

NAVAL POSTGRADUATE SCHOOL MONTEREY, CALIFORNIA



THESIS

**ANALYSIS, DESIGN, AND IMPLEMENTATION
OF A DATABASE MANAGEMENT SYSTEM
(DBMS) FOR COMMANDER IN CHIEF PACIFIC
FLEET PROPULSION EXAMINING BOARD
(CINCPACFLT PEB)**

by

Tony R. Encinias

September, 1995

Thesis Advisor:

Shu Liao

Approved for public release; distribution is unlimited.

19960401 068

DTIC QUALITY INSPECTED 1

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE September 1995	3. REPORT TYPE AND DATES COVERED Master's Thesis		
4. TITLE AND SUBTITLE Analysis, Design, and Implementation of a Database Management System (DBMS) for CINCPACFLT PEB		5. FUNDING NUMBERS		
6. AUTHOR Tony R. Encinias				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000		8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE		
13. ABSTRACT (maximum 200 words) The Commander in Chief Pacific Fleet Propulsion Examining Board (CINCPACFLT PEB) is a professional group of naval officers experienced in engineering readiness. This group of naval officers conducts examinations to assess the engineering readiness of the Pacific Fleet. There are two classes of examinations conducted by the PEB. The Operational Propulsion Plant Examination (OPPE) assesses the engineering readiness when the ship is underway. The Light Off Examination (LOE) assesses the engineering readiness after a ship's propulsion plant has been secured, for example overhauls, in excess of ninety days. The examination data is then collected in the exam summary worksheet and entered into the database. The current database system uses the dBASE IV Plus database management system (DBMS) software and is constructed as a flat file system. Redundant data, inaccuracies, and inefficiencies are prevalent throughout the current system. The purpose of this thesis is to design a relational database (Propulsion Examining Board Database System (PEBDS)), implement and replace the current system, and provide a user friendly graphical user interface (GUI). The PEBDS and the GUI are constructed using Paradox for Windows Version 4.5 DBMS software.				
14. SUBJECT TERMS Database Management System (DBMS), DBMS design, DBMS development, DBMS implementation, Paradox 4.5 for Windows, Propulsion Examining Board Database System (PEBDS)		15. NUMBER OF PAGES * 314		
		16. PRICE CODE		
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

Approved for public release; distribution is unlimited.

**ANALYSIS, DESIGN, AND IMPLEMENTATION OF A DATABASE
MANAGEMENT SYSTEM FOR THE CINCPACFLT PEB**

Tony R. Encinias
Lieutenant, United States Navy
B.A., University of Colorado, 1987

Submitted in partial fulfillment
of the requirements for the degree of

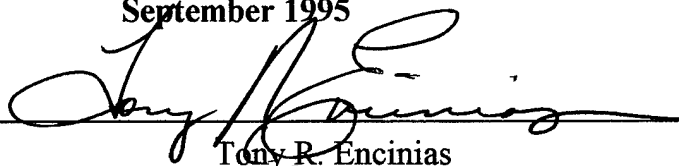
**MASTER OF SCIENCE IN INFORMATION TECHNOLOGY
MANAGEMENT**

from the

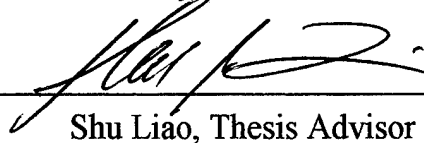
NAVAL POSTGRADUATE SCHOOL

September 1995

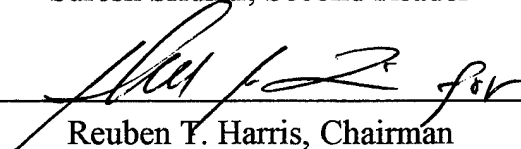
Author:


Tony R. Encinias

Approved by:


Shu Liao, Thesis Advisor


Suresh Sridhar, Second Reader


Reuben T. Harris, Chairman
Department of Systems Management

ABSTRACT

The Commander in Chief Pacific Fleet Propulsion Examining Board (CINCPACFLT PEB) is a professional group of naval officers experienced in engineering readiness. This group of naval officers conducts examinations to assess the engineering readiness of the Pacific Fleet. There are two classes of examinations conducted by the PEB. The Operational Propulsion Plant Examination (OPPE) assesses the engineering readiness when the ship is underway. The Light Off Examination (LOE) assesses the engineering readiness after a ship's propulsion plant has been secured, for example overhauls, in excess of ninety days. The examination data is then collected in the exam summary worksheet and entered into the database.

The current database system uses the dBASE IV Plus database management system (DBMS) software and is constructed as a flat file system. Redundant data, inaccuracies, and inefficiencies are prevalent throughout the current system. The purpose of this thesis is to design a relational database (Propulsion Examining Board Database System (PEBDS)), implement and replace the current system, and provide a user friendly graphical user interface (GUI). The PEBDS and the GUI are constructed using Paradox for Windows Version 4.5 DBMS software.

TABLE OF CONTENTS

I. INTRODUCTION	1
A. BACKGROUND	1
B. PEB DATABASE SYSTEM (PEBDS)	2
C. CHAPTER DESCRIPTIONS	2
II. DATABASE APPLICATION DEVELOPMENT	3
A. PHASE I: DEFINITION	3
B. PHASE II: REQUIREMENTS	4
1. Data Requirements	5
C. PHASE III: EVALUATION	7
D. PHASE IV: DESIGN	8
E. PHASE V: IMPLEMENTATION	9
IV. CONCLUSIONS	11
REFERENCES	13
APPENDIX A. SEMANTIC OBJECT DIAGRAM	15
APPENDIX B. DATA DICTIONARY	17
APPENDIX C. DATA FLOW DIAGRAMS	45
APPENDIX D. PROCESS SPECIFICATIONS	59
A. REPORTS	59
1. Produce CNO Monthly Report (1.1P)	59
2. Produce CINCPACFLT Monthly Report (1.2P)	59

B.	QUERIES	59
1.	Produce 12 Month OPPE Cumulative Trend Query (2.1P)	59
2.	Produce Ship With Special Situations Query (2.2P)	59
3.	Produce Monthly Summary of Activity Query (2.3P)	60
4.	Produce High Power Demo Query (2.4P)	60
5.	Produce Unsat Program Query (2.5P)	60
6.	Produce Pacific Fleet OPPE Summary Query (2.6P)	61
7.	Produce Task and Drill Summary Query (2.7P)	61
8.	Produce Fire Fighting Query (2.8P)	62
9.	Produce Boiler Flex Query (2.9P)	62
C.	EXAM	63
1.	Get Exam Update (3.1.1P)	63
2.	Add Exam (3.1.2P)	63
3.	Change Exam (3.1.3P)	63
4.	Delete Exam (3.1.4P)	63
D.	MATERIAL	63
1.	Get Material Update (3.2.1P)	63
2.	Add Material (3.2.2P)	63
3.	Change Material (3.2.3P)	63
4.	Delete Material (3.2.4P)	64
E.	FIRE FIGHTING	64
1.	Get Fire Fighting Update (3.3.1P)	64
2.	Add Fire Fighting (3.3.2P)	64
3.	Change Fire Fighting (3.3.3P)	64
4.	Delete Fire Fighting (3.3.4P)	64
F.	OPERATIONS	64
1.	Get Operations Update (3.4.1P)	64
2.	Add Operations (3.4.2P)	64
3.	Change Operations (3.4.3P)	64

4.	Delete Operations (3.4.4P)	65
G.	PROGRAM MANAGEMENT	65
1.	Get Program Management Update (3.5.1P)	65
2.	Add Program Management (3.5.2P)	65
3.	Change Program Management (3.5.3P)	65
4.	Delete Program Management (3.5.4P)	65
H.	LEVEL OF KNOWLEDGE	65
1.	Get Level of Knowledge Update (3.6.1P)	65
2.	Add Level of Knowledge (3.6.2P)	65
3.	Change Level of Knowledge (3.6.3P)	65
4.	Delete Level of Knowledge (3.6.4P)	66
I.	TRAINING	66
1.	Get Training Update (3.7.1P)	66
2.	Add Training (3.7.2P)	66
3.	Change Training (3.7.3P)	66
4.	Delete Training (3.7.4P)	66
J.	SHIP	66
1.	Get Ship Update (3.8.1P)	66
2.	Add Ship (3.8.2P)	66
3.	Change Ship (3.8.3P)	67
4.	Delete Ship (3.8.4P)	67
APPENDIX E. RELATIONAL DIAGRAM		69
APPENDIX F. INPUT AND QUERY FORMS		71
APPENDIX G. OBJECTPAL SOURCE CODE		81
APPENDIX H. USER'S MANUAL		291

INITIAL DISTRIBUTION LIST	303

I. INTRODUCTION

A. BACKGROUND

The Commander in Chief Pacific Fleet Propulsion Examining Board (CINCPACFLT PEB) was established to ensure the engineering readiness needs of the Pacific Fleet. Its specific mission is to conduct Operational Propulsion Plant Examinations (OPPE) and Light Off Examinations (LOE). These examinations serve as a verification to CINCPACFLT as to the degree of engineering readiness of the Pacific Fleet. The OPPE is an operational examination which verifies engineering readiness in six different areas. These areas consist of the following: Material Readiness, Fire Fighting, Program Management, Training, Level of Knowledge, and Operations. The LOE is an examination which verifies engineering readiness when the propulsion plant has been secured, for example overhauls, for more than ninety days. The LOE is required before lighting off the propulsion plant and verifies engineering readiness in five different areas. These areas consist of the following: Material Readiness, Fire Fighting, Program Management, Training, and Level of Knowledge.

The PEB is segregated into three sub-organizations: Gas Turbine, Steam, and Diesel. These sub-organizations are representations of the various propulsion types that are currently installed on Pacific Fleet ships. Within these sub-organizations, teams are selected to conduct the examinations and consist of five examiners. The team examiners include a senior examiner, usually a 0-5 or 0-6 and four junior examiners, usually a 0-3 or 0-4. Additionally a Project Officer (PO) is chosen from the junior examiners. He or she is responsible for the overall coordination of the examination and is responsible for maintaining the examination summary worksheet.

The examination summary worksheet contains all the vital data for each examination area. When the examinations are complete, the PO is required to submit the examination summary worksheet to the Database Administrator (DBA) for entry into the database.

The current database was developed in house using dBASE IV Plus for DOS database management system (DBMS) software. This database, however, is a flat file system and has resulted in numerous instances of redundant data. As a result, the performance of the database is very poor and the report and query processing is very inefficient and inaccurate.

The DBA requested a study be generated regarding improvements to the current database system. He suggested developing a graphical user interface (GUI), improving database performance, and providing efficient and accurate reports and queries. In addition, the senior examiner responsible for statistical information on examinations suggested providing graphical presentations of queries. This thesis proposes a system designed to accomplish these tasks.

B. PEB DATABASE SYSTEM (PEBDS)

The Propulsion Examining Board Database System (PEBDS) is designed to replace the current database system and make retrieval and storage of information easier and efficient. Additionally, providing a user friendly GUI and graphical presentations of queries will facilitate decision making for all levels of management. To accomplish this, the author interviewed junior and senior PEB examiners to determine their requirements.

The DBMS software that was chosen to construct the PEBDS is Paradox for Windows Version 4.5. The PEBDS is a menu driven application and mimics a Windows application. This ensures that users with a background in a Windows environment can easily navigate around the PEBDS without having knowledge of Paradox for Windows.

C. CHAPTER DESCRIPTIONS

Chapter II will discuss the application development methodology used in developing the PEBDS. The definition, requirements, evaluation, design, and implementation phases will be discussed. This chapter will define the scope and objectives of the application as well as how these objectives will be accomplished.

Chapter III will discuss conclusions, discussions, and recommendations.

Appendices A. through H. provides supporting and substantiation of requirements, data dictionary, application documentation (ObjectPAL text), data flow diagrams (DFD's), and a user's guide.

II. DATABASE APPLICATION DEVELOPMENT

The PEBDS was developed using five phases. The five phases are the definition, requirements, evaluation, design, and implementation. These phases and their requirements will be discussed in this chapter.

A. PHASE I: DEFINITION

The current database system has several problems. It is unable to provide accurate and timely data, the queries are static, and it uses a flat file configuration which allows storing redundant data. The current system uses dBASE IV Plus DBMS, which is not user friendly and users require training.

The CINCPACFLT PEB has recently upgraded to IBM compatible 486 PC's and is operating in a Windows environment. The users are asking for a user friendly graphical user interface (GUI) to interact with the database.

The scope of this study is to build a relational database system with a GUI, which will run in a Windows environment. In addition, support for dynamic queries and graphical displays of query results will be developed.

After the problem has been defined and the scope of the project established, the feasibility of the project must be determined. Areas to look at in establishing the feasibility are cost, time, and schedule constraints. The cost of the project was not considered relevant because the hardware and software were already available. The schedule was set to begin in April 1995 with a system completion date set for August - September 1995. The time and schedule to complete the project is reasonable.

Benefits from implementing the PEBDS are as follows:

- Easier system to learn requiring less training
- Time savings for data entry personnel

- Higher integrity of data
- Increased ability to conduct statistical analysis of data

B. PHASE II: REQUIREMENTS

There are two major styles for developing a database. Top-down development proceeds from the generic to the specific. It begins with a study of the strategic goals of the organization, the means by which those goals can be accomplished, the information requirements that must be satisfied to accomplish those goals, and the systems that must exist to provide that information. For such a study, a data model is developed at a high level of abstraction. [Ref. 5:p. 84]

Bottom-up development operates in the reverse order of abstraction. It begins with the need to develop a specific system. The means of selecting the first system varies from organization to organization. In some organizations, a steering committee picks the application, in other organization, the users may pick it themselves; and in some, the loudest voice in the executive rank wins out. [Ref. 5:p. 84]

The first decision was to decide which database development style to use. Bottom-up development was selected because it produces quick results, is less risky, and does not contribute to the phenomenon of "analysis paralysis" as does the Top-down development style. The next step in this phase is to interview all possible users of the system. The author interviewed the DBA and the leading senior examiner in charge of constructing statistical analysis reports on examination data. They both described the functional requirements in great detail. The initial interviews lasted approximately four hours and were beneficial in constructing the initial prototype of the system. They specifically requested the following reports and queries:

- CNO monthly report
- CINCPACFLT monthly report
- Twelve Month OPPE Cumulative Trend query

- Ships With Special Situations query
- Monthly Summary of Activity query
- High Power Demonstrations query
- Program Management query
- Exam Area Summary query
- Evolutions and Drills Summary query
- Fire Fighting query
- Boiler Flexibility Checks query
- ECCTT query
- Level of Knowledge query

The initial prototype was completed in July 1995. Minor changes were required and no further prototypes were necessary.

1. Data Requirements

The semantic object modeling approach was chosen over the conventional entity-relational approach to model the PEBDS data. Semantic object and entity-relational modeling are alike in that both try to represent the user's view of the data. They are also similar with how they describe attributes, cardinalities, and domains. In semantic object modeling, however, each object in the semantic object diagram holds all the details about itself. With entity-relational modeling entities, one must look at several other entities to get the "big picture." This makes semantic object modeling objects more comprehensible than entity-relational modeling entities. One can see the relationships between objects without having to trace through a network of lines as one must do for the entity-relational model. The semantic object diagram for the PEBDS is located in Appendix A.

Based on the user interviews and prototype process, it was determined that eight objects were necessary to meet the requirements of the PEBDS. The objects for the PEBDS

are as follows:

- SHIP
- EXAM
- OPERATIONS
- MATERIAL
- FIRE FIGHTING
- PROGRAM MANAGEMENT
- TRAINING
- LEVEL OF KNOWLEDGE

The SHIP object is the central object. SHIP will hold all the information of all ships in the Pacific Fleet. This includes ship name, propulsion type, commanding officer, executive officer, chief engineer, ISIC, previous examination grade, and hull number. This will act as a lookup table for all other tables. The data must be entered first in this table before any other data is entered into the system. The EXAM object identifies examinations conducted and containing information on the particular examination. This includes exam ending date, adjective grade, overall finding, ship name, next exam date, project officer, senior examiner, examination type, and examination comments. OPERATIONS is a subtype of EXAM contains information on the watch sections, evolutions, drills, ECCTT, and overall grade. MATERIAL is a subtype of EXAM and contains information on high power demonstrations, material self assessment, valve maintenance, gage calibrations, cleanliness, preservation, stowage, number of major and minor discrepancies, boiler flexibility, material comments, number of reported and uncovered degradations. FIREFIGHTING is a subtype of EXAM and contains information on the number of major and minor damage control and repair five discrepancies, repair five inventory, AFFF grade, halon grade, fire drill grade and comments, DCTT grade and comments and overall grade. PROGRAM MANAGEMENT is a subtype

of EXAM and contains information on boiler water/feed water, lube oil quality management, fuel oil quality management, diesel jacket water test and treatment, diesel engine trend analysis, operating logs, legal records, bearing records, marine gas turbine equipment service records, tag out, electrical safety, online verification, hearing conservation, quality assurance, and overall finding grades and comments. TRAINING is a subtype of EXAM and contains information on personal qualification standards, key personnel, training program, number of satisfactory gas turbine watch stations, steam watch stations, and diesel watch stations, and overall grade and comments. LEVEL OF KNOWLEDGE is a subtype of EXAM and contains information on passing percentage for the written , supervisory, and damage control examinations.

The above objects are displayed in the semantic object diagram in Appendix A. The object and domain definitions are displayed in the data dictionary in Appendix B.

The collection of data flow diagrams is displayed in Appendix C. These diagrams describe the overall flow of the information in the system. Attributes and functions of the PEBDS are listed with the process specifications in Appendix D.

The PEBDS will have a back up capability utilizing the Microsoft backup application installed with Windows 3.1. This will be a menu option on the main menu for the user. The database files should be backed up on a weekly basis as a minimum.

C. PHASE III: EVALUATION

Evaluation of the PEBDS project was completed and produced two system constraints. The first constraint was the hardware the PEBDS had to run on. For efficient operation, the minimum hardware requirement is an IBM compatible 486 PC with 4 MB of memory. This constraint, however, did not pose any significant technological barriers for development of the PEBDS. The second constraint was the DBMS software. Since limited monetary resources were available, the PEBDS is required to be designed and operated with Paradox for Windows Version 4.5 DBMS software already installed on site. An extensive review of this DBMS software determined that it satisfied all and possible future requirements of the PEBDS.

The user's requirements were further reviewed and it was determined that the PEBDS could be completed on time and utilize the current hardware and DBMS software available. No additional costs would be necessary unless recommendations for future enhancements are desired.

D. PHASE IV: DESIGN

The logical database design is centered around the primary object SHIP. The key of the SHIP object is (ShipName) and contains one-to-many semantic object attribute links to the following objects:

- EXAM
- OPERATIONS
- MATERIAL
- FIRE FIGHTING
- PROGRAM MANAGEMENT
- TRAINING
- LEVEL OF KNOWLEDGE

The minimum cardinality is equal to zero and the maximum cardinality is equal to N.

The key of the EXAM object is a composite key composed of (ExamEndDate) and the semantic object link (SHIP). The cardinality of the semantic object link is one-to-one with the minimum and maximum cardinality equal to one.

The following objects are subtypes of the EXAM object and inherit the same composite key and minimum and maximum cardinalities:

- OPERATIONS
- MATERIAL
- FIRE FIGHTING

- PROGRAM MANAGEMENT
- TRAINING
- LEVEL OF KNOWLEDGE

The above objects and relationships are represented graphically in the Relational Diagram, Appendix E and the data definitions in Appendix B.

The main menu, input forms, and query forms are listed in Appendix F. The ObjectPAL source code for the PEBDS is listed in Appendix G and the user's guide is listed in Appendix H.

E. PHASE V: IMPLEMENTATION

The PEBDS was implemented in August 1995. The PEBDS was installed on a IBM compatible 486 PC. The PEBDS will utilize the current software installed on the PC. The PEBDS will use Paradox for Windows Version 4.5 and Windows 3.1. Upgrades to both software applications are available but is not within the budgetary considerations for fiscal year 1996. If upgrades are procured, the PEBDS will operate without restrictions on Paradox for Windows 5.0 and Windows 95 software upgrades.

A strategy for implementation of the PEBDS was considered and a pilot strategy was decided upon. The reason this strategy was chosen was due in part to the incompatibilities of the flat file system with the new relational PEBDS. The data in the single table format of the current database could not be converted to the new multi table relational PEBDS. Therefore, the pilot strategy will enable the users to use the PEBDS for monthly summary queries and inputs for the monthly reports to CINCPACFLT and CNO. Until the full conversion is complete, the DBA will have to maintain both systems. Full conversion will take place in approximately six months when enough records are within the system to facilitate the other queries within the PEBDS. The pilot strategy will enable the organization to gradually implement the PEBDS to ensure stability and minimize the risk of data loss.

