmd + "")"

fconnection.Execute cmd

Set rs = New Recordset

cmd = "select * from Inventory where spieces = "

cmd = cmd + dbo_species + " and "

cmd = cmd + "ItemNo = "

cmd = cmd + " + lab_array(22).Caption + ""

rs.Open cmd, fconnection,
adOpenForwardOnly, adLockReadOnly

rs.MoveFirst
tmp2 = rs("Quantity")

rs.Close

Set rs = Nothing

Text25.Text = 0

Text25.Text = Text23.Text + tmp2

cmd = "update Inventory set Quantity ="

cmd = cmd + Text25.Text

cmd = cmd + " where Spieces = "

cmd = cmd + dbo_species + " and "

cmd = cmd + "ItemNo = "

cmd = cmd + " + lab_array(22).Caption + ""

fconnection.Execute cmd
tmp = True

End If

If Text24.Text <> "" Then

If frmProdInput.exist = False Then

cmd = "insert into Production values ("

cmd = cmd + ""

frmProdInput.txtProduction.Text + ""

cmd = cmd + ", " + "Date()"

cmd = cmd + ", "

frmProdInput.val_species.Caption + ""

cmd = cmd + ", " + " + "

frmProdInput.cboVessel.Text + ""

frmProdInput.editText + ""

frmProdInput.cbeSource.Text + ""

cmd = cmd + ", "

frmProdInput.cbeArea.Text + ""

cmd = cmd + ", " + frmProdInput.txtRate.Text

frmProdInput.txtDayX.Text

cmd = cmd + ", "

frmProdInput.txtActualX.Text

cmd = cmd + ", "

frmProdInput.txtBaseX.Text

cmd = cmd + ")"

fconnection.Execute cmd

frmProdInput.exist = True

End If

cmd = "insert into Frozen values ("

frmProdInput.editText + ""

cmd = cmd + ", "

frmProdInput.cbeSource.Text + ""

cmd = cmd + ", "

frmProdInput.cbeArea.Text + ""

cmd = cmd + ", " + frmProdInput.txtRate.Text

frmProdInput.txtDayX.Text

cmd = cmd + ", "

frmProdInput.txtActualX.Text

cmd = cmd + ", "

frmProdInput.txtBaseX.Text

cmd = cmd + ")"

fconnection.Execute cmd

Set rs = New Recordset

cmd = "select * from Inventory where spieces = "

frmProdInput.exist = False

If frmProdInput.exist = False Then

cmd = cmd + dbo_species + " and "

frmProdInput.editText + ""

cmd = cmd + ", ItemNo = "

frmProdInput.editText + ""

cmd = cmd + " + lab_array(22).Caption + ""

rs.Open cmd, fconnection,
adOpenForwardOnly, adLockReadOnly

rs.MoveFirst
tmp2 = rs("Quantity")

rs.Close

Set rs = Nothing

Text25.Text = 0

Text25.Text = Text24.Text + tmp2

cmd = "update Inventory set Quantity ="

frmProdInput.editText + ""

cmd = cmd + Text25.Text

cmd = cmd + " where Spieces = "

frmProdInput.editText + ""

cmd = cmd + dbo_species + " and "

frmProdInput.editText + ""

cmd = cmd + " + "

frmProdInput.editText + ""