The input and query screens, and reports were built using Paradox's screen painter. Color, font size, and overall design considerations were easily facilitated with the screen

painter and a ergonomical environment was created. Since Paradox for Windows application language is object-oriented, it was used to provide links, correlations and relationships between each object within the PEBDS. The custom source code used to construct the PEBDS is transparent to the user, but is a major reason of why the PEBDS is user friendly. All objects within the PEBDS have custom source code embedded and the source code is listed in Appendix G.

Training and familiarization with the PEBDS was held with the DBA. A user's guide was provided to help the DBA and any other PEB staff members to easily navigate through the PEBDS and is listed in Appendix H. Since the PEBDS is a menu driven application which uses pushbuttons and mouse point and click procedures, lengthy training sessions were not required. The DBA received additional training in system installation, troubleshooting, and data backup.

Testing of the PEBDS was done during the implementation phase and throughout the entire life cycle of the project. The testing procedures were conducted on sample controlled data and stressed the PEBDS beyond what one would expect the users of the PEBDS would do. The data was entered and all queries were executed to see if the expected results would be given. Modifications to the PEBDS were done to correct the minor inconsistencies found throughout the testing procedure.

Restructuring the PEBDS to accommodate future enhancements can be accomplished with the design feature of Paradox for Windows. The original system files (.db, .fsl, .rsl) were provided to the DBA, on two 3.5 inch floppy disks. The design feature for these system files was not disabled to allow access to every object within the PEBDS. However, the enhancements to the PEBDS will require extensive knowledge of the application language objectPAL and Paradox for Windows and should be attempted by only qualified programmers.

IV. CONCLUSIONS

The PEBDS is currently installed and is running in accordance with the design specifications of the users. It has significantly reduced the time the DBA spends on data entry and query processing for the senior members of the PEB. The monthly summary reports can now be completed in less than thirty minutes, which includes the printing of the query reports and graphs. The PEBDS has increased the productivity of all who use the PEBDS and the generation of future enhancements is highly recommended. The users have shown that they are satisfied with the system so far and to date the system has achieved the goals that it was intended to accomplish.

The user's requirements have been met, but as with all new systems, the users can now envision new requirements that would be equally beneficial. As the users develop new requirements, the objectPAL source code can be changed or added to better meet these new requirements. Minor changes or additions to the objectPAL source code and maintenance of the PEBDS can be accomplished by the DBA. If these changes are major however, they may be extensive enough to have a following on effort by another thesis student.

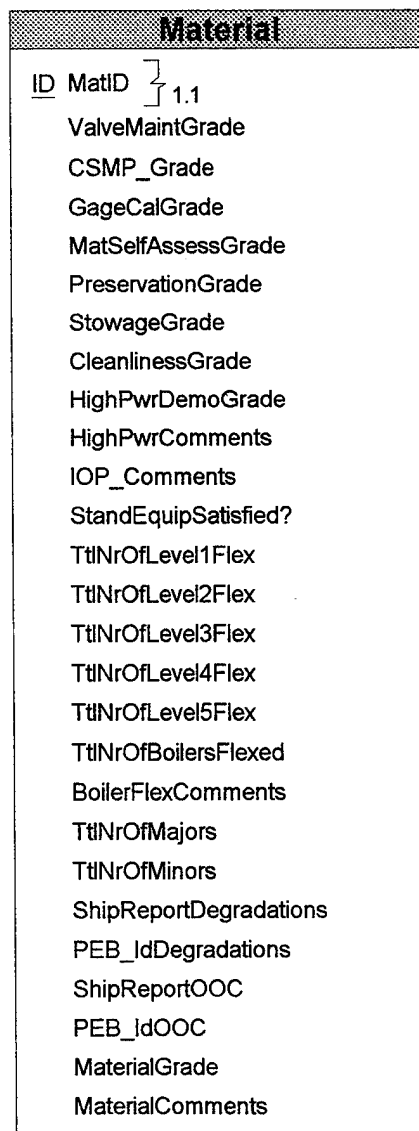
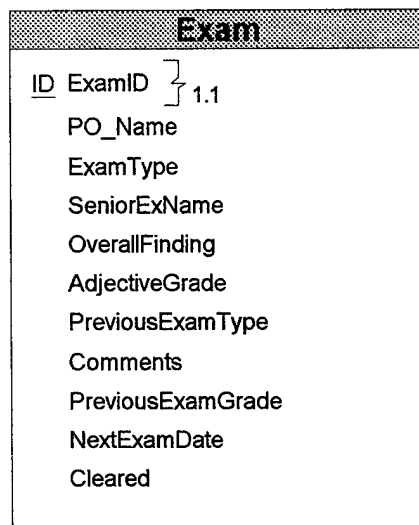
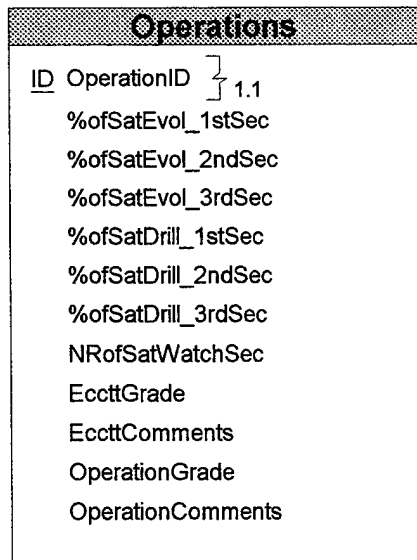
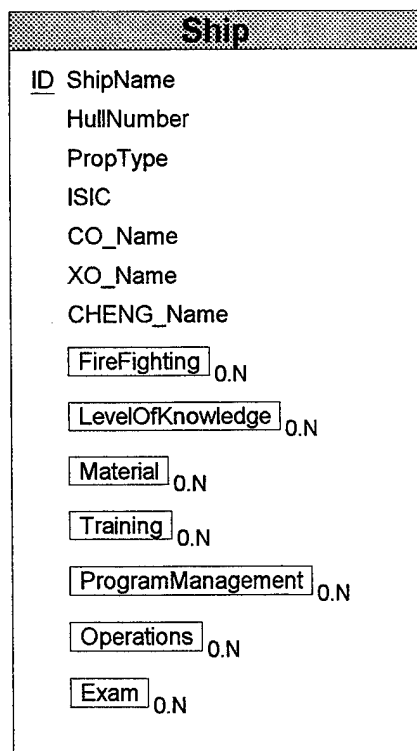
Modifications and upgrades to the office's infrastructure, hardware and software are an ongoing issue for every office environment. The PEBDS is versatile and can operate on any software upgrades to Paradox for Windows. If a major change occurs with the infrastructure, the PEBDS can accommodate this change to the infrastructure, but will require personnel knowledgeable with the PEBDS, Paradox for Windows, and Paradox for Windows application language objectPAL.

Although the backup procedures installed with the PEBDS is sufficient and will satisfy all backup requirements, the time required to backup on 3.5 inch floppy disks will increase as more and more records are introduced into the database. The addition of a backup tape drive will facilitate the back up procedures and provide easy and relatively less painful means of restoring the PEBDS data in the event of total data loss. A tape backup will solve this problem and would only require minimal additional funds.

REFERENCES

1. Dewitz, Sandra. , Semantic Object Modeling with Salsa, McGraw-Hill, Inc., 1994.
2. Gurewich, Ori., Paradox 4.5 for Windows Unleashed, Sams Publishing, 1994.
3. Hawryskiewicz, I. T., Introduction to Systems Analysis and Design, Prentice Hall of Australia Pty Ltd, 1988.
4. Jensen, Cary., Programming Paradox 4.5 for Windows, 2nd Ed., Sybex Inc., 1994.
5. Kroenke, David M., Database Processing, 4th Ed., Macmillan, 1992.
6. Shlaer Sally, Object-Oriented Systems Analysis, Prentice Hall, Inc., 1988.
7. Tinney, Diane., Paradox for Windows Programming by Example, Que, 1993.
8. Yourdon, Edward., Modern Structured Analysis, Prentice Hall Inc., 1989.

APPENDIX A. Semantic Object Diagram



ProgramManagement

ID PM_ID } 1.1
LOQM_Grade
BWWF_Grade
FOQM_Grade
DJWTT_Grade
DETA_Grade
OpLogsGrade
LegalRecsGrade
BearingRecsGrade
MGTESR_Grade
TagoutGrade
ElectSafetyGrade
HearingConsGrade
QA_Grade
OLV_Grade
ProgramManageGrade
ProgramComments

FireFighting

ID FireFightingID } 1.1
NrOfDC_Majors
NrOfDC_Minors
NrOfRepV_Majors
NrOfRepV_Minors
RepV_InventoryGrade
MSFD_Grade
SpaceDC_EquipGrade
DCTT_Grade
DCTT_Comments
HalonGrade
AFFF_Grade
FireDrill1_Grade
FireDrill2_Grade
FireDrill3_Grade
FireDrillComments
FireFightingGrade
FireFightingComments

Training

ID TrainingID } 1.1
PQS_Grade
TrainingGrade
NrOfSatBlrOp
NrOfSatGenOp
NrOfSatBTOW/ConsoleOP
NrOfSatMsgr/EngOp
NrOfSatMMOW
NrOfSatENOW
NrOfSatEDG/SWBD_Op
NrOfSatEoow
NrOfSatPACC
NrOfSatEPCC
NrOfSatAuxOP
NrOfSatOilKing
NrOfSatMEROP
KeyPersonnelLeaving
TrainingProgramGrade
TrainingProgramComments

LevelOfKnowledge

ID LOK_ID } 1.1
%PassWrittenExam
AvgScoreWritten
%PassDC_Exam
AvgScoreDC
%PassSupExam
AvgScoreSup
AvgScoreEM_Sup

APPENDIX B. DATA DICTIONARY

%ofSatDrill_1stSec	Type: Simple Value Profile: Percent Contained in: Operations Caption: Description: The percentage of satisfactory drills in the 1st section ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Floating Point Length: Format: Initial Value:

%ofSatDrill_2ndSec	Type: Simple Value Profile: Percent Contained in: Operations Caption: Description: The percentage of satisfactory drills in the 2nd watch section ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Floating Point Length: Format: Initial Value:

%ofSatDrill_3rdSec	Type: Simple Value Profile: Percent Contained in: Operations Caption: Description: The percentage of satisfactory drills in the 3rd watchsection ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Floating Point Length: Format: Initial Value:

%ofSatEvol_1stSec	Type: Simple Value Profile: Percent Contained in: Operations Caption: Description: The percent of satisfactory evolutions in the 1st watchsection ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Floating Point Length: Format: Initial Value:

%ofSatEvol_3rdSec Type: Simple Value
Profile: Percent
Contained in: Operations
Caption:
Description: The percentage of satisfactory evolutions
 in the 3rd section
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Floating Point
Length:
Format:
Initial Value:

%PassDC_Exam Type: Simple Value
Profile: Percent
Contained in: LevelOfKnowledge
Caption:
Description: The percentage of engineering personnel
 passing the damage control exam
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Floating Point
Length:
Format:
Initial Value:

%PassSupExam Type: Simple Value
Profile: Percent
Contained in: LevelOfKnowledge
Caption:
Description: The percentage of supervisors passing th
 e supervisory exam
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Floating Point
Length:
Format:
Initial Value:

%PassWrittenExam Type: Simple Value
Profile: Percent
Contained in: LevelOfKnowledge
Caption:
Description: The percentage of watchstanders passing
 the written exam
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Floating Point
Length:
Format:
Initial Value:

AdjectiveGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Exam
Caption:
Description: The descriptive grade for the exam
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

AFFF_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The grade on the ship's AFFF system ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:
AvgScoreDC	Type: Simple Value Profile: Quantity Contained in: LevelOfKnowledge Caption: Description: The average score on the damage control exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
AvgScoreEM_Sup	Type: Simple Value Profile: Quantity Contained in: LevelOfKnowledge Caption: Description: The average score of the EM supervisory exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
AvgScoreSup	Type: Simple Value Profile: Quantity Contained in: LevelOfKnowledge Caption: Description: The average score on the supervisory exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
AvgScoreWritten	Type: Simple Value Profile: Quantity Contained in: LevelOfKnowledge Caption: Description: The average score for the written exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

BearingRecsGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The bearing records program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

BoilerFlexComments	Type: Simple Value Profile: Description Contained in: Material Caption: Description: The comments on the boiler flexes ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:

BWFW_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The boiler water/feed water program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

CHENG_Name	Type: Simple Value Profile: PersonName Contained in: Ship Caption: Description: The chief engineer's last name ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Text Length: 35 Format: Initial Value:

CleanlinessGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: The grade for cleanliness ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

Cleared	Type: Simple Value Profile: Cleared Contained in: Exam Caption: Description: This is to check whether a ship's exam is cleared from the database ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 1 Format: Initial Value: Y
CO_Name	Type: Simple Value Profile: PersonName Contained in: Ship Caption: Description: The commanding officers last name ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Text Length: 35 Format: Initial Value:
Comments	Type: Simple Value Profile: Description Contained in: Exam Caption: Description: The comments on the exam overall ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:
CSMP_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: The grade for the consolidated ships maintenance plan ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:
DCTT_Comments	Type: Simple Value Profile: Description Contained in: FireFighting Caption: Description: The damage control training team comments ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:

DCTT_Grade Type: Simple Value
Profile: PreviousExamGrade
Contained in: FireFighting
Caption:
Description: The damage control training team grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

DETA_Grade Type: Simple Value
Profile: PreviousExamGrade
Contained in: ProgramManagement
Caption:
Description: The diesel engine trend analysis program
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

DJWTT_Grade Type: Simple Value
Profile: PreviousExamGrade
Contained in: ProgramManagement
Caption:
Description: The diesel jacket water test and treatment program grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

EccttComments Type: Simple Value
Profile: Description
Contained in: Operations
Caption:
Description: The comments for the engineering casualty control training team
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Memo
Length:
Format:
Initial Value:

EccttGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Operations
Caption:
Description: The grade of the engineering casualty control training team
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

ElectSafetyGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: ProgramManagement
Caption:
Description: The electrical safety program grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

Exam Type: Object Link
Profile: Exam
Contained in: Ship
Caption:
Description:
ID Status: None
Minimum Required: 0
Maximum Allowed: N (No Limit)

ExamEndDate Type: Simple Value
Profile: ExamEndDate
Contained in: LevelOfKnowledge.LOK_ID
Caption:
Description: The date the exam ends
ID Status: None
Minimum Required: 1
Maximum Allowed: 1
Value Type: Date
Length:
Format:
Initial Value:

ExamEndDate Type: Simple Value
Profile: ExamEndDate
Contained in: Training.TrainingID
Caption:
Description: The date the exam ends
ID Status: None
Minimum Required: 1
Maximum Allowed: 1
Value Type: Date
Length:
Format:
Initial Value:

ExamEndDate Type: Simple Value
Profile: ExamEndDate
Contained in: Material.MatID
Caption:
Description: The date the exam ends
ID Status: None
Minimum Required: 1
Maximum Allowed: 1
Value Type: Date
Length:
Format:
Initial Value:

ExamEndDate	Type: Simple Value Profile: ExamEndDate Contained in: ProgramManagement.PM_ID Caption: Description: The date the exam ends ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Date Length: Format: Initial Value:

ExamEndDate	Type: Simple Value Profile: ExamEndDate Contained in: Exam.ExamID Caption: Description: The date the exam ends ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Date Length: Format: Initial Value:

ExamEndDate	Type: Simple Value Profile: ExamEndDate Contained in: Operations.OperationID Caption: Description: The exam end date ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Date Length: Format: Initial Value:

ExamEndDate	Type: Simple Value Profile: ExamEndDate Contained in: FireFighting.FireFightingID Caption: Description: The date the exam ends ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Date Length: Format: Initial Value:

ExamID	Type: Group Profile: ExamID Contained in: Exam Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate

ExamType	Type: Simple Value Profile: ExamType Contained in: Exam Caption: Description: The type of inspection with values of OPP E/LOE/REOPPE/RELOE ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

FireDrill1_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The grade on fire drill 1 ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

FireDrill2_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The grade on fire drill 2 ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

FireDrill3_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The grade of fire drill 3 ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

FireDrillComments	Type: Simple Value Profile: Description Contained in: FireFighting Caption: Description: The comments on all 3 fire drills ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:

FireFighting	Type: Object Link Profile: FireFighting Contained in: Ship Caption: Description: ID Status: None Minimum Required: 0 Maximum Allowed: N (No Limit)		
FireFightingComments	Type: Simple Value Profile: Description Contained in: FireFighting Caption: Description: The fire fighting area comments ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:		
FireFightingGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The fire fighting area grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
FireFightingID	Type: Group Profile: FireFightingID Contained in: FireFighting Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate
FOQM_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The fuel oil quality management program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		

GageCalGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Material
Caption:
Description: The grade for gage calibrations program
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

HalonGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: FireFighting
Caption:
Description: The grade of the installed halon system
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

HearingConsGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: ProgramManagement
Caption:
Description: The hearing conservation program grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

HighPwrComments Type: Simple Value
Profile: Description
Contained in: Material
Caption:
Description: The comments on the high power demon
stration
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Memo
Length:
Format:
Initial Value:

HighPwrDemoGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Material
Caption:
Description: The high power demonstration grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

LevelOfKnowledge	Type: Object Link Profile: LevelOfKnowledge Contained in: Ship Caption: Description: ID Status: None Minimum Required: 0 Maximum Allowed: N (No Limit)		
LOK_ID	Type: Group Profile: LOK_ID Contained in: LevelOfKnowledge Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate
LOQM_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The lube oil quality management program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
Material	Type: Object Link Profile: Material Contained in: Ship Caption: Description: ID Status: None Minimum Required: 0 Maximum Allowed: N (No Limit)		
MaterialComments	Type: Simple Value Profile: Description Contained in: Material Caption: Description: The comments on the material area of the exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:		
MaterialGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: The grade for the material area of the ex am ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		

MatID	Type: Group Profile: MatID Contained in: Material Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate
MatSelfAssessGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: The grade for material self assessment ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
MGTESR_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The marine gas turbine equipment service record program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
MSFD_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The main space fire doctrine grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
NextExamDate	Type: Simple Value Profile: EventDate Contained in: Exam Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Date Length: Format: The date the next exam is due Initial Value:		

NrOfDC_Majors Type: Simple Value
Profile: Quantity
Contained in: FireFighting
Caption:
Description: The total number of damage control major
s
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

NrOfDC_Minors Type: Simple Value
Profile: Quantity
Contained in: FireFighting
Caption:
Description: The number of damage control minors
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

NrOfRepV_Majors Type: Simple Value
Profile: Quantity
Contained in: FireFighting
Caption:
Description: The number of repair five majors
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

NrOfRepV_Minors Type: Simple Value
Profile: Quantity
Contained in: FireFighting
Caption:
Description: The number of repair five minors
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

NrOfSatAuxOP Type: Simple Value
Profile: Quantity
Contained in: Training
Caption:
Description: The number of satisfactory auxiliary oper
ators
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

NrOfSatBlrOp	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: Number of satisfactory boiler operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatBTOW/ConsoleOPT	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory boiler technicians/Console operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatEDG/SWBD_Op	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory emergency diesel/switchboard operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatENOW	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory emergency diesel generator/switchboard operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatEoow	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory engineering officers of the watch ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

NrOfSatEPCC	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of electrical plant control console operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatGenOp	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory generator operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatMEROP	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory of main engine room operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatMMOW	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory messengers of the watch ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatMsgR/EngOp	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory messenger/engine operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

NrOfSatOilKing	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory oil kings ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NrOfSatPACC	Type: Simple Value Profile: Quantity Contained in: Training Caption: Description: The number of satisfactory propulsion auxiliary control console operators ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:
NRofSatWatchSec	Type: Simple Value Profile: Identifier-Numeric Contained in: Operations Caption: Description: Total number of satisfactory watch sections ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Long Integer Length: Format: Initial Value:
OLV_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The online verification program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:
OperationComments	Type: Simple Value Profile: Description Contained in: Operations Caption: Description: The comments for the operation area of the exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Memo Length: Format: Initial Value:

OperationGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Operations Caption: Description: The grade for operations area of the exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

OperationID	Type: Group Profile: OperationID Contained in: Operations Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate

Operations	Type: Object Link Profile: Operations Contained in: Ship Caption: Description: ID Status: None Minimum Required: 0 Maximum Allowed: N (No Limit)

OpLogsGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The operation logs program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

OverallFinding	Type: Simple Value Profile: PreviousExamGrade Contained in: Exam Caption: Description: The overall grade for the exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