cmd = cmd + '"'
1929 End If
1930
1931 If tmp = True Then
1932 frmProdInput.exist = True
1933 frmProdInput.txtProduction.Enabled = False
1934 frmProdInput.cboArea.Enabled = False
1935 frmProdInput.cboSource.Enabled = False
1936 frmProdInput.cboVessel.Enabled = False
1937 frmProdInput.txtActualX.Enabled = False
1938 frmProdInput.txtBaseX.Enabled = False
1939 frmProdInput.txtDayX.Enabled = False
1940 frmProdInput.cmdFrozen.Enabled = False
1941 frmProdInput.txtRate.Enabled = False
1942 End If
1943
1944 Unload Me
1945 End Sub
1946
1947 Private Sub Form_Load()
1948 'Dim lab_array(24) As Label
1949 Dim cmd As String
1950 Dim fcon As Connection
1951 Dim tmp As Boolean
1952 Dim i As Integer
1953 tmp = False
1954 Set fcon = New Connection
1955 fcon.Open
1956 "Provider=Microsoft.Jet.OLEDB.4.0;Data
1957 Source=ADMS.mdb;Persist Security
1958 Info=False"
1959 cmd = "select * from Inventory where species
1960 = ""
1961 cmd = cmd + frmProdSpecs + ","
1962 Set rs = New Recordset
1963 rs.Open cmd, fcon, , adOpenForwardOnly, adLockReadOnly
1964 rs.MoveFirst
1965
1966 val_prod_no.Caption =
1967 frmProdInput.txtProduction.Text
1968 val_species.Caption = frmProdSpecs
1969 cmdSubmit.Enabled = False
1970 'Label1.Caption = rs("ItemNo")
1971 'rs.MoveNext
1972 'Label2.Caption = rs("ItemNo")
1973 'rs.MoveNext
1974 'Label3.Caption = rs("ItemNo")
1975 'rs.MoveNext
1976 'Label4.Caption = rs("ItemNo")
1977 'rs.MoveNext
1978 'Label5.Caption = rs("ItemNo")
1979 'rs.MoveNext
1980 'Label6.Caption = rs("ItemNo")
1981 'rs.MoveNext
1982 'Label7.Caption = rs("ItemNo")
1983 'rs.MoveNext
1984 'Label8.Caption = rs("ItemNo")
1985 'rs.MoveNext
1986 'Label9.Caption = rs("ItemNo")
1987 'rs.MoveNext
1988 'Label10.Caption = rs("ItemNo")
1989 'rs.MoveNext
1990 'Label11.Caption = rs("ItemNo")
1991 'rs.MoveNext
1992 'Label12.Caption = rs("ItemNo")
1993 'rs.MoveNext
1994 'Label13.Caption = rs("ItemNo")
1995 'rs.MoveNext
1996 'Label14.Caption = rs("ItemNo")
1997 'rs.MoveNext
1998 'Label15.Caption = rs("ItemNo")
1999 'rs.MoveNext
2000 'Label16.Caption = rs("ItemNo")
2001 'rs.MoveNext
2002 'Label17.Caption = rs("ItemNo")
2003 'rs.MoveNext
2004 'Label18.Caption = rs("ItemNo")
2005 'rs.MoveNext
2006 'Label19.Caption = rs("ItemNo")
2007 'rs.MoveNext
2008 'Label20.Caption = rs("ItemNo")
2009 'rs.MoveNext
2010 'Label21.Caption = rs("ItemNo")
2011 'rs.MoveNext
2012 'Label22.Caption = rs("ItemNo")
2013 'rs.MoveNext
2014 'Label23.Caption = rs("ItemNo")
2015 'rs.MoveNext
2016 'rs.MoveNext
Private Sub Text13_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text14_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text15_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text16_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text17_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text18_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text19_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text20_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text21_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text22_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub Text23_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub cmdSubmit_Change()
    cmdSubmit.Enabled = True
End Sub