PEB_IdDegradations	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: Total number of equipment degradations found by the propulsion examining board ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:		
PEB_IdOOC	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The number of out of commission equipment found by the propulsion examining board ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:		
PM_ID	Type: Group Profile: PM_ID Contained in: ProgramManagement Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate
PO_Name	Type: Simple Value Profile: PO_Name Contained in: Exam Caption: Description: The name of the project officer conducting the exam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
PQS_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: Training Caption: Description: The personal qualifications standard grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		

PreservationGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Material
Caption:
Description: The preservation of equipment grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

PreviousExamGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Exam
Caption:
Description: The grade of the previous exam
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

PreviousExamType Type: Simple Value
Profile: PreviousExamType
Contained in: Exam
Caption:
Description: The previous exam type the ship had
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 6
Format:
Initial Value:

ProgramComments Type: Simple Value
Profile: Description
Contained in: ProgramManagement
Caption:
Description: The program management comments
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Memo
Length:
Format:
Initial Value:

ProgramManageGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: ProgramManagement
Caption:
Description: The program managment grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

ProgramManagement	Type: Object Link Profile: ProgramManagement Contained in: Ship Caption: Description: ID Status: None Minimum Required: 0 Maximum Allowed: N (No Limit)
PropType	Type: Simple Value Profile: PO_Name Contained in: Ship Caption: Description: The propulsion type of the ship. Values are Gas Turbine, Diesel, or Steam ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:
QA_Grade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The quality assurance program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:
RepV_InventoryGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The grade of the repair five inventory ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:
SeniorExName	Type: Simple Value Profile: PO_Name Contained in: Exam Caption: Description: The name of the senior member of the PE B team ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

Ship	Type: Object Link Profile: Ship Contained in: Material.MatID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

Ship	Type: Object Link Profile: Ship Contained in: Operations.OperationID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

Ship	Type: Object Link Profile: Ship Contained in: ProgramManagement.PM_ID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

Ship	Type: Object Link Profile: Ship Contained in: Exam.ExamID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

Ship	Type: Object Link Profile: Ship Contained in: LevelOfKnowledge.LOK_ID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

Ship	Type: Object Link Profile: Ship Contained in: FireFighting.FireFightingID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

Ship	Type: Object Link Profile: Ship Contained in: Training.TrainingID Caption: Description: ID Status: None Minimum Required: 1 Maximum Allowed: 1

ShipName	Type: Simple Value Profile: ShipName Contained in: Ship Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

ShipReportDegradations	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The number of equipment degradations reported by the ship ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

ShipReportOOC	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The number of out of commission equipment reported by the ship ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

SpaceDC_EquipGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: FireFighting Caption: Description: The grade on the in space damage control equipment ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

StandEquipSatisfied?	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: Is standard equipment satisfied ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 1 Format: Initial Value:

StowageGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: The grade for stowage ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
TagoutGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: ProgramManagement Caption: Description: The tagout program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
Training	Type: Object Link Profile: Training Contained in: Ship Caption: Description: ID Status: None Minimum Required: 0 Maximum Allowed: N (No Limit)		
TrainingGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Training Caption: Description: The training program grade ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:		
TrainingID	Type: Group Profile: TrainingID Contained in: Training Caption: Description: ID Status: Unique Minimum Required: 1 Maximum Allowed: 1 Minimum Count: 0 Maximum Count: ALL	Attributes Contained:	Ship ExamEndDate

TrainingProgramComments Type: Simple Value
Profile: Description
Contained in: Training
Caption:
Description: The training program area comments
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Memo
Length:
Format:
Initial Value:

TrainingProgramGrade Type: Simple Value
Profile: PreviousExamGrade
Contained in: Training
Caption:
Description: The training program area grade
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Text
Length: 10
Format:
Initial Value:

TtlNrOfBoilersFlexed Type: Simple Value
Profile: Quantity
Contained in: Material
Caption:
Description: The total number of boiler flexes
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

TtlNrOfLevel1Flex Type: Simple Value
Profile: Quantity
Contained in: Material
Caption:
Description: Total number of boiler flexes to level 1
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

TtlNrOfLevel2Flex Type: Simple Value
Profile: Quantity
Contained in: Material
Caption:
Description: The total number of boiler flexes to level 2
ID Status: None
Minimum Required: 0
Maximum Allowed: 1
Value Type: Short Integer
Length:
Format:
Initial Value:

TtlNrOfLevel3Flex	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The total number of boiler flexes to level 3 ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

TtlNrOfLevel4Flex	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The total number of boiler flexes to level 4 ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

TtlNrOfLevel5Flex	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The total number of boiler flexes to level 5 ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

TtlNrOfMajors	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: The total number of majors ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

TtlNrOfMinors	Type: Simple Value Profile: Quantity Contained in: Material Caption: Description: the total number of minors ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Short Integer Length: Format: Initial Value:

ValveMaintGrade	Type: Simple Value Profile: PreviousExamGrade Contained in: Material Caption: Description: The grade for valve maintenance ID Status: None Minimum Required: 0 Maximum Allowed: 1 Value Type: Text Length: 10 Format: Initial Value:

XO_Name	Type: Simple Value Profile: PersonName Contained in: Ship Caption: Description: The executive officer's last name ID Status: None Minimum Required: 1 Maximum Allowed: 1 Value Type: Text Length: 35 Format: Initial Value:

APPENDIX C. DATA FLOW DIAGRAMS

Context Diagram

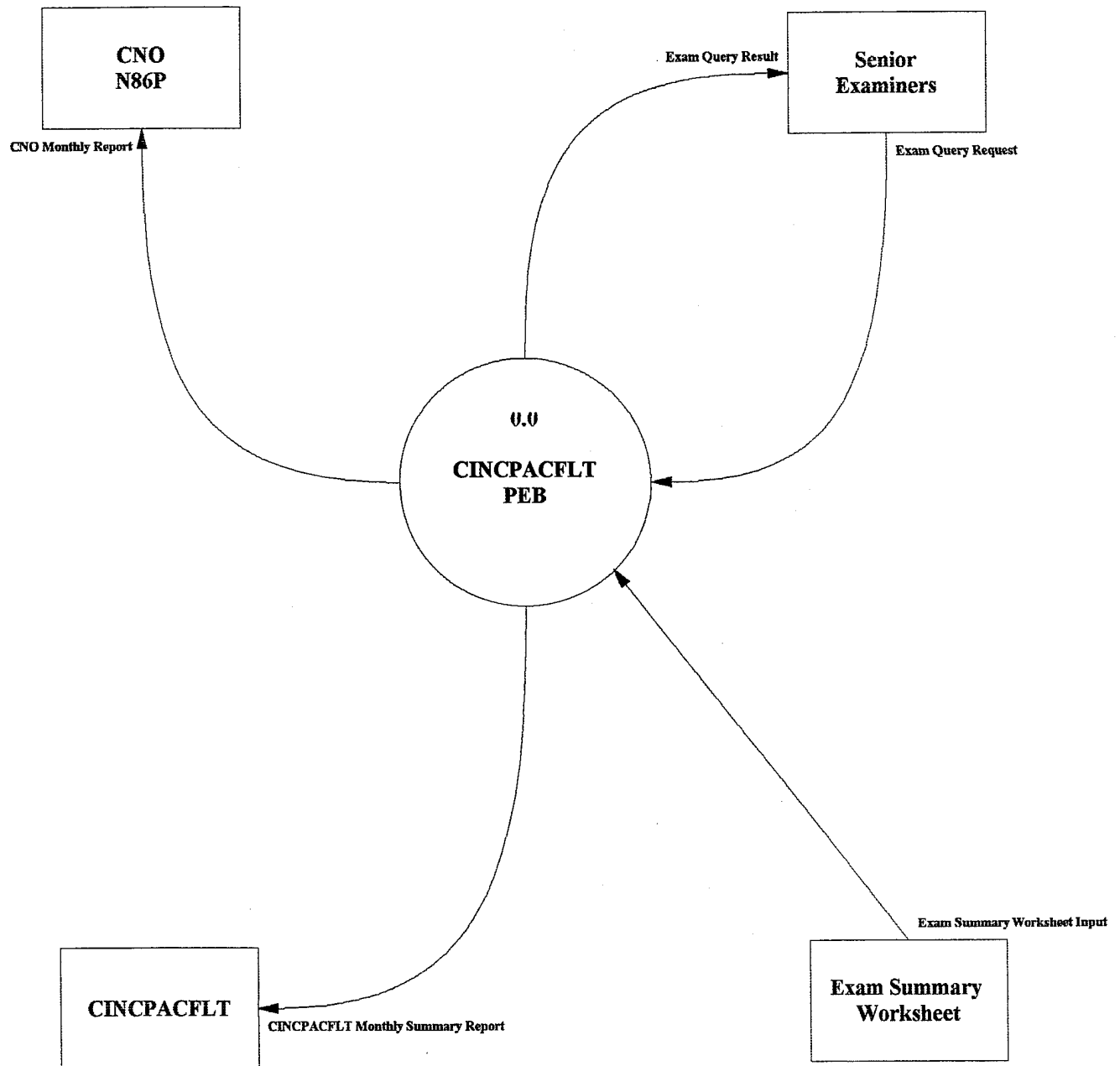


**Figure 0
Diagram**

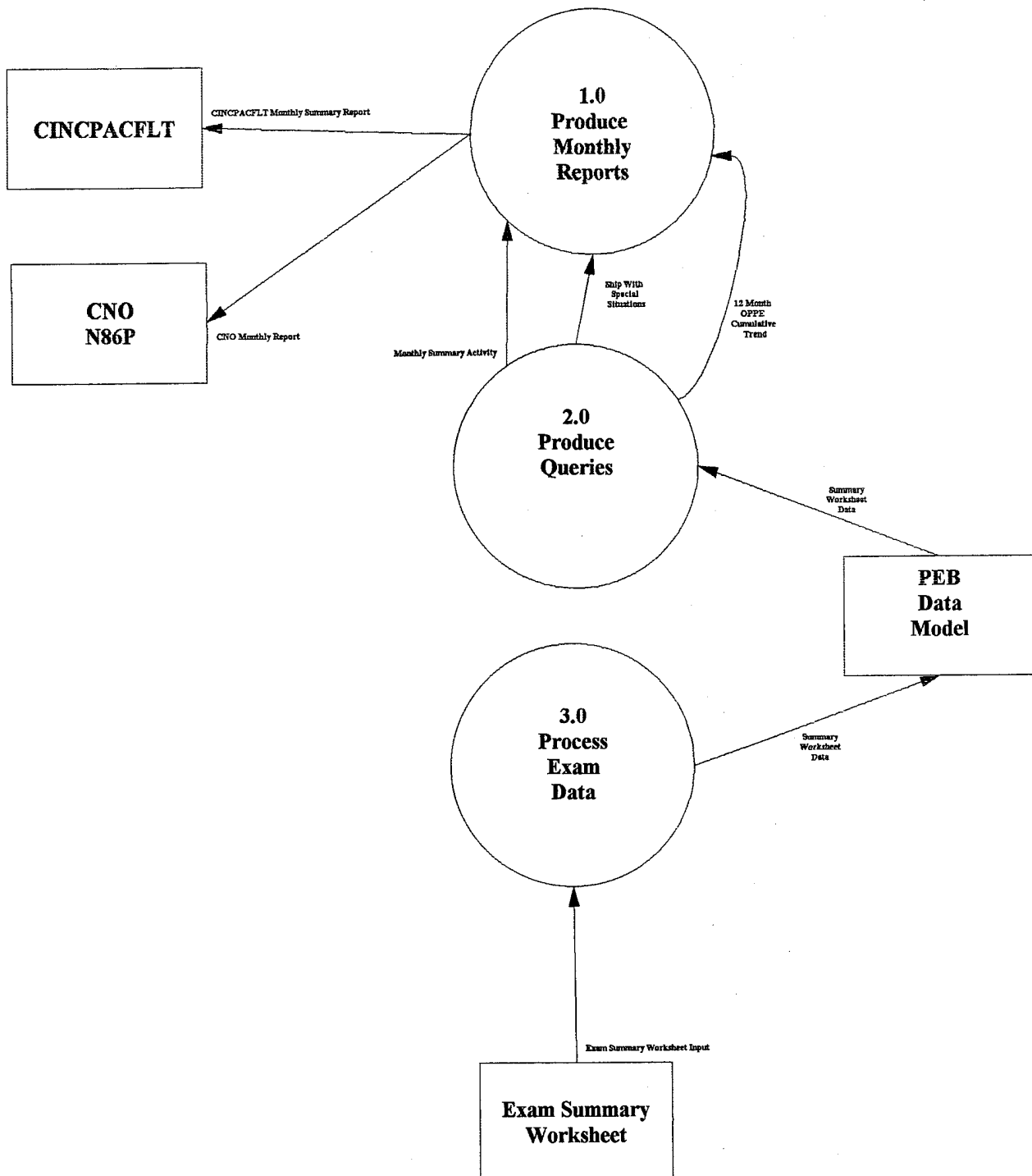


**Figure 1
Diagram**

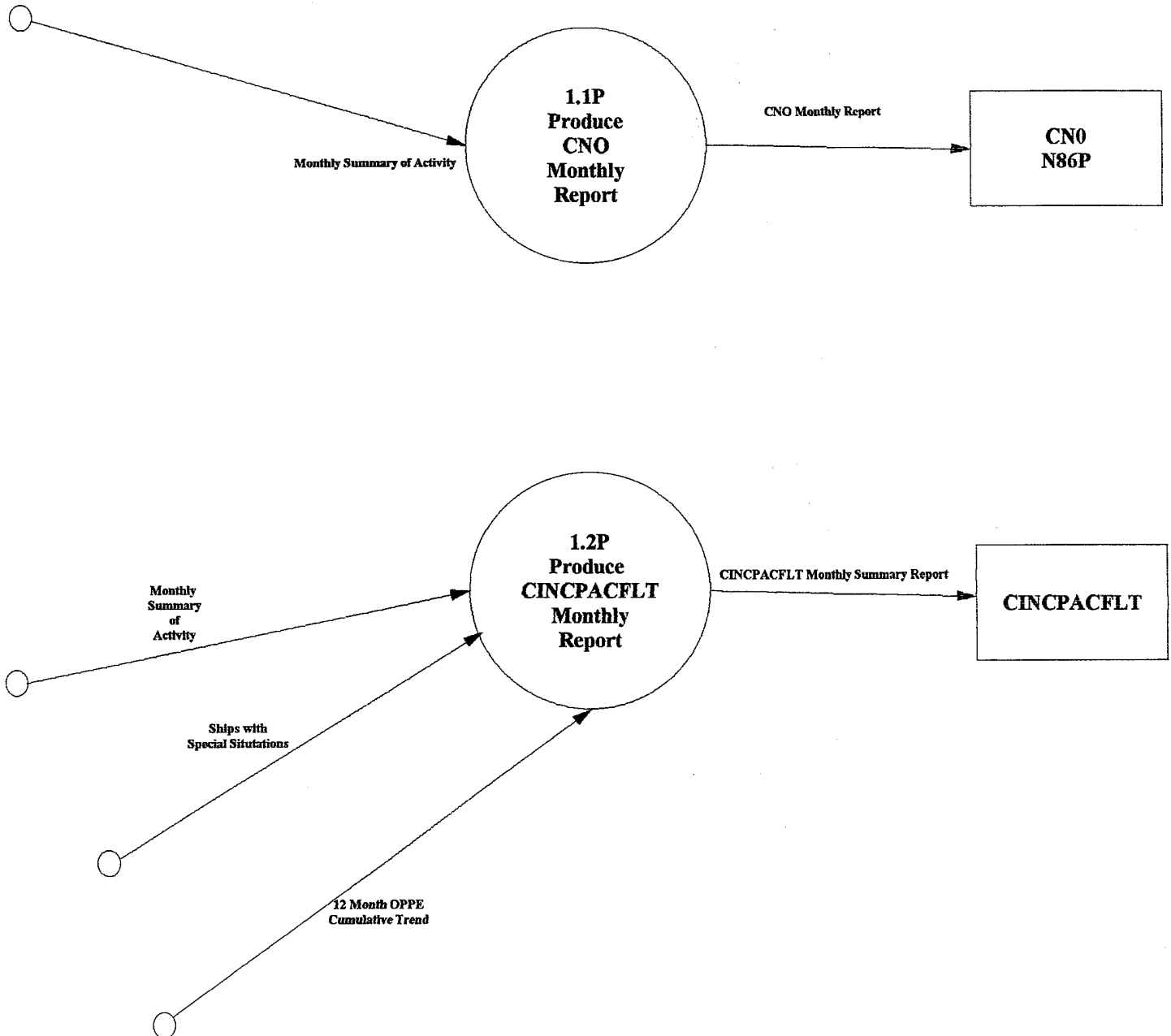


**Figure 2
Diagram**

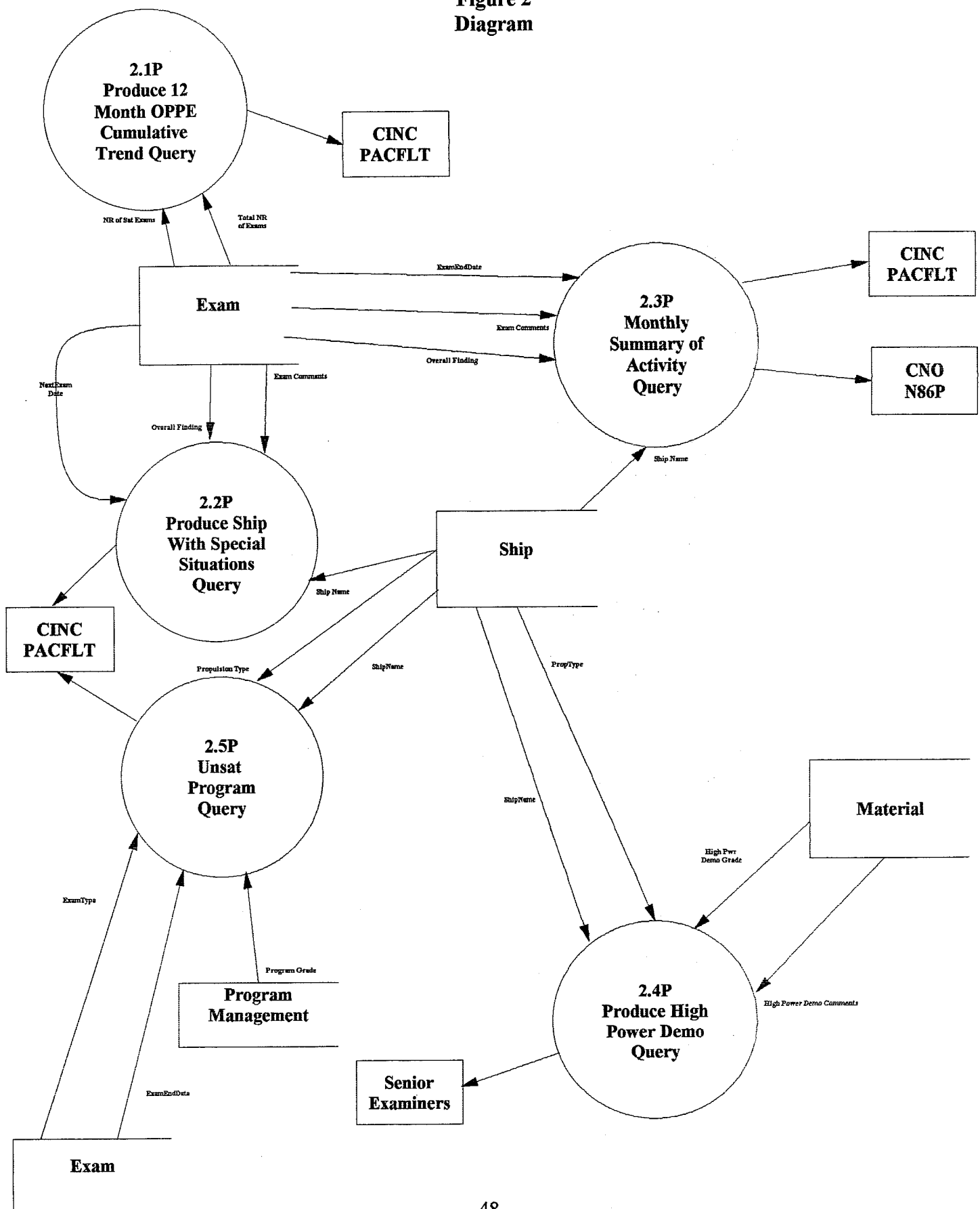


Figure 2
Diagram

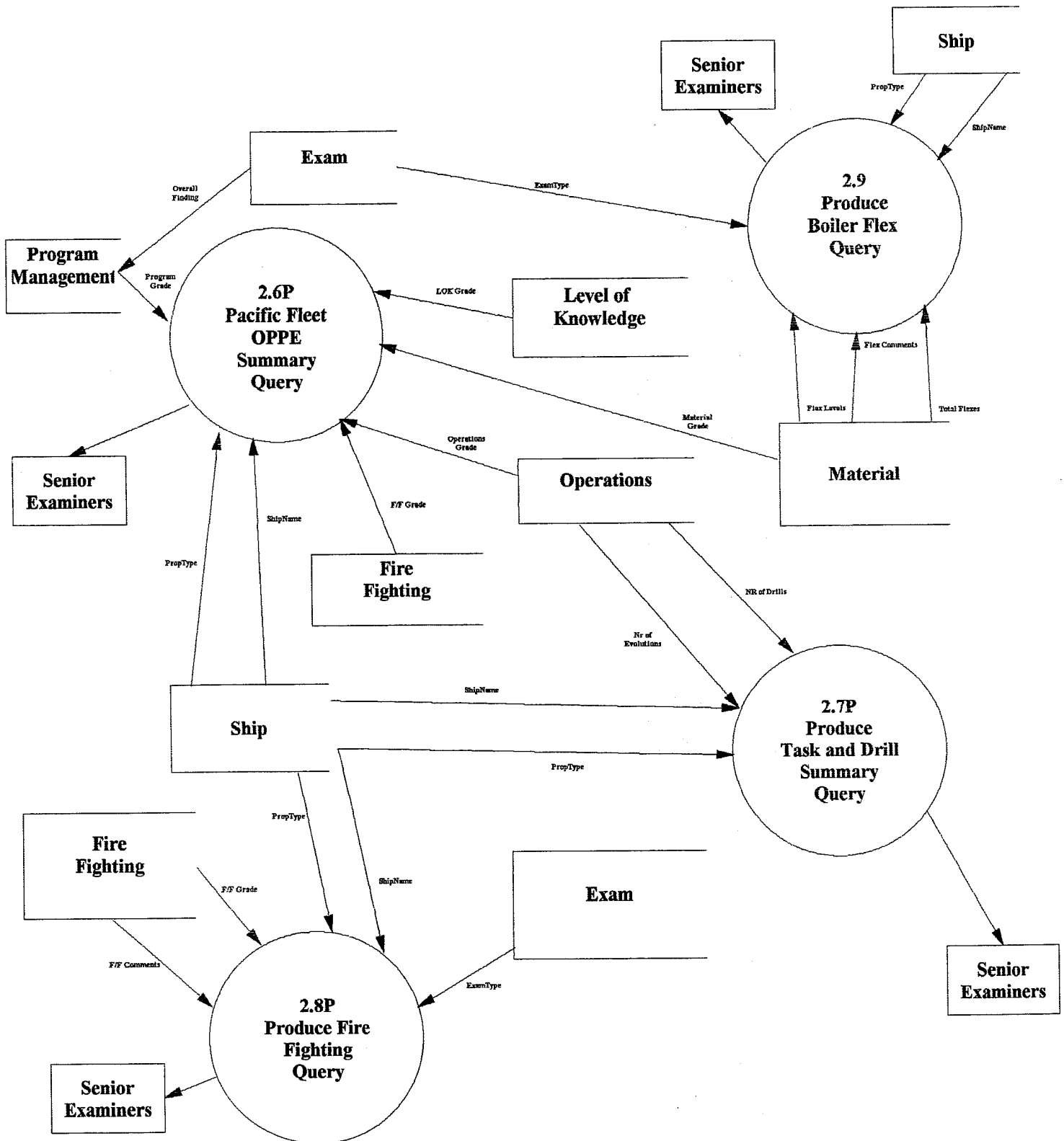


Figure 3
Diagram

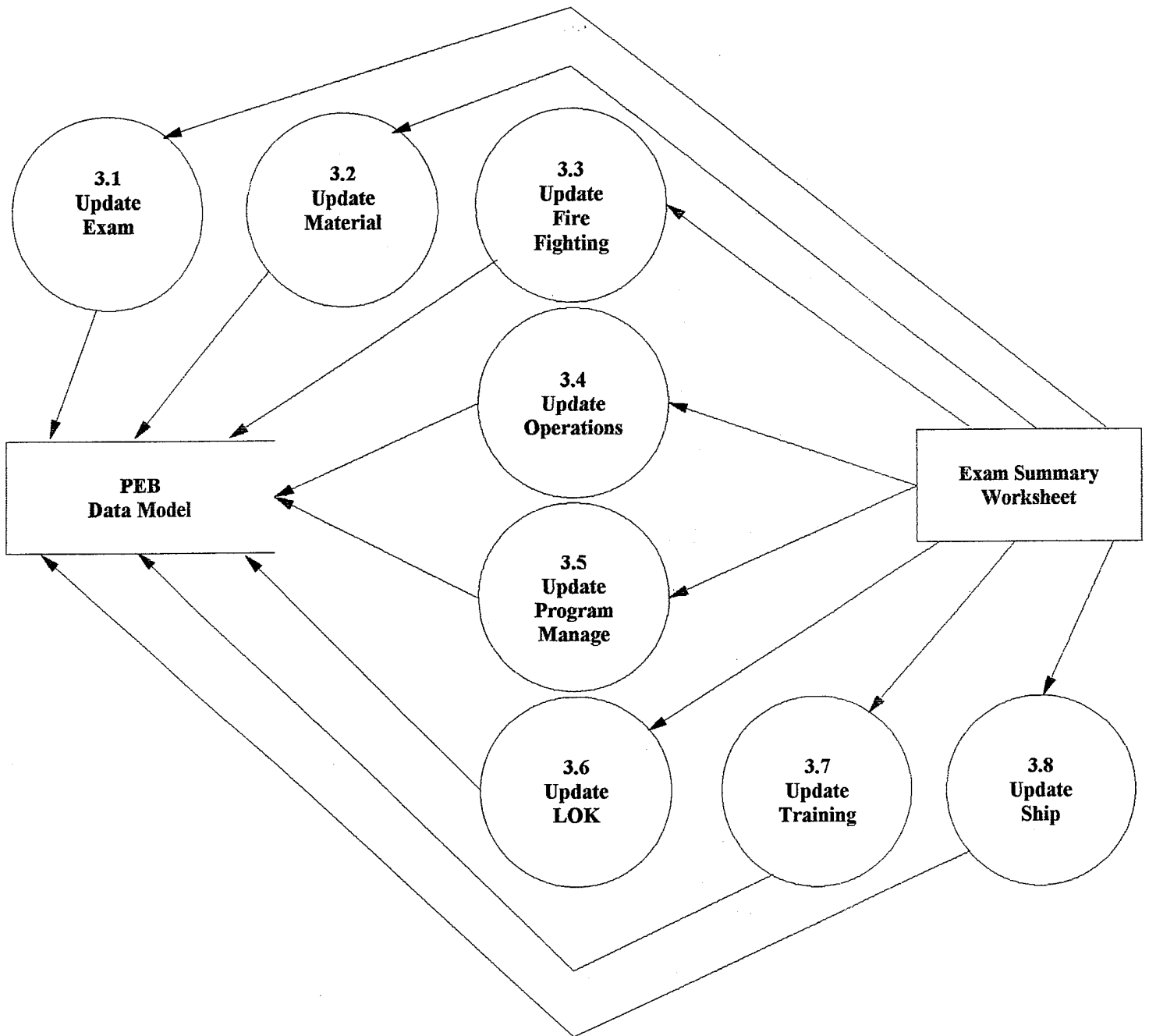


Figure 3.1
Diagram

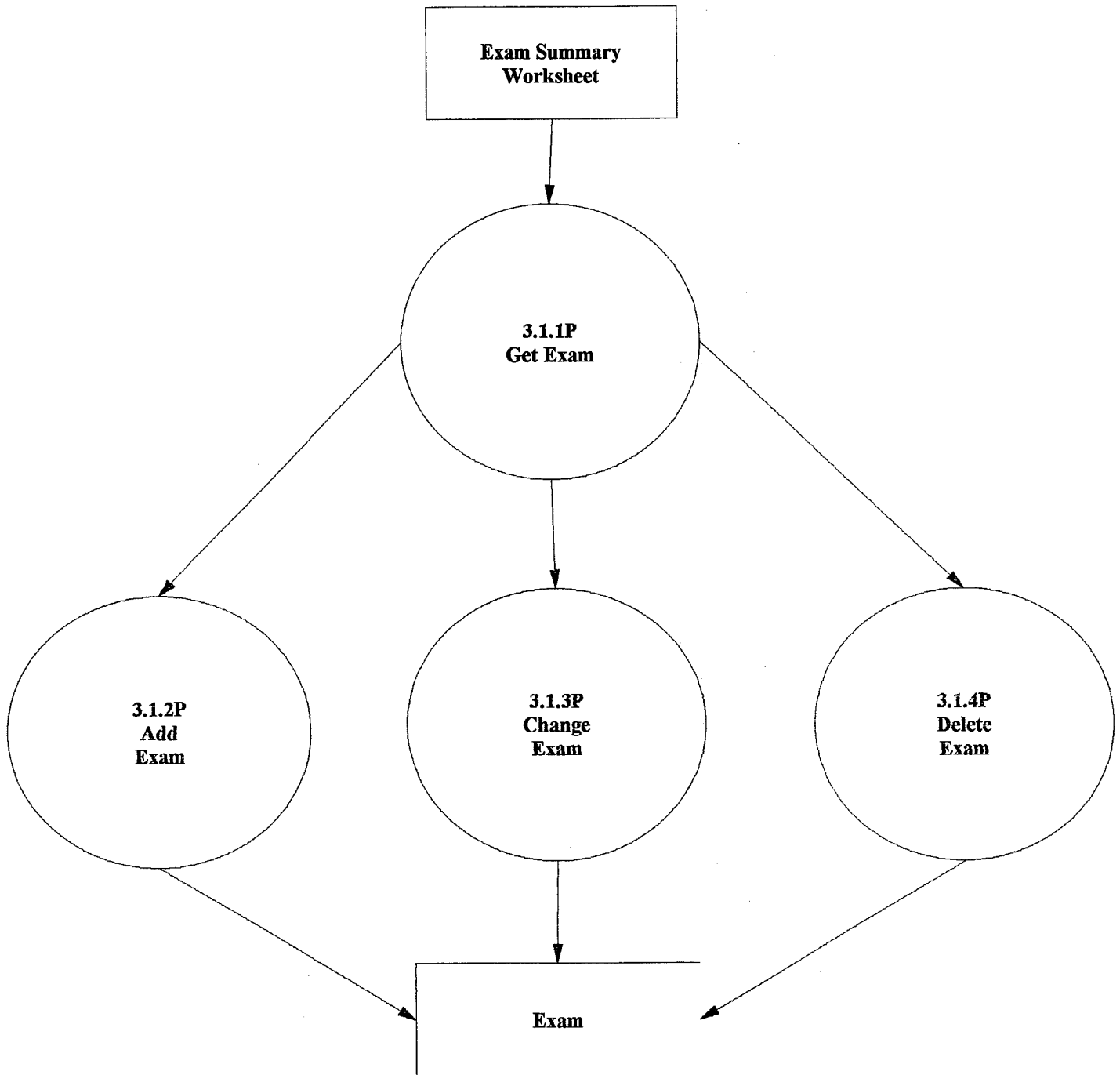


**Figure 3.2
Diagram**

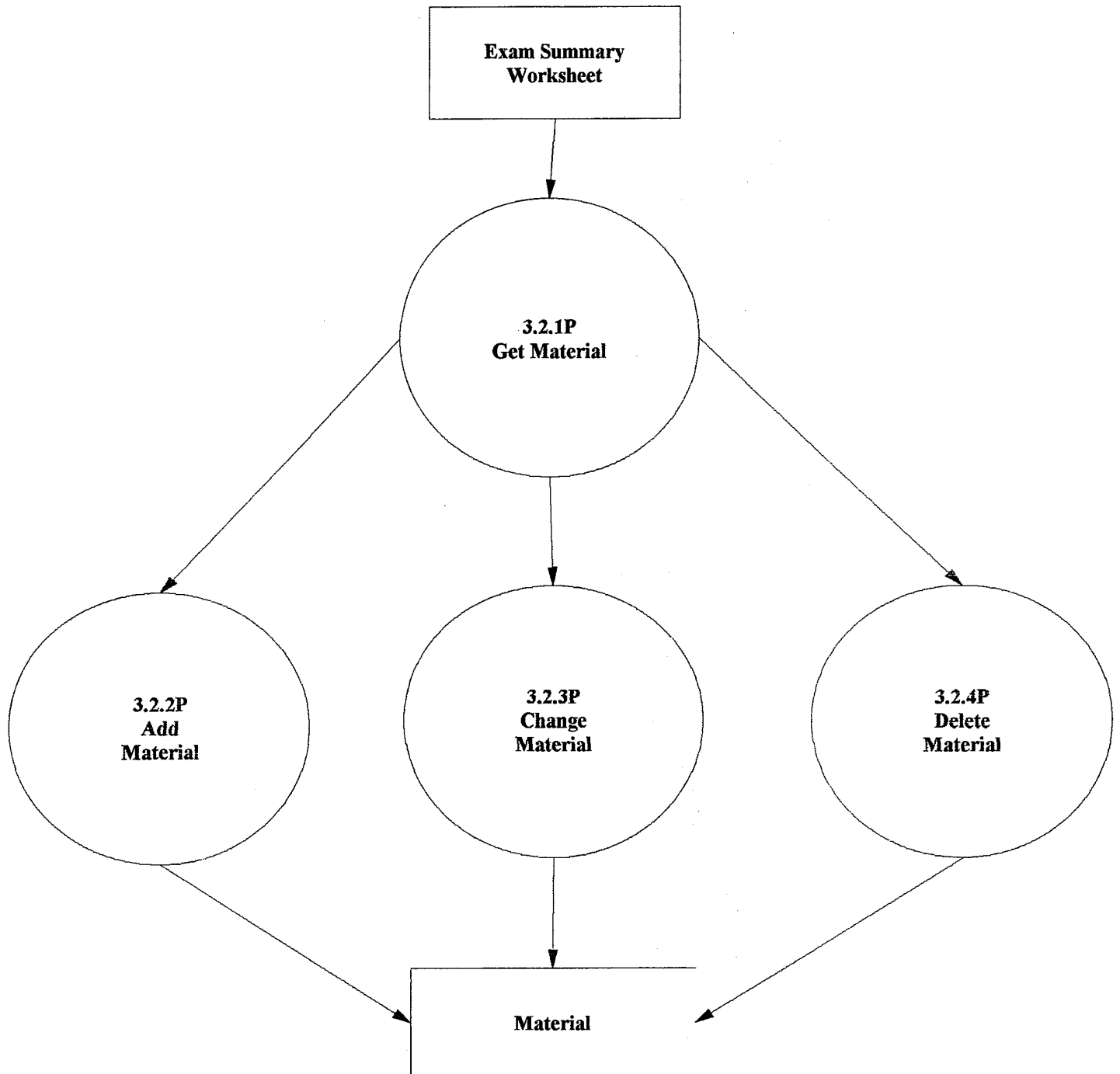


**Figure 3.3
Diagram**

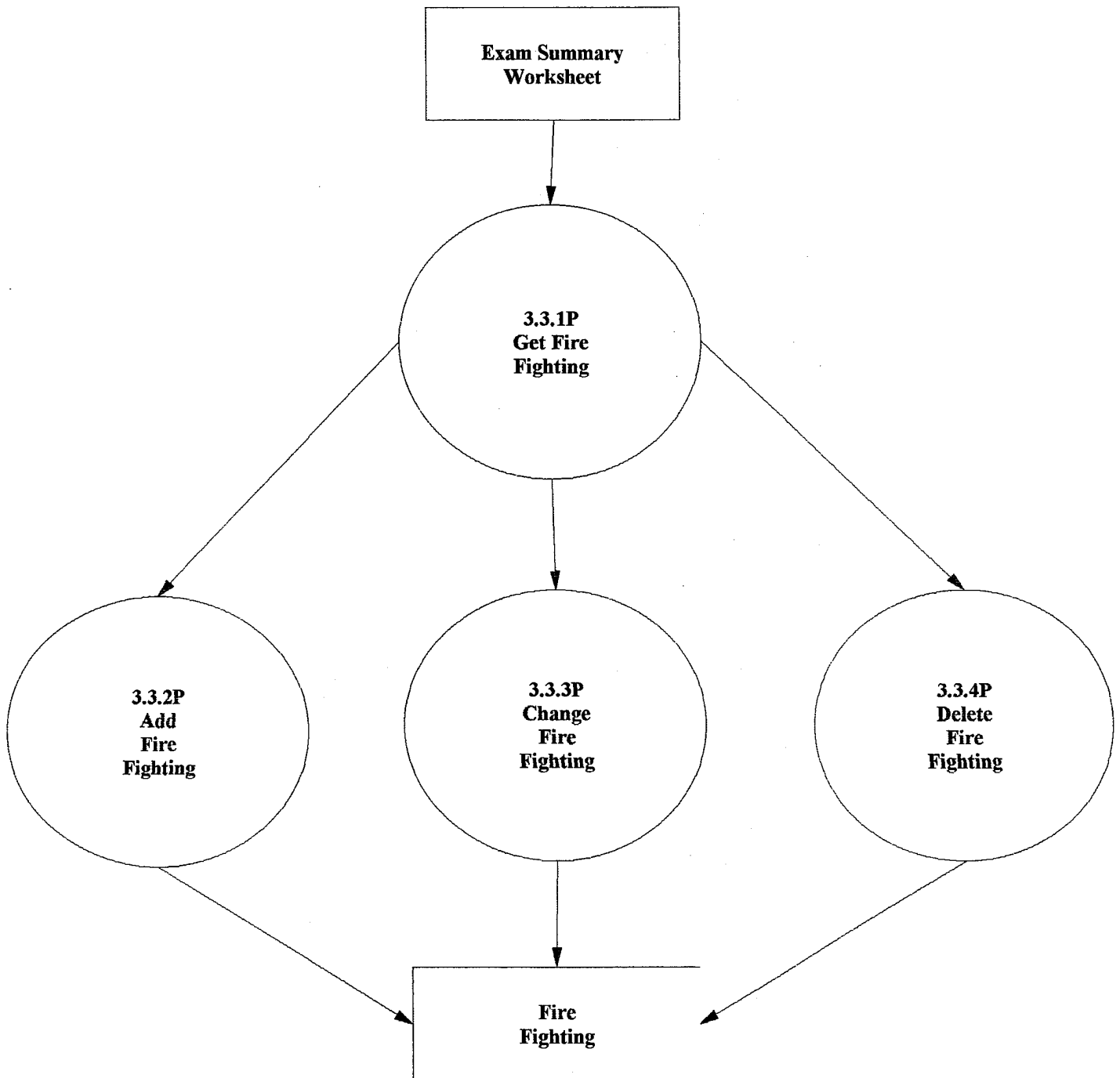


**Figure 3.4
Diagram**

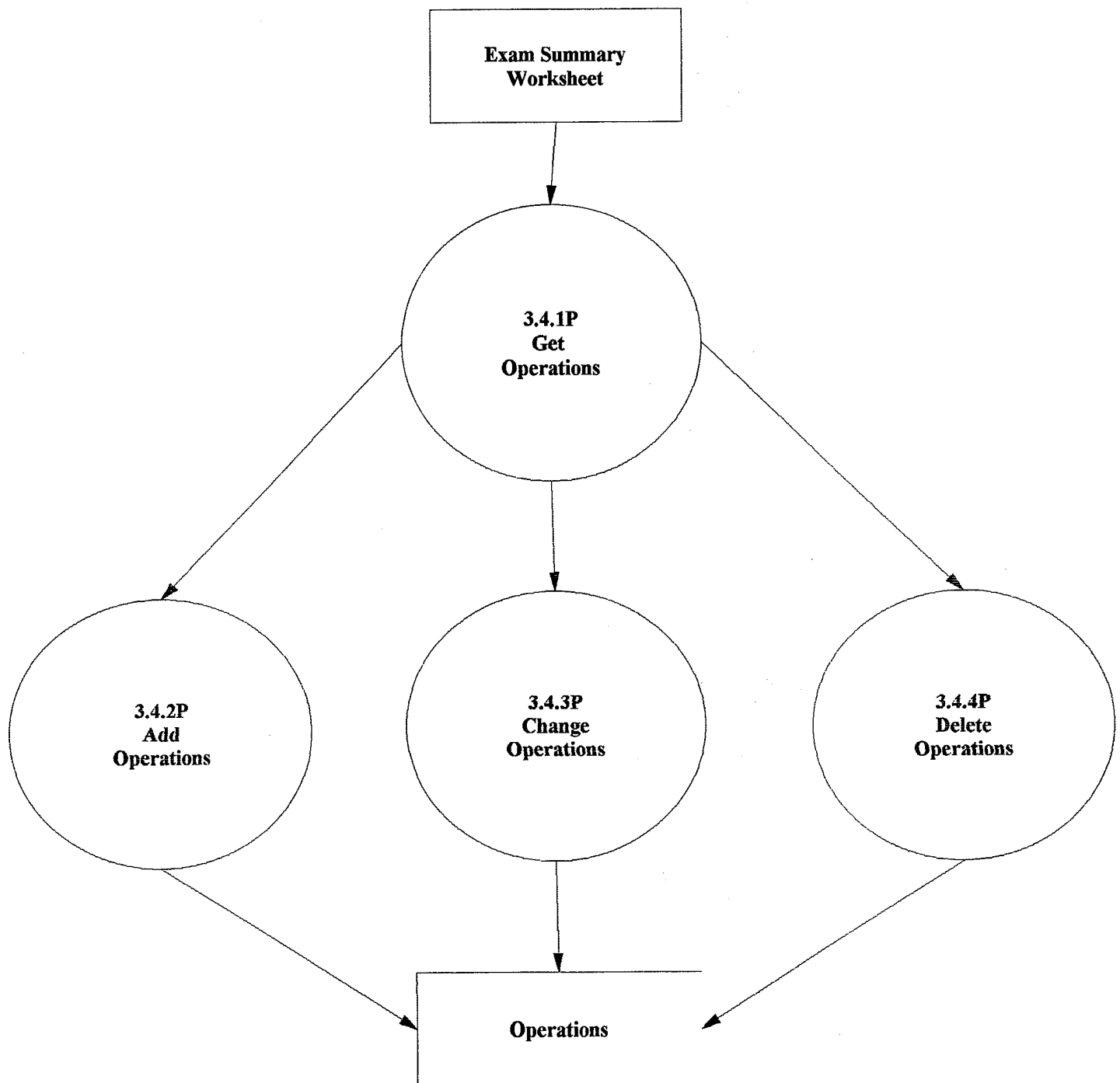


**Figure 3.5
Diagram**

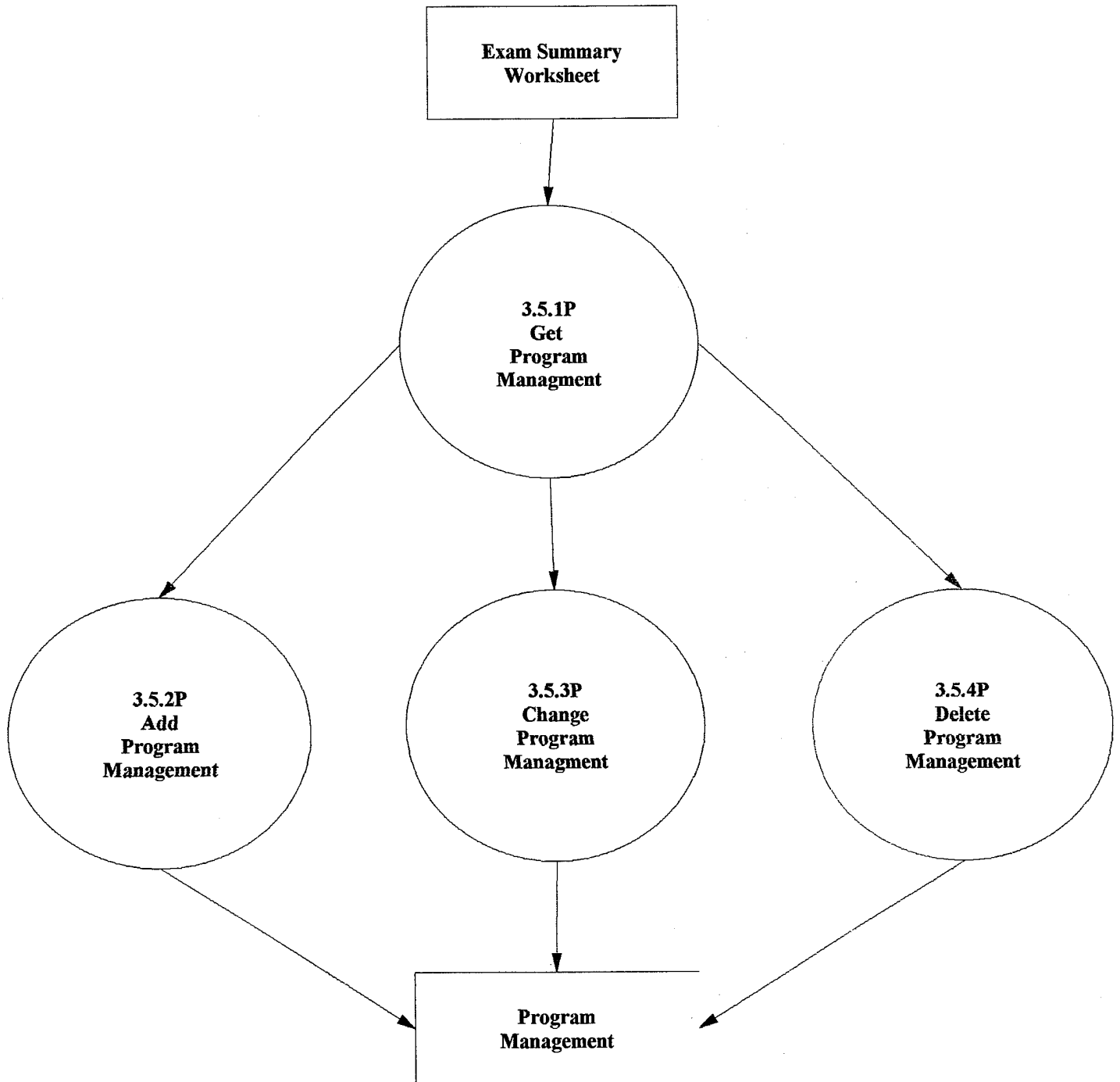


Figure 3.6
Diagram

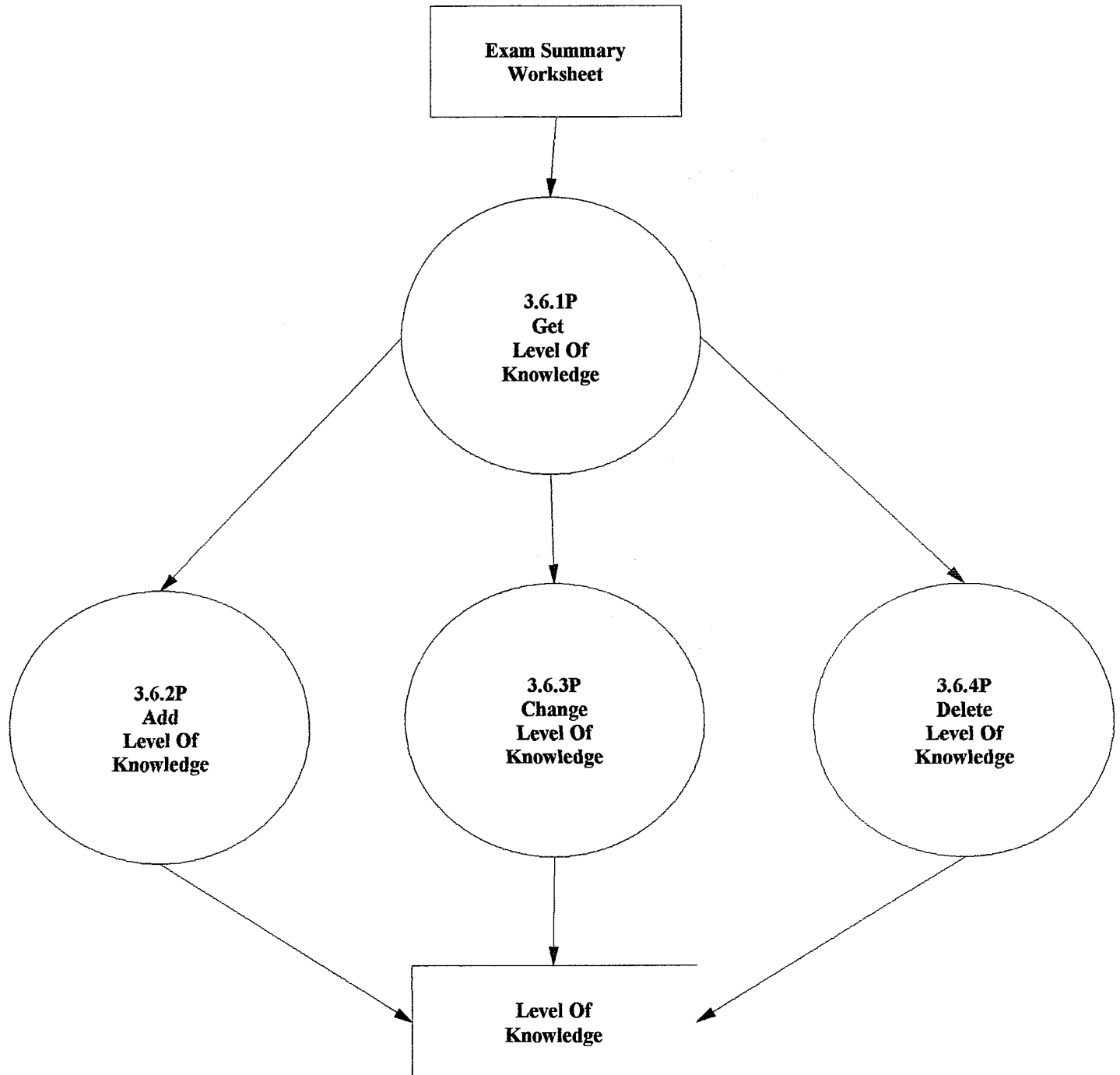


**Figure 3.7
Diagram**

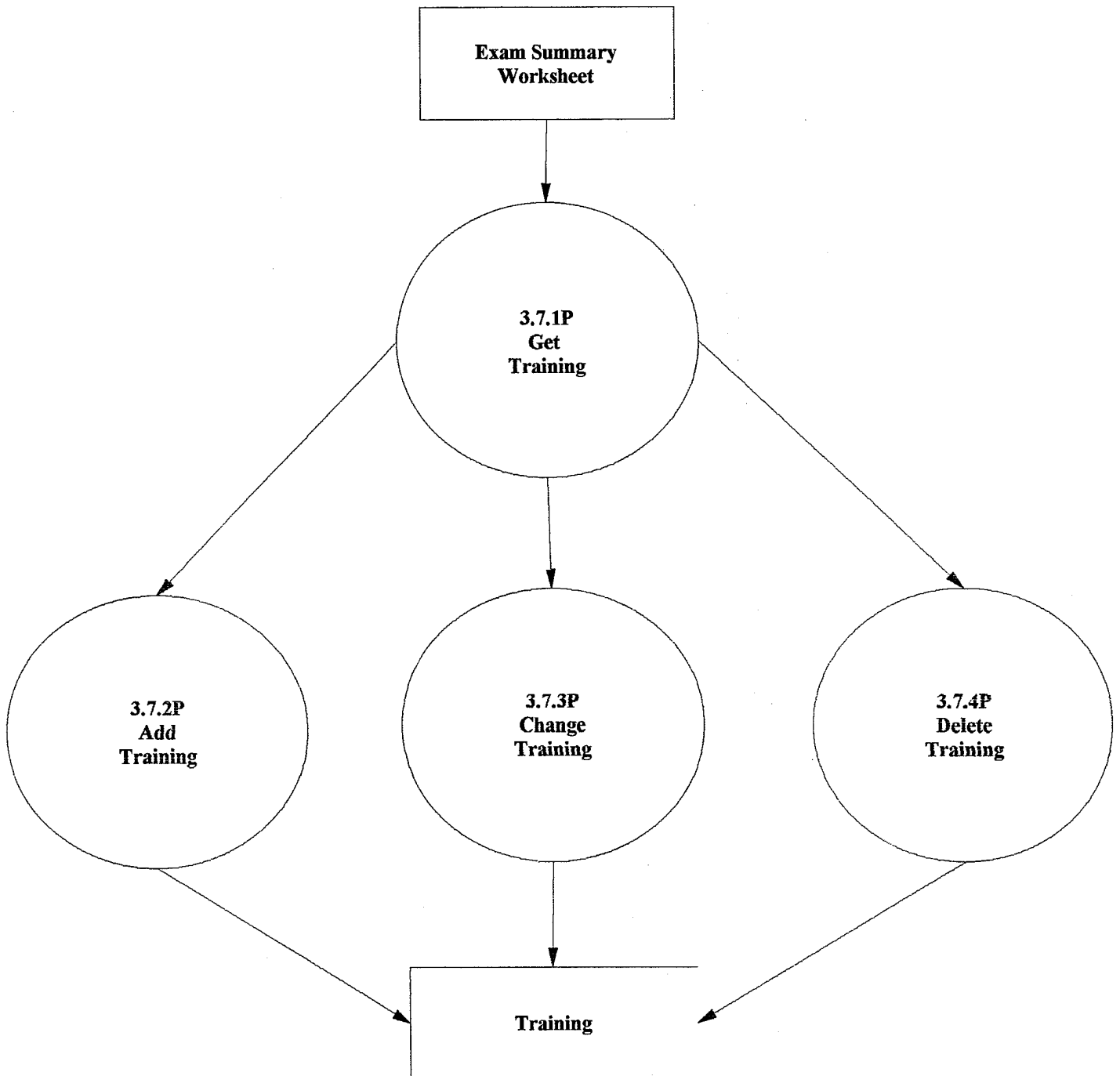
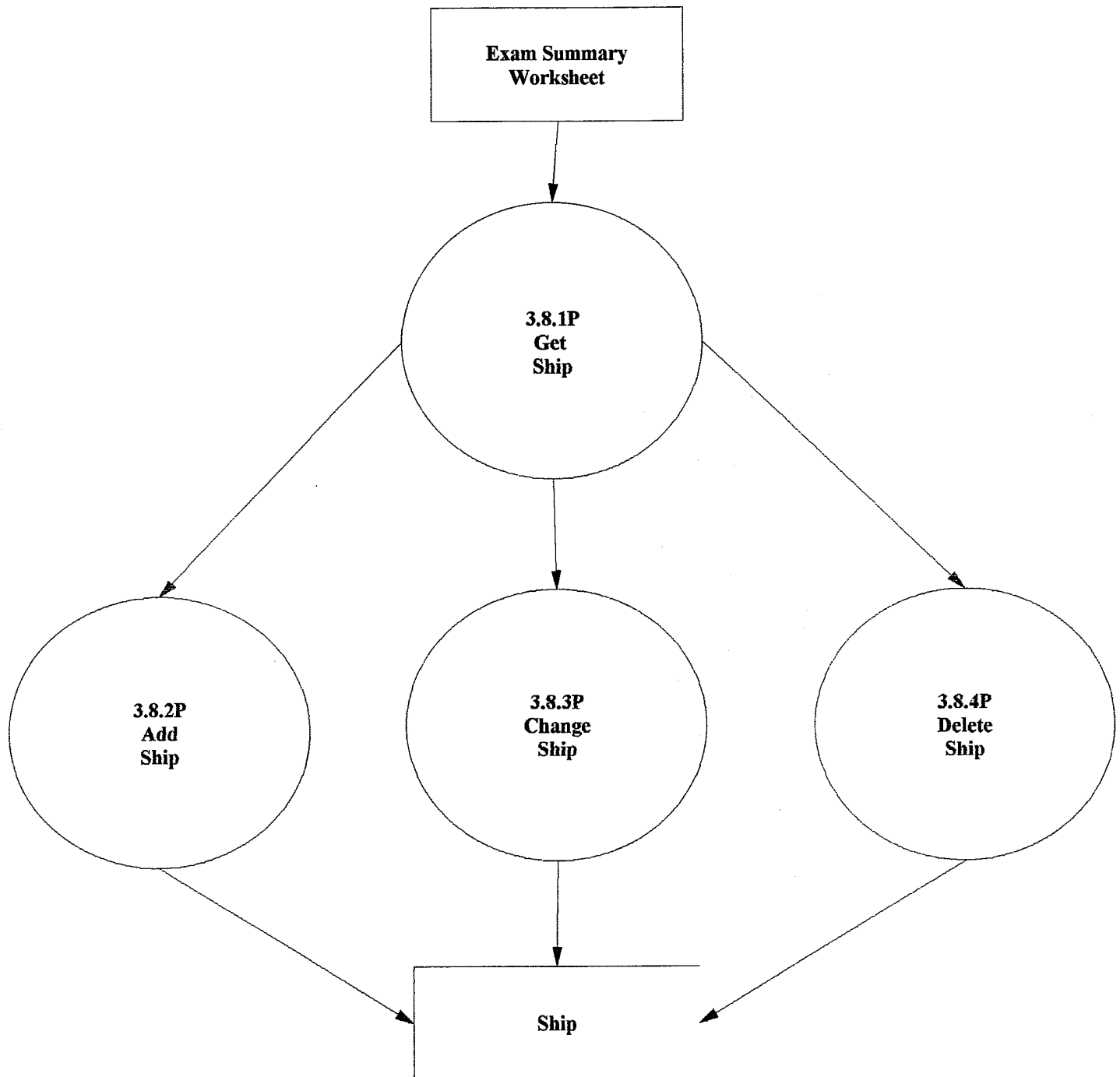


Figure 3.8
Diagram



APPENDIX D. PROCESS SPECIFICATIONS

A. REPORTS

1. Produce CNO Monthly Report (1.1P)

BEGIN

Run monthly summary of activity query

Display report

Print report

END

2. Produce CINCPACFLT Monthly Report (1.2P)

BEGIN

Run monthly summary of activity query

Run ships with special situations query

Run twelve month OPPE cumulative trend query

Display report

Print report

END

B. QUERIES

1. Produce 12 Month OPPE Cumulative Trend Query (2.1P)

BEGIN

n=1

FOR n=1 to 12

Get start date of period from user

Get end date of period from user

Percentage=number of satisfactory exams/total number of exams

FROM start date to end date

Get number of satisfactory exams

Get total number of exams

Display percentage

END

n=n+1

END LOOP

END

2. Produce Ship With Special Situations Query (2.2P)

BEGIN

Get start date of period from user

Get end date of period from user

FROM start date to end date

Get overall finding
 Get next exam date
 Get exam comments
 Get ship name
 Display over all finding
 Display next exam date
 Display exam comments
 Display ship name

END

END

3. Produce Monthly Summary of Activity Query (2.3P)

BEGIN

Get start date of period from user
 Get end date of period from user
 FROM start date to end date
 Get ship name
 Get overall finding
 Get exam comments
 Get exam end date
 Display ship name
 Display overall finding
 Display exam end date
 Display exam comments

END

END