Private Sub cmdSubmit_Click()
    Dim cmd As String
    Dim fconnection As Connection
    'Dim piconnection As Connection
    Dim tmp As Boolean
    tmp = False
    Set fconnection = New Connection
    fconnection.Open
    "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=ADMS.mdb;Persist Security Info=False"
    If Text1.Text <> "" Then
        If frmProdInput.exi = False Then
            cmd = "insert into Production values ("
            cmd = cmd + "" + frmProdInput.txtProduction.Text
            cmd = cmd + ""
            cmd = cmd + ", " + frmProdInput.val_species.Caption + ""
            cmd = cmd + ", " + frmProdInput.cboVessel.Text
            cmd = cmd + ""
            cmd = cmd + ", " + frmProdInput.cboSource.Text
            cmd = cmd + ""
            cmd = cmd + ", " + frmProdInput.cboArea.Text
            cmd = cmd + ""
            cmd = cmd + ", " + frmProdInput.txtRate.Text
            cmd = cmd + ""
            cmd = cmd + ", " + frmProdInput.txtDayX.Text
2192 cmd = cmd + "," +
2193 frmProdInput.txtActualX.Text
2194 cmd = cmd + "," +
2195 frmProdInput.txtBaseX.Text
2196 cmd = cmd + ")"
2197 fconnection.Execute cmd
2198 frmProdInput.exist = True
2199 End If
2200 cmd = "insert into Slab values ("
2201 cmd = cmd + "" + val_prod_no.Caption + ""
2202 cmd = cmd + "," + Text1.Text
2203 cmd = cmd + "," +
2204 cmd = cmd + Text2.Text
2205 cmd = cmd + ")"
2206 fconnection.Execute cmd
2207 tmp = True
2208 End If
2209
2210 If tmp = True Then
2211 frmProdInput.exist = True
2212 frmProdInput.txtProduction.Enabled = False
2213 frmProdInput.cboArea.Enabled = False
2214 frmProdInput.cboSource.Enabled = False
2215 frmProdInput.cboVessel.Enabled = False
2216 frmProdInput.txtActualX.Enabled = False
2217 frmProdInput.txtBaseX.Enabled = False
2218 frmProdInput.txtDayX.Enabled = False
2219 frmProdInput.cmdSlabs.Enabled = False
2220 frmProdInput.txtRate.Enabled = False
2221 End If
2222 Unload Me
2223 End Sub
2224
2225 Private Sub Form_Load()
2226 val_prod_no.Caption =
2227 frmProdInput.txtProduction.Text
2228 val_species.Caption = frmProd.sSpecies
2229 cmdSubmit.Enabled = False
2230 End Sub
2231
2232 Private Sub Text1_Change()
2233 cmdSubmit.Enabled = True
2234 End Sub
2235
2236 Private Sub Text2_Change()
2237 cmdSubmit.Enabled = True
2238 End Sub
2239
2240 'Code for frame of Order
2241 Option Explicit
2242 Public customerName As String
2243
2244 Private Sub cmdExit_Click()
2245 Unload Me
2246 End Sub
2247
2248 Private Sub cmdNewCustomer_Click()
2249 Unload Me
2250 frmNewCustomer.Show "vbModal", Me
2251 End Sub
2252
2253 Private Sub cmdProcess_Click()
2254 'Dim iOrderTemp As Variant
2255 iOrderTemp = Shell("C:\Program
2256 Files\Microsoft Office\Office\EXCEL ",&
2257 "D:\EE\Sys\Thesis\ADMS\InvProTemplate.x")
2258
2259 customerName = DataComboBox.Text
2260 ActivateInvoice
2261 End Sub
2262
2263 Public appADMS As Excel.Application
2264 Public wbADMS As Excel.Workbook'
2265
2266 Sub ActivateInvoice()
2267 Dim shtInvoice As Excel.Worksheet
2268 Dim cn As ADODB.Connection
2269 Dim rs As ADODB.Recordset
2270 Dim cmd As String
2271 Dim i As Integer
2272
2273 On Error Resume Next
2274 Set appADMS = GetObject(
2275 "Excel.Application")
2276 If Err.Number <> 0 Then
2277 Set appADMS =
2278 CreateObject("Excel.Application")
End If
Err.Clear
On Error GoTo 0
Set wbADMS = appADMS.Workbooks.Open(App.Path & "\InvProTemplate.xlt")
wbADMS.Application.Visible = True
wbADMS.Window(1).Visible = True
Set shtInvoice = wbADMS.Sheets(1)
shtInvoice.Cells(6, 3).Value =
frmOrder.DataCombo1.Text
Set cn = New Connection
cn.Open "Provider = Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and Settings\Guowei\Desktop\ADMS\ADMS.mdb"
Persist Security Info=False"
Set rs = New Recordset
cmd = "select * from Inventory order by index"
rs.Open cmd, cn, adOpenForwardOnly, adLockReadOnly
rs.MoveFirst
i = 17
Do While Not rs.EOF
shtInvoice.Cells(i, 9).Value = rs("Quantity")
i = i + 1
rs.MoveNext
Loop
rs.Close
Set rs = Nothing
End Sub
End Sub
Private Sub cmdSummit_Click()
Adodc1.Recordset.MoveNext
Adodc1.Refresh
Unload Me
frmOrder.Show 'vbModal, Me
End Sub
Sub ExcelToAccess()'
'ExcelToAccess Macro
'Macro recorded 1/12/2002 by Guowei'
'Keyboard Shortcut: Ctrl+a
Dim r As Long
Dim cn As ADODB.Connection
Dim rsInvoice As ADODB.Recordset
Dim rsInvCom As ADODB.Recordset
Dim rsInventory As ADODB.Recordset
Dim rs As ADODB.Recordset
Dim tmpInvoice As Integer
Dim tmpInventory As Integer
Dim tmp As String
Dim temItemNo As String
Dim sqlCmd As String
'connect to the Access database
Set cn = New ADODB.Connection
cn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and Settings\Guowei\Desktop\ADMS\ADMS.mdb"
Persist Security Info=False"
Dim rsInvoice = New ADODB.Recordset
rsInvoice.Open "Invoice", cn, adOpenKeyset, adLockOptimistic, adCmdTable ' all records in a table
Set rsInvCom = New ADODB.Recordset

rsInvCom.Open "Inv_Com", cn, adOpenKeyset,
adLockOptimistic, adCmdTable, ' all records in a
table
Set rsInventory = New ADODB.Recordset
rsInventory.Open "Inventory", cn,
adOpenKeyset, adLockOptimistic,
adCmdTable, ' all records in a table
rsInvCom.AddNew
rsInvCom.Fields("Invoice_No") =
Range("L" & 6).Value
rsInvCom.Fields("Date") = Range("L" &
7).Value
rsInvCom.Fields("Customer") = Range("C"
& 6).Value
rsInvCom.Fields("Ship") = Range("C" &
7).Value
rsInvCom.Update
r = 17 ' the start row in the worksheet
Do While Len(Range("C" & r).Formula) >
0 ' repeat until first empty cell in column A
With rsInvoice
If Range("H" & r).Value <> "" Then
tmpInvoice = Range("H" & r).Value
tmplItemNo = Range("G" & r).Value
' add values to each field in the
record
.AddNew ' create a new record
.Fields("Invoice_No") = Range("L" &
6).Value
.Fields("Item_No") = Range("G" &
r).Value
.Fields("Quantity") = Range("H" &
r).Value
.Fields("Unit_Price") = Range("J" &
r).Value
.Update ' stores the new record
sqlCmd = "select * from Inventory where
ItemNo = "
sqlCmd = sqlCmd + tmpItemNo +
Set rs = New ADODB.Recordset
rs.Open sqlCmd, cn,
adOpenForwardOnly, adLockReadOnly
rs.MoveFirst
tmpInventory = rs("Quantity")
rs.Close
Set rs = Nothing
tmp = tmpInventory - tmplInvoice
sqlCmd = "update Inventory set
Quantity ="
sqlCmd = sqlCmd + tmp
sqlCmd = sqlCmd + " where ItemNo
= "
sqlCmd = sqlCmd + tmpItemNo +
cn.Execute sqlCmd
End If
End With
r = r + 1 ' next row
Loop
rsInvoice.Close
Set rsInvoice = Nothing
rsInvCom.Close
Set rsInvCom = Nothing
rsInventory.Close
Set rsInventory = Nothing
cn.Close
Set cn = Nothing
End Sub
Code for the frame of Invoice
Option Explicit
Public tmpNo As String
Public cn As String
Public rs As Recordset
Public appADMS As Excel.Application
Public wbADMS As Excel.Workbook
Dim tmp As String
Private Sub cmbInvoiceNo_Click()
Dim cmd As String
tmpNo = cmbInvoiceNo.Text
If tmpNo <> "" Then
cmdPreview.Enabled = True
Set rs = New Recordset
cmd = "select * from Inv_Corn where Invoice_No = "
cmd = cmd + tmpNo
cmd = cmd + ""
rs.Open cmd, frmDescription.cn,
adOpenForwardOnly, adLockReadOnly
txtDate.Text = rs("Date")
txtCustomer.Text = rs("Customer")
txtShip.Text = rs("Ship")
rs.Close
Set rs = Nothing
End If
End Sub
Private Sub cmdExit_Click()
Unload Me
End Sub
Private Sub cmdPreview_Click()
Dim rs_Invoice As Recordset
Dim rs_Inventory As Recordset
Dim cmd As String
Dim i As Integer, j As Integer, rowNo As Integer
Dim cellValue As Variant
Dim shtInvoice As Excel.Worksheet
On Error Resume Next
Set appADMS = GetObject("Excel.Application")
"Excel.Application"
If Err.Number <> 0 Then
Set appADMS =
CreateObject("Excel.Application")
End If
Err.Clear
On Error GoTo 0
Set wbADMS =
appADMS.Workbooks.Open(App.Path & "InvoiceTemplate.xlt")
wADMS.Application.Visible = True
Set shtInvoice = wbADMS.Sheets(1)
shtInvoice.Cells(5, 11) =
cmbInvoiceNo.Text
shtInvoice.Cells(5, 3) = txtCustomer.Text
shtInvoice.Cells(6, 3) = txtShip.Text
shtInvoice.Cells(6, 11) = txtDate.Text
shtInvoice.Cells(5, 11) = cmbInvoiceNo.Text
cmd = cmd + ""
frmInvoice.cmbInvoiceNo.Text
cmd = cmd + ""
Set rs_Invoice = New Recordset
rs_Invoice.Open cmd, frmDescription.cn,
adOpenForwardOnly, adLockReadOnly
rs_Invoice.MoveNext
i = 16
'rowNo = rs_Invoice.RecordCount
'shtInvoice.Cells(3, 2) = rowNo
Set rs_Inventory = New Recordset
Do Until rs_Inventory.EOF
tmp = rs_Invoice("item_no")
cmd = "select * from Inventory where ItemNo = "
cmd = cmd + tmp
cmd = cmd + ""
rs_Inventory.Open cmd,
frmDescription.cn, adOpenForwardOnly,
adLockReadOnly
rs_Inventory.MoveNext
shtInvoice.Cells(i, 2) =
rs_Inventory("Spices")
shtInvoice.Cells(i, 3) =
rs_Inventory("Size")
shtInvoice.Cells(i, 4) =
rs_Inventory("SO/B")
shtInvoice.Cells(i, 5) =
rs_Inventory("Package")
shtInvoice.Cells(i, 6) =
srs_Invoice("Item_No")
shtInvoice.Cells(i, 7) =
srs_Invoice("Quantity")
shtInvoice.Cells(i, 9) =
srs_Invoice("Unit_Price")
    i = i + 1
rs_Inventory.Close
rs_Invoice.MoveNext
Loop
rowNo = i + 2
j = rowNo
For j = rowNo To 48 Step 1
    shtInvoice.Rows(rowNo).Delete
Next j
    rowNo = rowNo + 1
Set rs_Inventory = Nothing
rs_Invoice.Close
Set rs_Inventory = Nothing
End Sub
Private Sub cmbProdNo_Click()
    Dim cmd As String
    tmp_prod_no = cmbProdNo.Text
    If cmd <> "" Then
        cmdPreview.Enabled = True
        Set rs = New Recordset
        cmd = "select * from production where prod_no = ""
        cmd = cmd + tmp_prod_no
        cmd = cmd + ""
        rs.Open cmd, frmDescription.cn, adOpenForwardOnly, adLockReadOnly
        txtArea.Text = rs("Area")
        txtDate.Text = rs("Date")
        txtSpecies.Text = rs("Species")
        txtVessel.Text = rs("Vessel")
        txtSource.Text = rs("Source")
        End If
    End Sub
    Set rs = Nothing
Dim rowNo As Integer
Dim cellValue As Variant
Dim temp As String