4. Produce High Power Demo Query (2.4P)

BEGIN

Get start date of period from user
 Get end date of period from user
 FROM start date to end date
 Get high power demo grade
 Get high power demo comments
 Get ship name
 Get propulsion type
 Display high power demo grade
 Display high power demo comments
 Display ship name
 Display propulsion type

END

END

5. Produce Unsat Program Query (2.5P)

BEGIN

```

Get start date of period from user
Get end date of period from user
FROM start date to end date
    Get program grade
    Get exam type
    Get exam end date
    Get ship name
    Get propulsion type
    Display program grade
    Display exam type
    Display exam end date
    Display ship name
    Display propulsion type

```

END

END

6. Produce Pacific Fleet OPPE Summary Query (2.6P)

BEGIN

```

Get start date of period from user
Get end date of period from user
FROM start date to end date
    Get overall finding
    Get program grade
    Get level of knowledge grade
    Get material grade
    Get operations grade
    Get fire fighting grade
    Get ship name
    Get propulsion type
    Display overall finding grade
    Display program grade
    Display level of knowledge grade
    Display material grade
    Display operations grade
    Display fire fighting grade
    Display ship name
    Display propulsion type

```

END

END

7. Produce Task and Drill Summary Query (2.7P)

BEGIN

```

Get start date of period
end date of period
FROM start date to end date

```

Get number of drills
 Get number of tasks
 Get shipname
 Get propulsion type
 Display number of drills
 Display number of tasks
 Display shipname
 Display propulsion type

END

END

8. Produce Fire Fighting Query (2.8P)

BEGIN

Get start date of period
 Get end date of period
 FROM start date to end date
 Get fire fighting grade
 Get fire fighting comments
 Get propulsion type
 Get ship name
 Get exam type
 Display fire fighting grade
 Display fire fighting comments
 Display propulsion type
 Display ship name
 Display exam type

END

END

9. Produce Boiler Flex Query (2.9P)

BEGIN

Get start date of period
 Get end date of period
 FROM start date do end date
 Get shipname
 Get propulsion type
 Get flex levels
 Get total flexes
 Get flex comments
 Display shipname
 Display propulsion type
 Display flex levels
 Display total flexes
 Display flex comments