cmd = "select * from species_info where species = "

'... Rest of the code...'

shtInvoice.Cells(15, 2).Value = cellValue
shtInvoice.Cells(9, 3).Value = cellValue
shtInvoice.Cells(10, 3).Value = cellValue
shtInvoice.Cells(10, 4).Value = cellValue
shtInvoice.Cells(70, 3).Value = cellValue
shtInvoice.Cells(71, 3).Value = cellValue
shtInvoice.Cells(72, 3).Value = cellValue
shtInvoice.Cells(74, 3).Value = cellValue
shtInvoice.Cells(75, 3).Value = cellValue
shtInvoice.Cells(77, 3).Value = cellValue
shtInvoice.Cells(78, 3).Value = cellValue
shtInvoice.Cells(80, 3).Value = cellValue
shtInvoice.Cells(70, 7).Value = cellValue
shtInvoice.Cells(71, 7).Value = cellValue
shtInvoice.Cells(70, 7).Value = cellValue
shtInvoice.Cells(71, 7).Value = cellValue
shtInvoice.Cells(70, 7).Value = cellValue
shtInvoice.Cells(71, 7).Value = cellValue

rs_frozen.MoveFirst
Set rs_Inventory = New Recordset
Do Until rs_frozen.EOF
tmp = rs_frozen("item_no")
cmd = "select * from inventory where ItemNo"
cmd = cmd + tmp
cmd = cmd + ""
rs_Inventory.Open cmd, frmDescription.cn,
adOpenForwardOnly, adLockReadOnly
rs_Inventory.MoveFirst

    cellValue = rs("species")
    shtInvoice.Cells(i, 1).Value = cellValue
    cellValue = rs_Inventory("size")
    shtInvoice.Cells(i, 2).Value = cellValue
    cellValue = rs_Inventory("SO/B")
    shtInvoice.Cells(i, 3).Value = cellValue
    cellValue = rs_Inventory("Package")
    shtInvoice.Cells(i, 4).Value = cellValue
    cellValue = tmp
    shtInvoice.Cells(i, 5).Value = cellValue
    cellValue = rs_frozen("weight")
    shtInvoice.Cells(i, 6).Value = cellValue
    cellValue = rs("us_return")
    shtInvoice.Cells(i, 8).Value = cellValue
    i = i + 1
    rs_Inventory.Close
    rs_frozen.MoveNext
Loop

    cmd = "select * from fresh where prod_no"
cmd = cmd + frmWS.cmbProdNo.Text
cmd = cmd + ""
i = 20
Set rs_fresh = New Recordset
rs_fresh.Open cmd, frmDescription.cn,
adOpenForwardOnly, adLockReadOnly
rs_fresh.MoveFirst
Do Until rs_fresh.EOF
cellValue = rs_fresh("size")
shtInvoice.Cells(i, 1).Value = cellValue
    cellValue = rs_fresh("weight")
shtInvoice.Cells(i, 2).Value = cellValue
    cellValue = rs("us_return")
shtInvoice.Cells(i, 3).Value = cellValue
    i = i + 1
    rs_fresh.MoveNext
    Loop
End Sub

Private Sub Form_Load()
    'Set cn = New Connection
    'cn.Open "Provider =
    Microsoft.Jet.OLEDB.4.0;Data
    Source=C:\Documents and
    Settings\Guwei\Desktop\ADMS\ADMS.mdb
    :Persist Security Info=False"
    cmdPrevieEnabled = False
Set rs = New Recordset
rs.Open "production", frmDescription.cn,
adOpenForwardOnly, adLockReadOnly
rs.MoveFirst
Do Until rs.EOF
    frmWS.cmbProdNo.AddItem
    (rs("prod_no"))
    rs.MoveNext
    Loop
rs.Close
Set rs = Nothing
End Sub

Sub SavetoDB()
    ' SavetoDB Macro
    ' Macro recorded 2/27/2002 by Guwei'
    ' Keyboard Shortcut: Ctrl+s'
    Dim cn As ADODB.Connection
    Dim rsIncomeVar As ADODB.Recordset
    Dim tmp As Double
    'Dim sqlCmd As String
    ' connect to the Access database
    Set cn = New ADODB.Connection

End Sub

Set rsIncomeVar = Nothing

Option Explicit

Public sSpecies As String

Public cn As Connection

Public rs As Recordset

Public appADMS As Excel.Application

Public wbADMS As Excel.Workbook

Private Sub cmdPreview_Click()
    Dim cmd As String
    Dim i As Integer, j As Integer
    Dim rowNo As Integer

    'Dim cellRange As String
    Dim cellValue As Variant
    'Dim cellPlace As String
    'Dim cellName As String
    'Dim totalRows As Integer
    'Dim nameExcel As String
    'Dim temp As String
    Dim shInvoice As Excel.Worksheet
    Dim ssName As String
    On Error Resume Next

    Set appADMS = GetObject(
        "Excel.Application"
    )
    If Err.Number <> 0 Then
        Set appADMS = =
        CreateObject("Excel.Application")
    End If
    Err.Clear
    On Error GoTo 0