END

END

C. EXAM

1. Get Exam Update (3.1.1P)

BEGIN

Get user selection

Process user selection

END

2. Add Exam (3.1.2P)

BEGIN

Get new exam information

Store in Exam data store

END

3. Change Exam (3.1.3P)

BEGIN

Get desired exam information

Change exam information

Store in Exam data store

END

4. Delete Exam (3.1.4P)

BEGIN

Get desired exam information

Delete exam information

END

D. MATERIAL

1. Get Material Update (3.2.1P)

BEGIN

Get user selection

Process user selection

END

2. Add Material (3.2.2P)

BEGIN

Get new material information

Store in Material data store

END

3. Change Material (3.2.3P)

BEGIN

Get desired material information

Change material information

Store in Material data store

END

4. **Delete Material (3.2.4P)**
BEGIN
 Get desired material information
 Delete material information
END
- E. **FIRE FIGHTING**
 1. **Get Fire Fighting Update (3.3.1P)**
BEGIN
 Get user selection
 Process user selection
END
 2. **Add Fire Fighting (3.3.2P)**
BEGIN
 Get new fire fighting information
 Store in Fire Fighting data store
END
 3. **Change Fire Fighting (3.3.3P)**
BEGIN
 Get desired fire fighting information
 Change fire fighting information
 Store in Fire Fighting data store
END
 4. **Delete Fire Fighting (3.3.4P)**
BEGIN
 Get desired material information
 Delete material information
END
- F. **OPERATIONS**
 1. **Get Operations Update (3.4.1P)**
BEGIN
 Get user selection
 Process user selection
END
 2. **Add Operations (3.4.2P)**
BEGIN
 Get new operations information
 Store in Operations data store
END
 3. **Change Operations (3.4.3P)**
BEGIN
 Get desired operations information
 Change operations information
 Store in Operations data store

```

    END
4.  Delete Operations (3.4.4P)
    BEGIN
        Get desired operations information
        Delete operations information
    END
G.  PROGRAM MANAGEMENT
1.  Get Program Management Update (3.5.1P)
    BEGIN
        Get user selection
        Process user selection
    END
2.  Add Program Management (3.5.2P)
    BEGIN
        Get new program management information
        Store in Program Management data store
    END
3.  Change Program Management (3.5.3P)
    BEGIN
        Get desired program management information
        Change program management information
        Store in Program Management data store
    END
4.  Delete Program Management (3.5.4P)
    BEGIN
        Get desired program management information
        Delete program management information
    END
H.  LEVEL OF KNOWLEDGE
1.  Get Level of Knowledge Update (3.6.1P)
    BEGIN
        Get user selection
        Process user selection
    END
2.  Add Level of Knowledge (3.6.2P)
    BEGIN
        Get new program management information
        Store in Program Management data store
    END
3.  Change Level of Knowledge (3.6.3P)
    BEGIN
        Get desired level of knowledge information

```


Change level of knowledge information
Store in Level of Knowledge data store

END

4. Delete Level of Knowledge (3.6.4P)

BEGIN

Get desired level of knowledge information
Delete level of knowledge information

END

I. TRAINING

1. Get Training Update (3.7.1P)

BEGIN

Get user selection
Process user selection

END

2. Add Training (3.7.2P)

BEGIN

Get new training information
Store in Training data store

END

3. Change Training (3.7.3P)

BEGIN

Get desired training information
Change training information
Store in Training data store

END

4. Delete Training (3.7.4P)

BEGIN

Get desired training information
Delete training information

END

J. SHIP

1. Get Ship Update (3.8.1P)

BEGIN

Get user selection
Process user selection

END

2. Add Ship (3.8.2P)

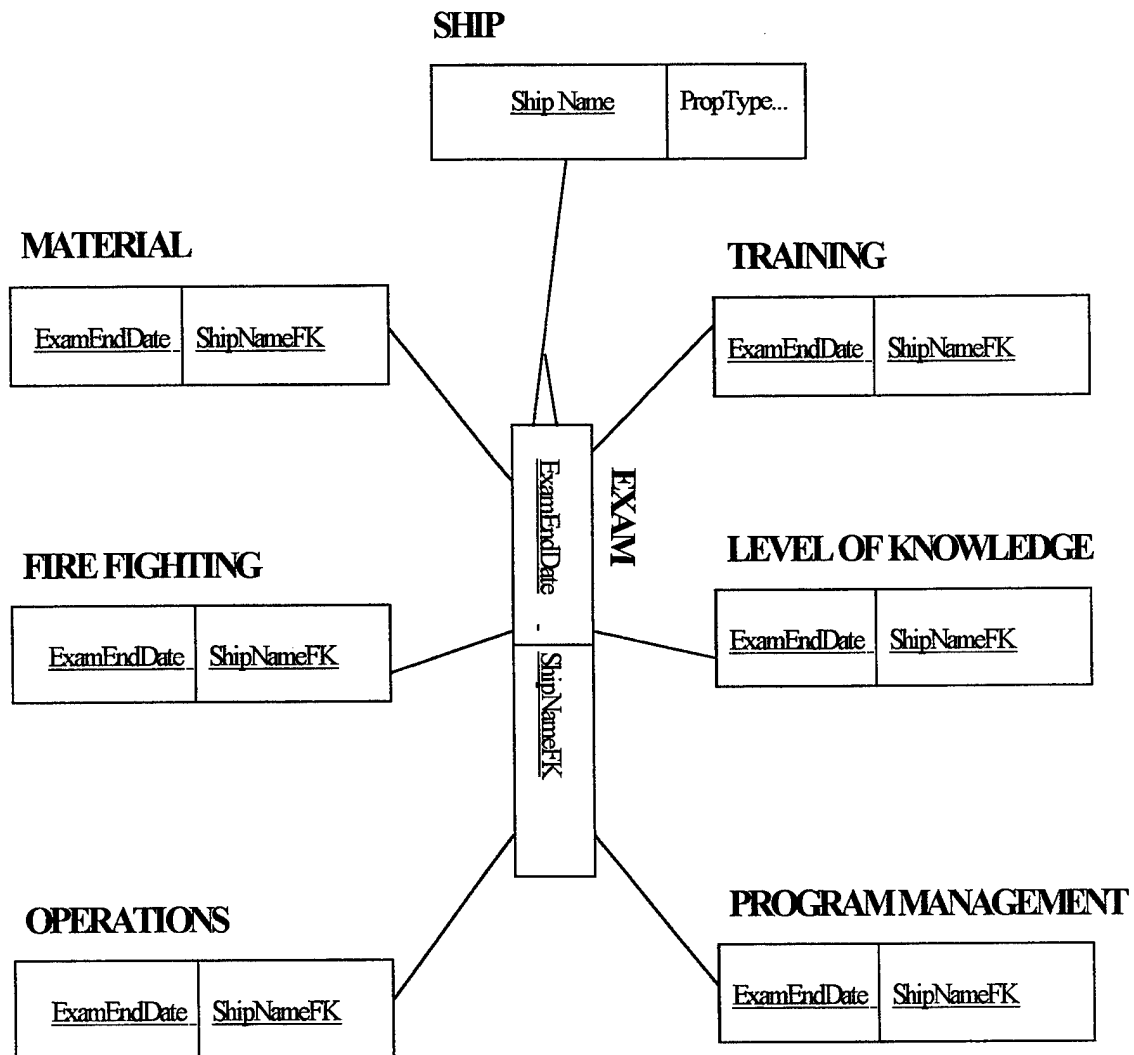
BEGIN

Get new ship information
Store in Ship data store

END

3. **Change Ship (3.8.3P)**
BEGIN
 Get desired ship information
 Change ship information
 Store in Ship data store
END
4. **Delete Ship (3.8.4P)**
BEGIN
 Get desired ship information
 Delete ship information
END

APPENDIX E. RELATIONAL DIAGRAM



APPENDIX F. INPUT AND QUERY FORMS

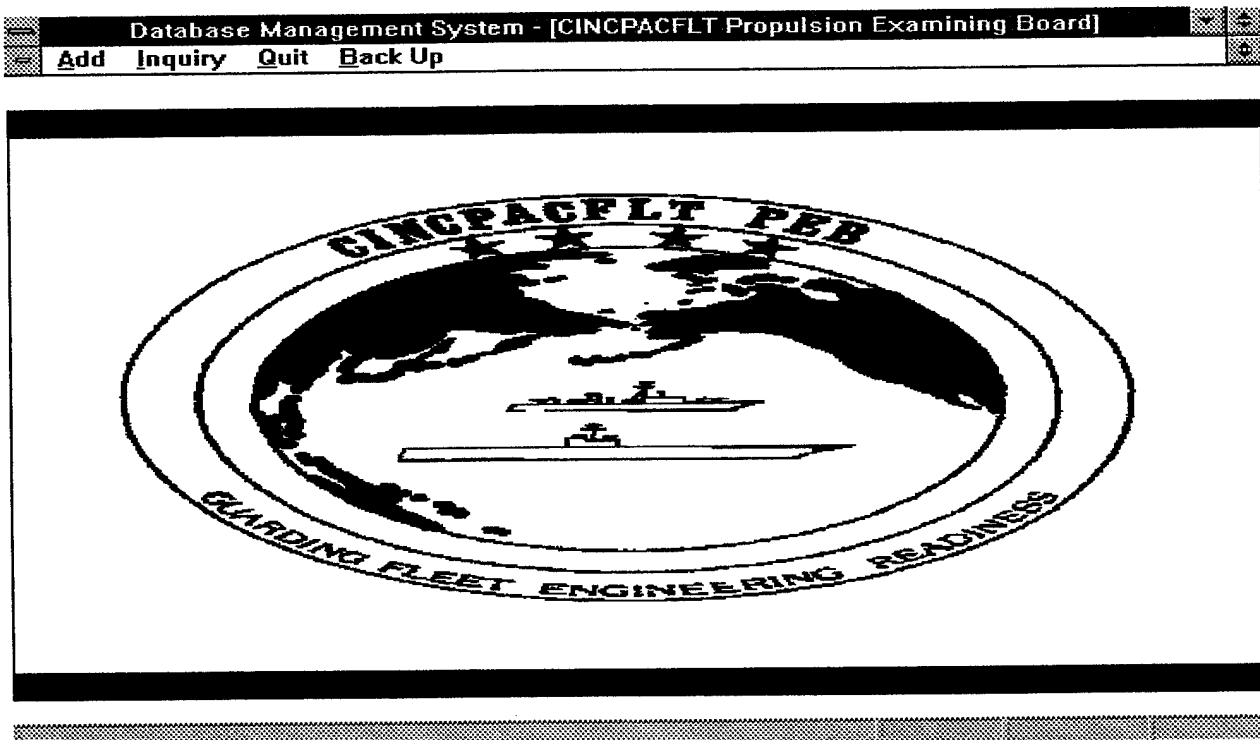


Figure [1] Main Menu

Database Management System - [Ship Input Form]

Record Quit Help

Ship

Ship Name	ENTER HERE
Hull Number	0054
PropType	GT
ISIC	CDS4
CO Name	ENCINIAS
XO Name	ENCINIAS
CHENG Name	ENCINIAS

INSERT NEW SHIP

1 of 5 [SHIP DB]

Figure [2] Ship Input Form

Database Management System - [Exam Input Form]	
Record Quit Help	
Exam	
Ship Name : ENIGLAND Hull Number : CG22 Prop Type : STM ISIC : CDSB CO Name : SMITH XO Name : JONES CHENG Name : JOHNSON	INSERT NEW EXAM ExamEndDate : 11/11/94 ExamType : OPPE Project Officer : SMITH Senior Examiner : JONES Adjective Grade : UNS Overall Finding : UNS Cleared : Y
PreviousExamType : OPPE PreviousExamGrade : SAT NextExamDate : 11/11/99	Comments :
1 of 7 [EXAM.DB] Edit	

Figure [3] Exam Input Form

Database Management System - [Fire Fighting Input Form]	
Record Quit Help	
Fire Fighting	
Ship Name : AMSTERAM HullNumber : CG54 PropType : GT ISIC : CDS4 CO Name : ENCINIAS XO Name : ENCINIAS CHENG Name : ENCINIAS	INSERT NEW F/F ExamEndDate : 11/11/95 MSFD Grade : SAT Space DC Equip Grade : SAT Halon Grade : SAT APFF Grade : SAT RepV Inventory Grade : SAT
Rep V Majors : 22 Rep V Minors : 22 DC Majors : 22 DC Minors : 22	DCTT Grade : SAT DCTT : GOOD Comments :
1 of 4 [FIREFIGHT.DB] Edit	

Figure [4] Fire Fighting Input Form

Database Management System - [Material Input Form]	
Record Quit Help	
Material	
Ship Name : ANTHETAM	Exam End Date : 11/11/94
Hull Number : CG54	Total Number Of Majors : 44
PropType : GT	Total Number Of Minors : 44
ISIC : CDS4	Ship Reported Degradations : 44
CO Name : ENCINIAS	PEB ID Degradations : 44
XO Name : ENCINIAS	Ship Reported OOC : 44
CHENG Name : ENCINIAS	PEB ID OOC : 44
Valve Maintenance Grade : SAT	Standard Equipment Satisfied? : Y
CSMP Grade : SAT	Total Number Of Level 1 Flex : 0
Gage Cal Grade : SAT	Total Number Of Level 2 Flex : 0
Material Self Assess Grade : SAT	Total Number Of Level 3 Flex : 0
Preservation Grade : SAT	Total Number Of Level 4 Flex : 0
1 of 5 [MATERIAL.DB] Edit	

Figure [5] Material Input Form

Database Management System - [Operations Input Form]	
Record Quit Help	
Operations	
Ship Name : ANTHETAM	Exam End Date : 11/11/94
Hull Number : CG54	% of Sat Evolutions 1st Sec : 45.00
PropType : GT	% of Sat Evolutions 2nd Sec : 55.00
ISIC : CDS4	% of Sat Evolutions 3rd Sec : 30.00
CO Name : ENCINIAS	% of Sat Drills 1st Sec : 45.00
XO Name : ENCINIAS	% of Sat Drills 2nd Sec : 55.00
CHENG Name : ENCINIAS	% of Sat Drills 3rd Sec : 30.00
	Number Of Sat Watch Sections : 1
Eccit Grade : SAT	Operation Grade : UNS
Eccit Comments :	Operation Comments :
1 of 2 [OPERATIO.DB] Edit	

Figure [6] Material Input Form

Database Management System - [Program Management Input Form]		
Record Quit Help		
Program Managment		
Ship Name: ANTIFAM	Exam End Date: 11/11/94	
Hull Number: CG54	BWFW Grade: UNS	LOGM Grade: SAT
Prop Type: GT	DJWTT Grade: NA	FOGM Grade: SAT
ISIC: CDS4	Operation Logs Grade: SAT	DETA Grade: NA
CO Name: ENCINIAS	MGTESR Grade: UNS	Legal Records Grade: SAT
XO Name: ENCINIAS	Tagout Grade: SAT	Bearing Records Grade: SAT
CHENG Name: ENCINIAS	Electrical Safety Grade: SAT	Hearing Conservation Grade: SAT
	QA Grade: SAT	OLV Grade: NA
Program Management Grade: SAT		
Program Comments:		
Record deleted Edit		

Figure [7] Program Management Input Form

Database Management System - [Training Input Form]	
Record Quit Help	
Training	
Ship Name: ANTIFAM	Exam End Date: 11/11/94
Hull Number: CG54	PGS Grade: SAT
Prop Type: GT	Training Grade: SAT
ISIC: CDS4	Number Of Sat Eopw: 4
CO Name: ENCINIAS	Number Of Sat Oil King: 4
XO Name: ENCINIAS	Number Of Sat ENOW: 0
CHENG Name: ENCINIAS	Number Of Sat Generator Operator: 0
Number Of Sat Boiler Operator: 3	Number Of Sat Messenger/Eng Op: 0
Number Of Sat BTCW/Console Operator: 3	
Number Of Sat MMOW: 3	
Number Of Sat EDO/SWBDOperator: 0	
1 of 2 [TRAINING.DB] Edit Locked	

Figure [8] Training Input Form

Database Management System - [Level Of Knowledge Input Form]

Record Quit Help

Level Of Knowledge

Ship Name: ANIE731 Hull Number: CG54 Prop Type: GT INSERT NEW LOK ISIC: CDS4 CO Name: ENCINAS XO Name: ENCINAS CHENG Name: ENCINAS LOK Comments: GOOD Program...	Exam End Date: 11/11/94 % Pass Written Exam: 66.00 Avg Score Written: 66.00 % Pass DC Exam: 66.00 Avg Score DC: 66.00 % Pass Sup Exam: 66.00 Avg Score Sup: 66.00 Avg Score EM Sup: 66.00