Set wbADMS = appADMS.Workbooks.Open(App.Path & "inventory.xlt")
wbADMS.Application.Visible = True
'wbADMS.Window(1).Visible = True
Set shtInvoice = wbADMS.Sheets(1)
shtInvoice.Cells(6, 3) = frmOrder.DataTable1.Text
'Dim saveCursor = Me.MousePointer
'Me.MousePointer = vbHourglass
cmd = "select * from inventory where Spieces = "
cmd = cmd + 
'cmd = cmd + sSpecies
'cmd = cmd + 
Set rs = New Recordset
rs.Open cmd, frmDescription.cn, adOpenForwardOnly, adLockReadOnly
'oleExcel.CreateEmbed "", "Excel.Sheet.8"
On Error GoTo 0
rowNo = 7
i = 5
'ssName = oleExcel.object.Name
'oleExcel.object.Worksheets(1).Row = 6
'oleExcel.object.Worksheets(1).Col = 24
Do While Not rs.EOF
'rowNo = rowNo + 1
For j = 0 To i
   cellValue = rs(j)
Next
'oleExcel.object.Worksheets(1).Cells(rowNo, j + 1).Value = cellValue
shtInvoice.Cells(rowNo, j + 3).Value = cellValue
Next
rowNo = rowNo + 1
rs.MoveNext
Loop
Set appADMS.Worksheets = oleExcel.object.Worksheets(1)
appADMS.Visible = True
rs.Close
Set rs = Nothing
'Exit Sub
'OLEError:
'MsgBox "An OLE error occurred, probably because Excel is not installed on this computer."
'Unload Me
End Sub
Private Sub cmdExit_Click()
'cn.Close
Unload Me
End Sub
Private Sub Form_Load()
'bChange = False
cmbSpecies.AddItem ("Catfish")
cmbSpecies.AddItem ("Cod")
cmbSpecies.AddItem ("Flounder")
cmbSpecies.AddItem ("Haddock")
cmbSpecies.AddItem ("Ocean Perch")
cmbSpecies.AddItem ("Polロック")
cmbSpecies.AddItem ("Turbot")
'Set cn = New Connection
cn.Open "Provider = Microsoft.Jet.OLEDB.4.0;Data Source=ADMS.mdb;Persist Security Info=False"
'If bChange = True Then
cmbSpecies.Text = sSpecies
'Else	sSpecies = cmbSpecies.Text
'End If
cmdPreview.Enabled = False
End Sub
'Code for the frame of Income Statement
Option Explicit
'Public tmp_prod_no As String
'Dim cn As Connection
Dim appADMS As Excel.Application
Dim wbADMS As Excel.Workbook
Dim tmp As String

Private Sub cmdExit_Click()
Unload Me
End Sub

Private Sub cmdOk_Click()
    dataIncome.Recordset.MoveLast
dataIncome.Recordset.Update
dataIncome.Refresh
cmdPreview.Enabled = True
End Sub

Private Sub cmdPreview_Click()
Dim rslnFix As Recordset
Dim rslnVar As Recordset
Dim shtInvoice As Excel.Worksheet

On Error Resume Next
Set appADMS = GetObject(
    "Excel.Application"
)If Err.Number <> 0 Then
    Set appADMS = CreateObject("Excel.Application")
End If
Err.Clr
On Error GoTo 0
Set wbADMS =
appADMS.Workbooks.Open(App.Path &
    "\IncomeTemplate.xlt")
wd ADMS.Application.Visible = True
Set shtInvoice = wbADMS.Sheets(1)
Set rslnFix = New Recordset
rslnFix.Open "IncomeFix",
frmDescription.cn, adOpenForwardOnly, adLockReadOnly
rslnFix.MoveFirst
Set rslnVar = New Recordset
rslnVar.Open "IncomeVar",
frmDescription.cn, adOpenForwardOnly, adLockReadOnly
rslnVar.MoveFirst
shtInvoice.Cells(11, 5) =
rslnVar("FreshRev")
shtInvoice.Cells(12, 5) =
rslnVar("FrozenRev")
shtInvoice.Cells(13, 5) = rslnVar("SlabRev")
shtInvoice.Cells(18, 5) =
rslnVar("FishCost")
shtInvoice.Cells(19, 5) =
rslnVar("Overhead")
shtInvoice.Cells(20, 5) = rslnVar("Freight")
shtInvoice.Cells(21, 5) = rslnVar("Packing")
shtInvoice.Cells(26, 5) = rslnVar("Labour")
shtInvoice.Cells(14, 5) =
rslnFix("OtherRev")
shtInvoice.Cells(24, 5) = rslnFix("MisCost")
shtInvoice.Cells(33, 5) = rslnFix("License")
shtInvoice.Cells(34, 5) =
rslnFix("Depreciation")
shtInvoice.Cells(35, 5) = rslnFix("Repairs")
shtInvoice.Cells(36, 5) =
rslnFix("Insurance")
shtInvoice.Cells(37, 5) = rslnFix("Interest")
rslnFix.Close
Set rslnFix = Nothing
rslnVar.Close
Set rslnVar = Nothing
End Sub

Private Sub Form_Load()
'Set cn = New Adodb.Connection
'cn.Open "Provider = Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Documents and
Settings\Guowei\Desktop\ADMS\ADMS.mdb
Persist Security Info=False"
cmdPreview.Enabled = False
txtOtherRev ToolTipText = "The revenue
other than fresh, frozen, or slab"
tblOther ToolTipText = "The revenue other
than fresh, frozen, or slab"
End Sub