1 of 3 [LEVELOK.DB] Edt

Figure [9] Level of Knowledge Input Form

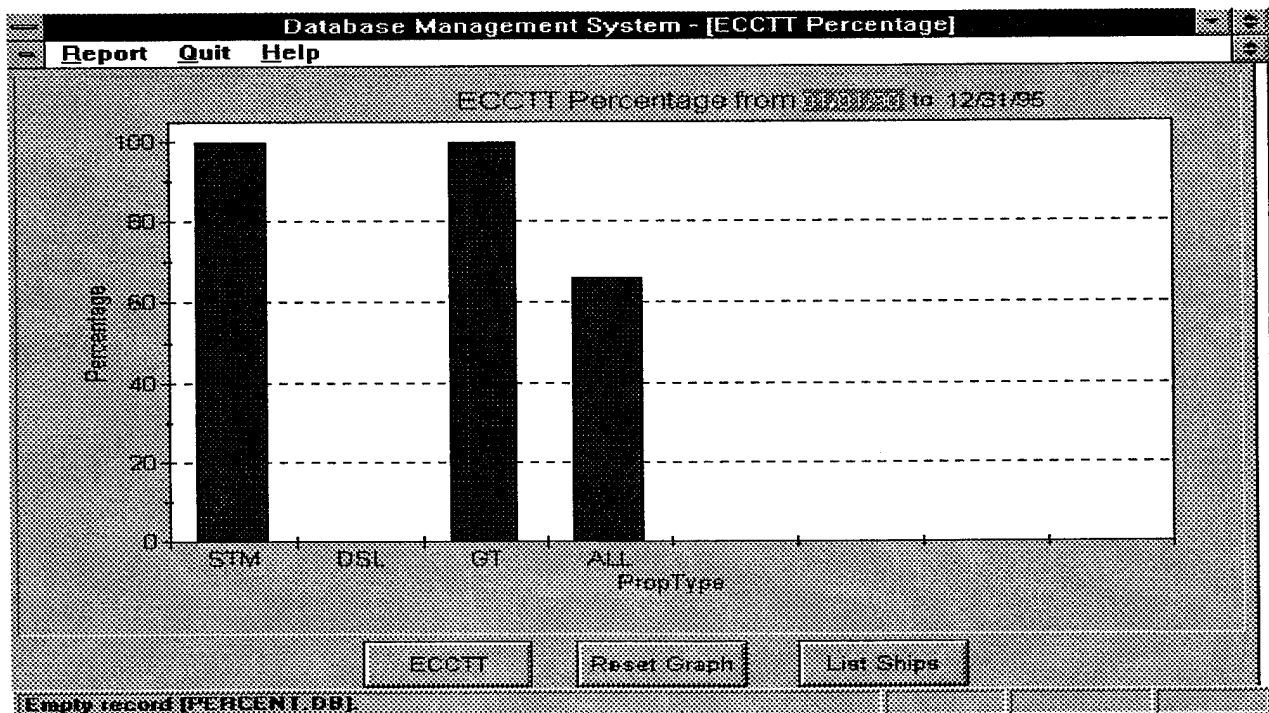


Figure [10] ECCTT Query

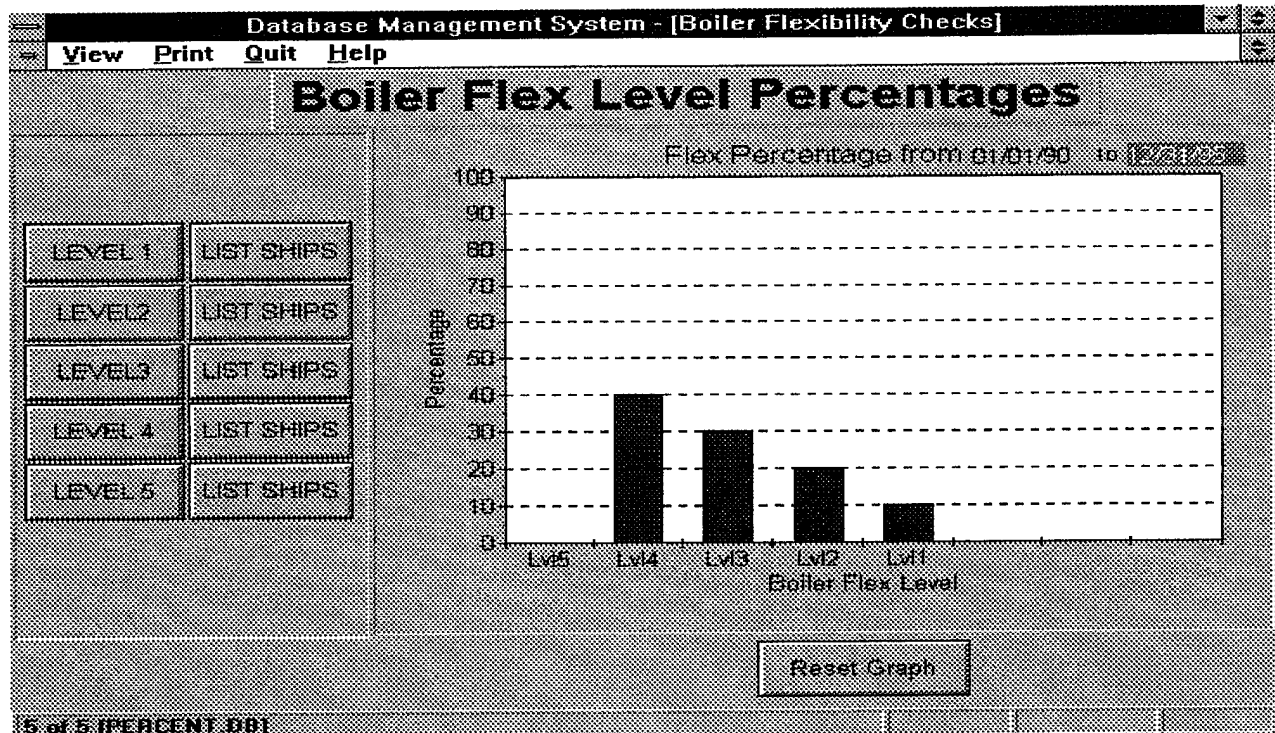


Figure [11] Boiler Flex Query

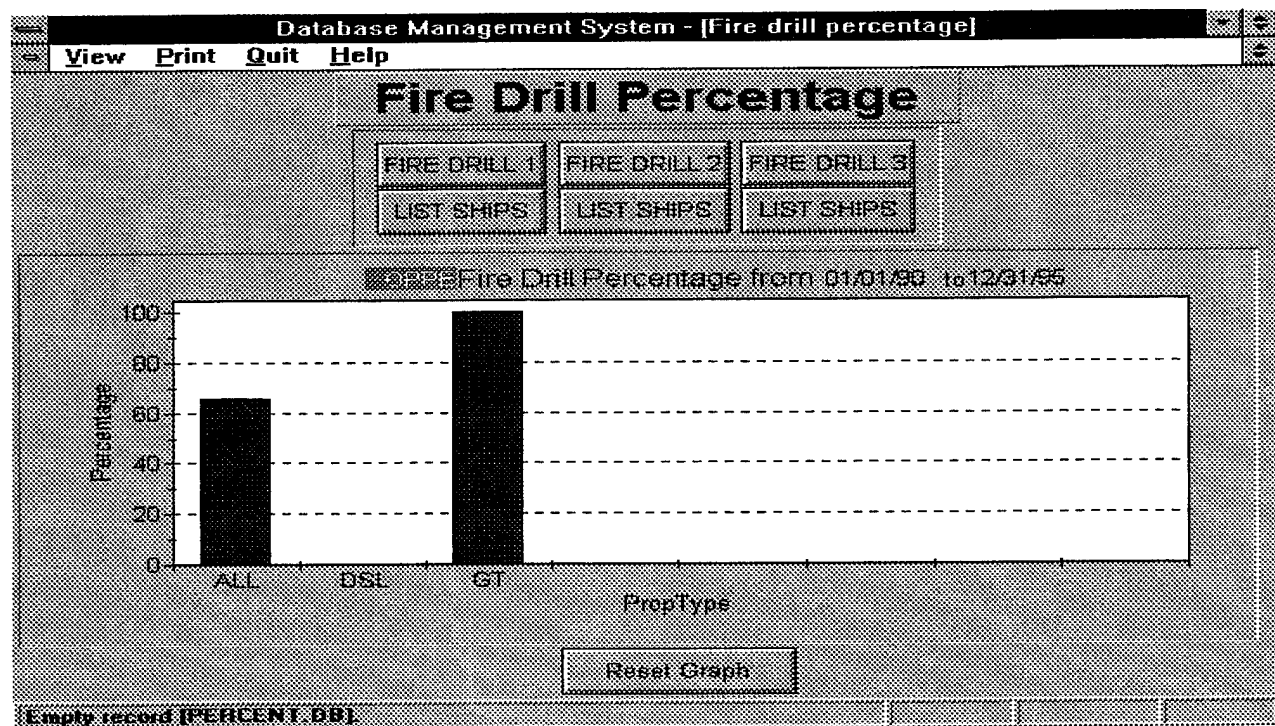


Figure [12] Fire Drill Percentage Query

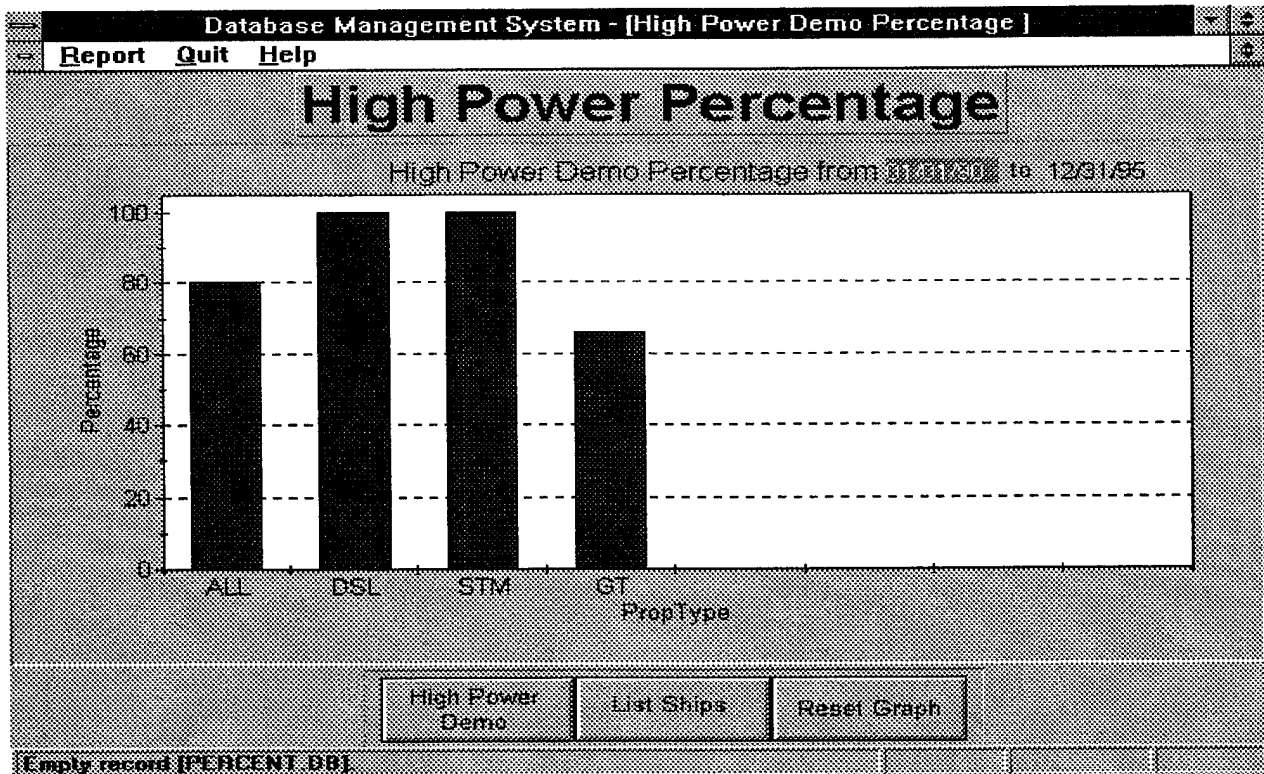


Figure [13] High Power Demo Percentage Query

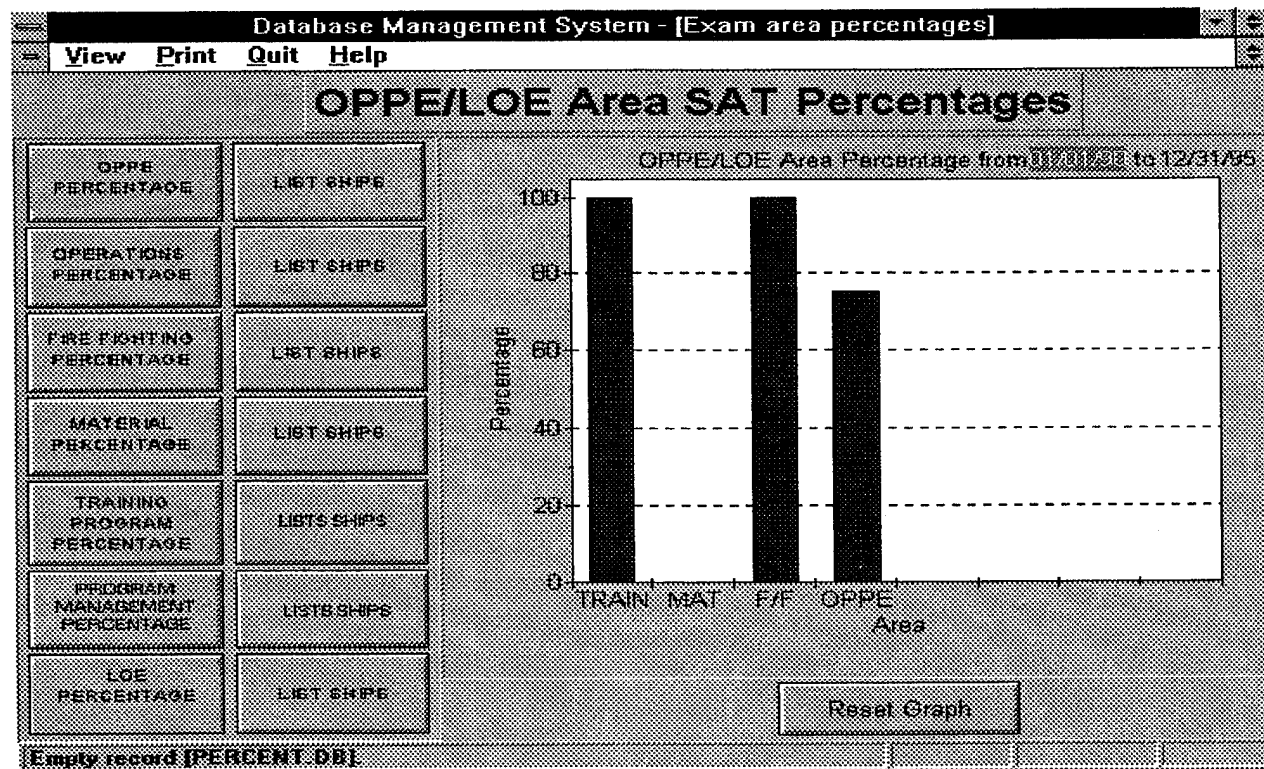


Figure [14] OPPE/LOE Area Percentage Query

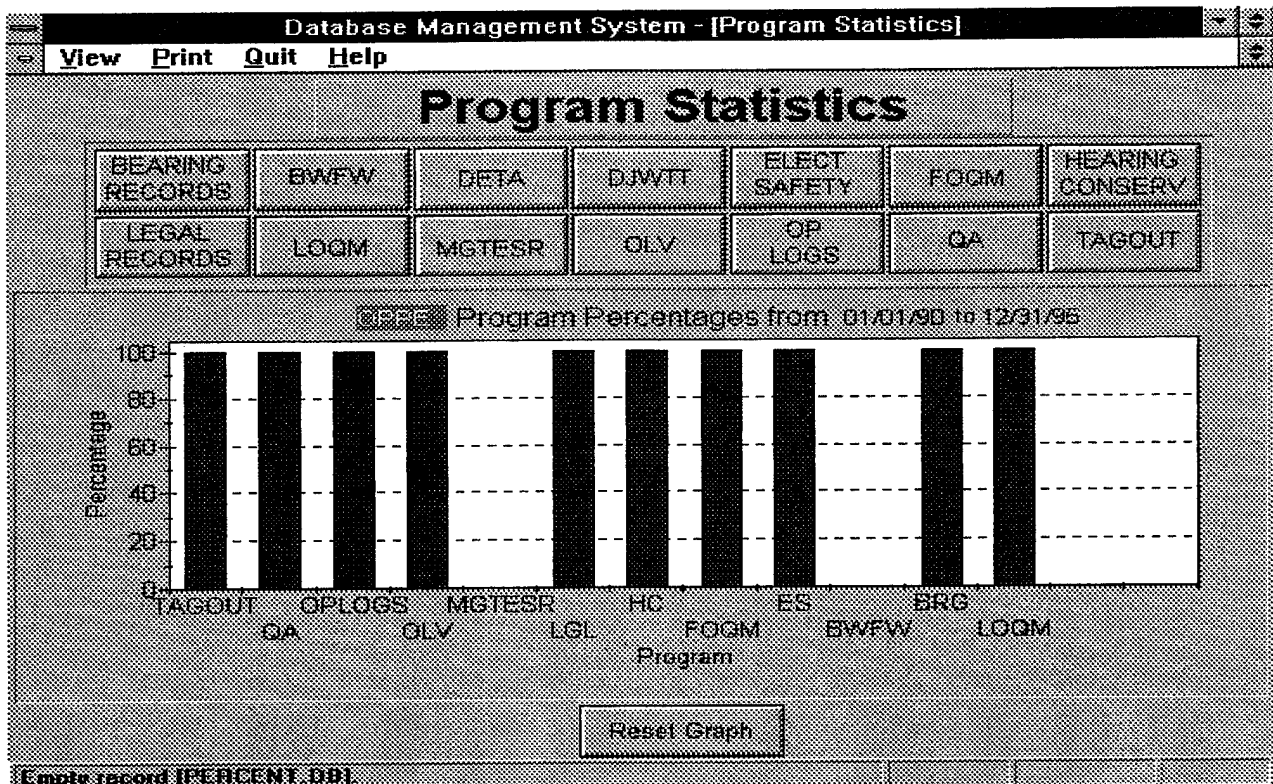


Figure [15] Program Statistics Query

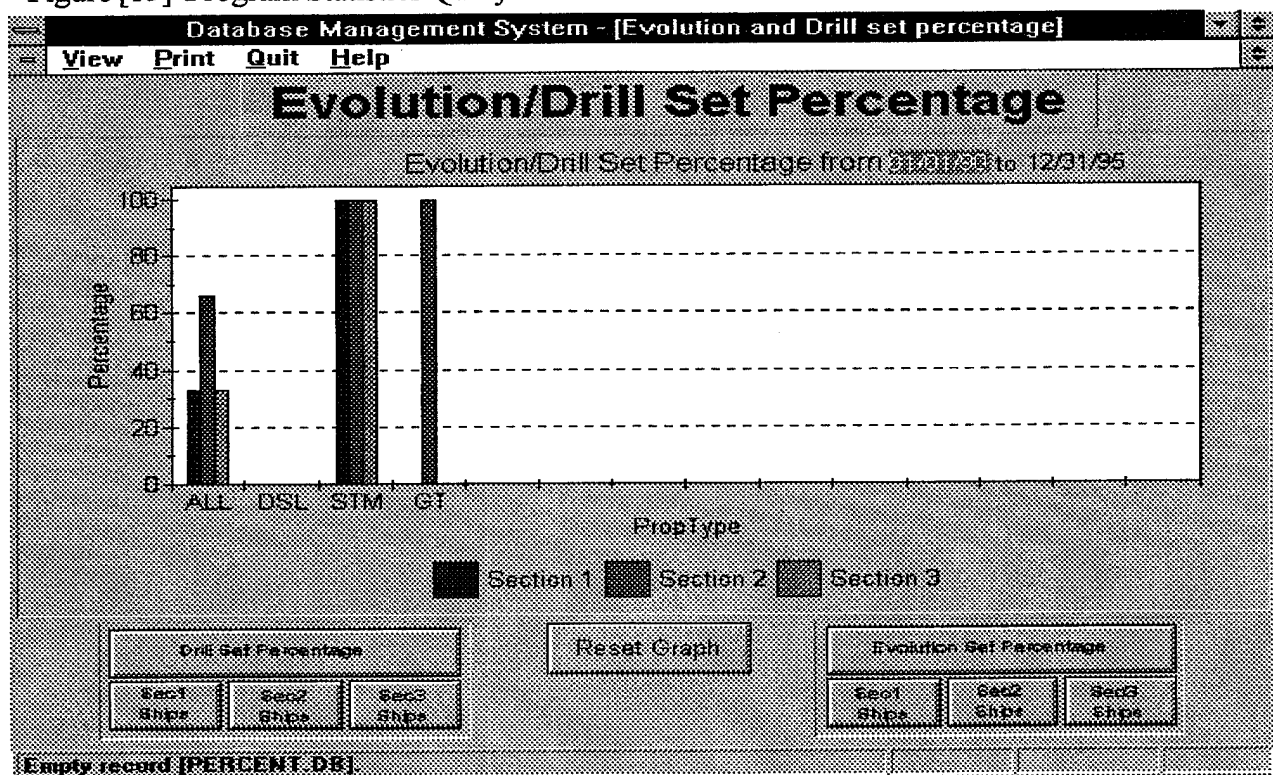


Figure [16] Evolution/Drill Set Percentage Query

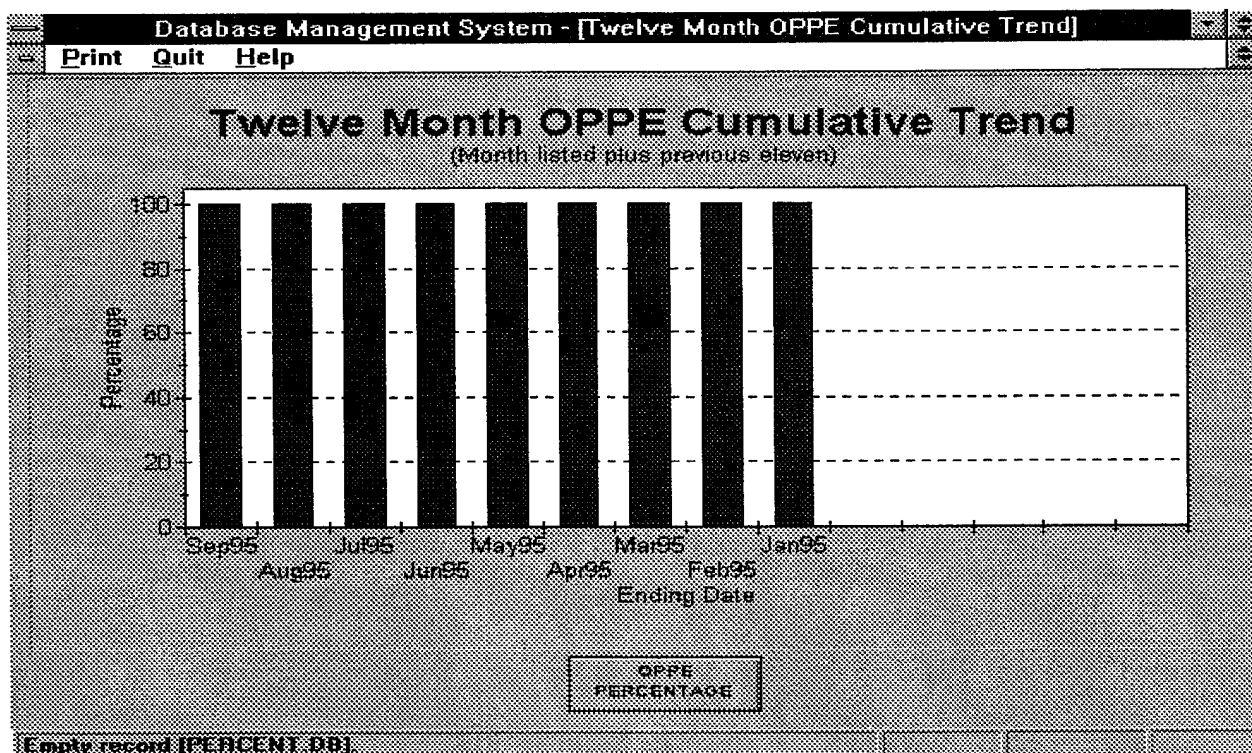


Figure [17] Twelve Month OPPE Cumulative Trend Query

Database Management System - [Monthly summary]

Quit Help

OPPE/LOE Monthly Summary Of Activity

SUMMARY QUERY

VIEW REPORT

PRINT REPORT

Press F8 to end edit mode.

Edit

Figure [18] OPPE/LOE Monthly Summary of Activity Query

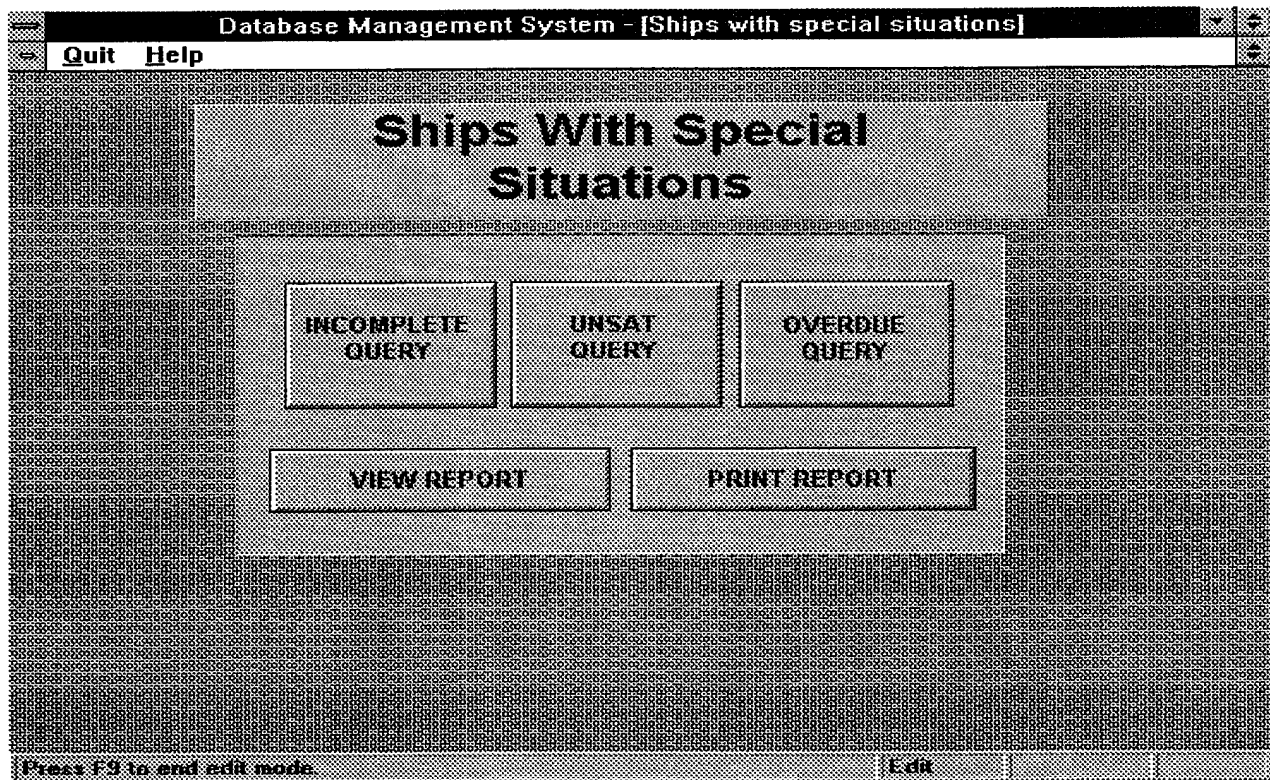


Figure [19] Ships With Special Situations Query

APPENDIX G. OBJECTPAL SOURCE CODE

Object : MENU

MethodName : open

```
Source :      method open(var eventInfo Event)
              if eventInfo.isPreFilter()
                  then
                      ; This code executes for each object on the form.

                  else
                      ; This code executes only for the form.
              hide()
              hideSpeedBar()
              endif
              endmethod
```

Object : MENU

MethodName : arrive

```
Source :      method arrive(var eventInfo MoveEvent)
              var
              examMenu Menu
              AddPop PopUpMenu
              InquiryPop PopUpMenu
              endVar
              if eventInfo.isPreFilter()
                  then
                      ; This code executes for each object on the form.

                  else
                      ; This code executes only for the form.
```

```
AddPop.addText("&Ship")
AddPop.addText("&Exam")
AddPop.addText("&Fire Fighting")
AddPop.addText("&Level Of Knowledge")
AddPop.addText("&Material")
AddPop.addText("&Operations")
AddPop.addText("&Program Management")
AddPop.addText("&Training")
```

```
InquiryPop.addText("&Boiler Flexes")
InquiryPop.addText("&ECCTT")
InquiryPop.addText("&Fire Drills")
InquiryPop.addText("&High Power Demos")
```



```

InquiryPop.addText("&OPPE/LOE Area Summary")
InquiryPop.addText("&Program Statistics")
InquiryPop.addText("&Monthly OPPE/LOE Summary")
InquiryPop.addText("&Evolutions and Drills")
InquiryPop.addText("&Ships With Special Situations")
InquiryPop.addText("&Twelve Month OPPE Cumulative Trend")

```

```

examMenu.addPopUp("&Add", AddPop)
examMenu.addPopUp("&Inquiry", InquiryPop)
examMenu.addText("&Quit")
examMenu.addText("&Back Up")
maximize()
examMenu.show()
endif
endmethod

```

Object : MENU

MethodName : depart

```

Source : method depart(var eventInfo MoveEvent)
var
form1 form
endVar
if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else
; This code executes only for the form.
;exit()
endif
endmethod

```

Object : MENU

MethodName : menuAction

```

Source : method menuAction(var eventInfo MenuEvent)
var
form1 form
form2 form
form3 form
form4 form
form5 form
form6 form
form7 form

```

```

reply string
endVar
if eventInfo.isPreFilter()
    then
        ; This code executes for each object on the form.

    else
        ; This code executes only for the form.
Switch
    case eventInfo.menuChoice() = "&Exam":
        form1.open("Exam")
    case eventInfo.menuChoice() = "&Ship":
        form1.open("Ship")
    case eventInfo.menuChoice() = "&Fire Fighting":
        form2.open("FireFigh")
    case eventInfo.menuChoice() = "&Level Of Knowledge":
        form3.open("LOK")
    case eventInfo.menuChoice() = "&Material":
        form4.open("Material")
    case eventInfo.menuChoice() = "&Operations":
        form5.open("Operaton")
    case eventInfo.menuChoice() = "&Program Management":
        form6.open("ProgMan")
    case eventInfo.menuChoice() = "&Training":
        form7.open("Train")
    case eventInfo.menuChoice() = "&Boiler Flexes":
        form1.open("BoilFlex")
    case eventInfo.menuChoice() = "&Fire Drills":
        form2.open("FireDril")
    case eventInfo.menuChoice() = "&High Power Demos":
        form3.open("HighPwr")
    case eventInfo.menuChoice() = "&OPPE/LOE Area Summary":
        form4.open("OPPESum")
    case eventInfo.menuChoice() = "&Program Statistics":
        form5.open("ProgStat")
    case eventInfo.menuChoice() = "&Monthly OPPE/LOE Summary":
        form6.open("Summary")
    case eventInfo.menuChoice() = "&Evolutions and Drills":
        form7.open("TaskDril")
    case eventInfo.menuChoice() = "&ECCTT":
        form6.open("ECCTTAVG")
    case eventInfo.menuChoice() = "&Ships With Special Situations":
        form1.open("Unsat")
    case eventInfo.menuChoice() = "&Twelve Month OPPE Cumulative Trend":
        form3.open("CumTrend")
    case eventInfo.menuChoice() = "&Quit":
        reply=msgQuestion("Quit", "Are you sure you want to leave the CINCPACFLT PEB Database?")
        If reply = "Yes" then

```

```

        close()
    else
        return
    endif
    case eventInfo.menuChoice() = "&Back Up":
        execute("mwbackup")
    endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

```

Source : method setFocus(var eventInfo Event)
var
    examMenu Menu
    AddPop PopUpMenu
    InquiryPop PopUpMenu
endVar
AddPop.addText("&Ship")
AddPop.addText("&Exam")
AddPop.addText("&Fire Fighting")
AddPop.addText("&Level Of Knowledge")
AddPop.addText("&Material")
AddPop.addText("&Operations")
AddPop.addText("&Program Management")
AddPop.addText("&Training")

InquiryPop.addText("&Boiler Flexes")
InquiryPop.addText("&ECCTT")
InquiryPop.addText("&Fire Drills")
InquiryPop.addText("&High Power Demos")
InquiryPop.addText("&OPPE/LOE Area Summary")
InquiryPop.addText("&Program Statistics")
InquiryPop.addText("&Monthly OPPE/LOE Summary")
InquiryPop.addText("&Evolutions and Drills")
InquiryPop.addText("&Ships With Special Situations")
InquiryPop.addText("&Twelve Month OPPE Cumulative Trend")

examMenu.addPopUp("&Add", AddPop)
examMenu.addPopUp("&Inquiry", InquiryPop)
examMenu.addText("&Quit")
examMenu.addText("&Back Up")
maximize()
examMenu.show()
endmethod

```

Object : SHIP_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    AddPop.addText("&Locate")
    AddPop.addText("&Delete")
    examMenu.addPopUp("&Record", AddPop)
    examMenu.addText("&Quit")
    examMenu.addText("&Help")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    action(DataInsertRecord)
  endif
endmethod
```

Object : SHIP_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
      case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete "+ShipName+"?") then
          deleteRecord()
        endif
      case eventInfo.menuChoice() = "&Help":
```

```

        case eventInfo.menuChoice() = "&Quit":
            reply=msgQuestion("Quit","Are you sure you want to leave this form?")
            If reply = "Yes" then
                close()
            else
                return
            endif
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box27.INSERT_NEW_SHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
 action(dataInsertRecord)
 endmethod

Object : #Page2.#Box27.PropType

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
 var
 tempValue anyType
 choice1 String
 choice2 String
 choice3 String
 choice4 String
 choice5 String
 endVar
 choice1="GT"
 choice2="DSL"
 choice3="STM"
 tempValue=self.value
 doDefault
 if eventInfo.errorCode()=0 then
 switch
 case eventInfo.newValue()=choice1:
 case eventInfo.newValue()=choice2:
 case eventInfo.newValue()=choice3:
 otherwise:
 self.value=tempValue
 msgStop("Problem","Value must be either GT, DSL, or STM")
 eventInfo.setErrorCode(-1)
 endSwitch
 endif
 endmethod

Object : EXAM_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

  AddPop.addText("&Locate")
  AddPop.addText("&Delete")
  examMenu.addPopUp("&Record", AddPop)
  examMenu.addText("&Quit")
  examMenu.addText("&Help")
  examMenu.show()
  maximize()
  hideSpeedBar()
  edit()
  action(dataInsertRecord)
endif
endmethod
```

Object : EXAM_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

  Switch
    case eventInfo.menuChoice() = "&Locate":
      action(DataSearch)
    case eventInfo.menuChoice() = "&Delete":
      if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
"+ExamID_Ship_ShipName_FK2+"?") then
        deleteRecord()
      endif
    case eventInfo.menuChoice() = "&Help":
```

```

case eventInfo.menuChoice() = "&Quit":
    reply=msgQuestion("Quit","Are you sure you want to leave this form?")
    If reply = "Yes" then
        close()
    else
        return
    endif
endSwitch
endif
endmethod

```

Object : #Page2.#Box40.Cleared

MethodName : arrive

```

Source : method arrive(var eventInfo MoveEvent)
if self.isBlank() then
    switch
        case overallFinding="UNS":
            self.value="N"
        case overallFinding="DEC":
            self.value="N"
        case overallFinding="INC":
            self.value="N"
        otherwise:
            self.value="Y"
    endSwitch
endif
endmethod

```

Object : #Page2.#Box40.AdjectiveGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
endVar
choice1="EXC"
choice2="GOOD"
choice3="SAT"
choice4="SAT(C)"
choice5="UNS"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:

```

```

case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:
    self.value=tempValue
    msgStop("Problem","Value must be either EXC, GOOD, SAT , SAT(C), UNS")
    eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box40.OverallFinding

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
    TC TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="DEC"
choice4="INC"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either UNS, SAT, DEC or INC")
            eventInfo.setErrorCode(-1)
        endSwitch
    endSwitch
endif
endmethod

```

Object : #Page2.#Box40.ExamType

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String

```



```

choice2 String
choice3 String
choice4 String
choice5 String
endVar

choice1="OPPE"
choice2="REOPPE"
choice3="LOE"
choice4="RELOE"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
otherwise:
self.value=tempValue
msgStop("Problem","Value must be either OPPE, REOPPE, LOE, or RELOE")
eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box40.ExamID_ExamEndDate

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
tempValue AnyType
endVar

tempValue=Self.value
doDefault
if eventInfo.errorCode()=0 then
if eventInfo.newValue() > today() then
Self.value=tempValue
msgStop("Problem","ExamEndDate cannot be a future date!")
eventInfo.setErrorCode(-1)
endif
endif
endmethod

```

Object : #Page2.#Box39.INSERT_NEW_EXAM_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
action(dataInsertRecord)
endmethod

```

Object : #Page2.#Box39.HullNumber

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
 disableDefault
 moveTo("ExamID_ExamEndDate")
 endmethod

Object : #Page2.#Box39.PropType

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
var
 tempValue anyType
 choice1 String
 choice2 String
 choice3 String
 choice4 String
 choice5 String
endVar
choice1="GT"
choice2="DSL"
choice3="STM"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
 switch
 case eventInfo.newValue()=choice1:
 case eventInfo.newValue()=choice2:
 case eventInfo.newValue()=choice3:
 otherwise:
 self.value=tempValue
 msgStop("Problem", "Value must be either GT, DSL, or STM")
 eventInfo.setErrorCode(-1)
 endSwitch
 endif
endmethod

Object : #Page2.#Box39.ExamID_Ship_ShipName_FK2

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
doDefault
if eventInfo.errorCode() = 0 then
 ;input accepted by Paradox
else
 msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")
endif
endmethod

Object : FIREFIGHTING_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    AddPop.addText("&Locate")
    AddPop.addText("&Delete")
    examMenu.addPopUp("&Record", AddPop)
    examMenu.addText("&Quit")
    examMenu.addText("&Help")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    action(DataInsertRecord)
  endif
endmethod
```

Object : FIREFIGHTING_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
      case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
"+FireFig_Ship_ShipName_FK5+"?") then
          deleteRecord()
        endif
      case eventInfo.menuChoice() = "&Help":
```

```

        case eventInfo.menuChoice() = "&Quit":
            reply=msgQuestion("Quit","Are you sure you want to leave this form?")
            if reply = "Yes" then
                close()
            else
                return
            endif
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box83.FireFightingGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box81.FireDrill3_Grade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box81.FireDrill2_Grade

MethodName : changeValue

Source :

```

method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault

```

```

if eventInfo.errorCode()=0 then
  switch
  case eventInfo.newValue()=choice1:
  case eventInfo.newValue()=choice2:
  case eventInfo.newValue()=choice3:
  case eventInfo.newValue()=choice4:
  case eventInfo.newValue()=choice5:
  otherwise:
    self.value=tempValue
    msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
    eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box81.FireDrill1_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
  case eventInfo.newValue()=choice1:
  case eventInfo.newValue()=choice2:
  case eventInfo.newValue()=choice3:
  case eventInfo.newValue()=choice4:
  case eventInfo.newValue()=choice5:
  otherwise:
    self.value=tempValue
    msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
    eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box82.DCTT_Grade

MethodName : changeValue

```
Source :      method changeValue(var eventInfo ValueEvent)
              var
                tempValue anyType
                choice1 String
                choice2 String
                choice3 String
                choice4 String
                choice5 String
              endVar

              choice1="SAT"
              choice2="UNS"
              choice3="NA"
              choice4="GOOD"
              choice5="EXC"

              tempValue=self.value
              doDefault
              if eventInfo.errorCode()=0 then
                switch
                  case eventInfo.newValue()=choice1:
                  case eventInfo.newValue()=choice2:
                  case eventInfo.newValue()=choice3:
                  case eventInfo.newValue()=choice4:
                  case eventInfo.newValue()=choice5:
                  otherwise:
                    self.value=tempValue
                    msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
                    eventInfo.setErrorCode(-1)
                endSwitch
              endif
            endmethod
```

Object : #Page2.#Box79.SpaceDC_EquipGrade

MethodName : changeValue

```
Source :      method changeValue(var eventInfo ValueEvent)
              var
                tempValue anyType
                choice1 String
                choice2 String
                choice3 String
                choice4 String
                choice5 String
              endVar

              choice1="SAT"
              choice2="UNS"
              choice3="NA"
              choice4="GOOD"
              choice5="EXC"
```

```

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box79.AFFF_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```


Object : #Page2.#Box79.HalonGrade

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod
```

Object : #Page2.#Box79.MSFD_Grade

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
```

```
choice4="GOOD"
choice5="EXC"
```

```
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod
```

Object : #Page2.#Box79.RepV_InventoryGrade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar
```

```
choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"
```

```
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
```

```
endif  
endmethod
```

Object : #Page2.#Box79.FireFightingI_ExamEndDate

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)  
var  
tempValue AnyType  
endVar  
  
tempValue=Self.value  
doDefault  
if eventInfo.errorCode()=0 then  
if eventInfo.newValue() > today() then  
Self.value=tempValue  
msgStop("Problem","ExamEndDate cannot be a future date!")  
eventInfo.setErrorCode(-1)  
endif  
endif  
endmethod
```

Object : #Page2.#Box78.INSERT_NEW_FF_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)  
action(dataInsertRecord)  
endmethod
```

Object : #Page2.#Box78.HullNumber

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)  
disableDefault  
moveTo("FireFightingI_ExamEndDate")  
endmethod
```

Object : #Page2.#Box78.FireFig_Ship_ShipName_FK5

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)  
doDefault  
if eventInfo.errorCode() = 0 then  
;input accepted by Paradox  
else  
msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")  
endif  
endmethod
```

Object : LOK_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    AddPop.addText("&Locate")
    AddPop.addText("&Delete")
    examMenu.addPopUp("&Record", AddPop)
    examMenu.addText("&Quit")
    examMenu.addText("&Help")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    action(DataInsertRecord)
  endif
endmethod
```

Object : LOK_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
      case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
"+LOK_ID_Ship_ShipName_FK1+"?") then
          deleteRecord()
        endif
      case eventInfo.menuChoice() = "&Help":
```

```

        case eventInfo.menuChoice() = "&Quit":
            reply=msgQuestion("Quit","Are you sure you want to leave this form?")
            If reply = "Yes" then
                close()
            else
                return
            endif
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box13.LOK_ID_ExamEndDate

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
        var
            tempValue AnyType
        endVar

        tempValue=Self.value
        doDefault
        if eventInfo.errorCode()=0 then
            if eventInfo.newValue() > today() then
                Self.value=tempValue
                msgStop("Problem","ExamEndDate cannot be a future date!")
                eventInfo.setErrorCode(-1)
            endif
        endif
    endmethod

```

Object : #Page2.#Box12.LOK_ID_Ship_ShipName_FK1

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
        doDefault
        if eventInfo.errorCode() = 0 then
            ;input accepted by Paradox
        else
            msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")
        endif
    endmethod

```

Object : #Page2.#Box12.INSERT_NEW_LOK_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
        action(dataInsertRecord)
    endmethod

```

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
 disableDefault
 moveTo("LOK_ID_ExamEndDate")
 endmethod

Object : MATERIAL_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    AddPop.addText("&Locate")
    AddPop.addText("&Delete")
    examMenu.addPopUp("&Record", AddPop)
    examMenu.addText("&Quit")
    examMenu.addText("&Help")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    action(DataInsertRecord)
  endif
endmethod
```

Object : MATERIAL_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
      case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
"+MatID_Ship_ShipName_FK4+"?") then
          deleteRecord()
        endif
      case eventInfo.menuChoice() = "&Help":
```

```

        case eventInfo.menuChoice() = "&Quit":
            reply=msgQuestion("Quit","Are you sure you want to leave this form?")
            if reply = "Yes" then
                close()
            else
                return
            endif
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box109.MaterialGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box108.MatID_ExamEndDate

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)


```

var
  tempValue AnyType
endVar

tempValue=Self.value
doDefault
if eventInfo.errorCode()=0 then
  if eventInfo.newValue() > today() then
    Self.value=tempValue
    msgStop("Problem","ExamEndDate cannot be a future date!")
    eventInfo.setErrorCode(-1)
  endif
endif
endmethod

```

Object : #Page2.#Box108.StandEquipSatisfied_

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="Y"
choice2="N"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either Y or N")
      eventInfo.setErrorCode(-1)
    endSwitch
  endif
endmethod

```

Object : #Page2.#Box107.TtlNrOfBoilersFlexed

MethodName : arrive

```

Source : method arrive(var eventInfo MoveEvent)
  TtlNrOfBoilersFlexed.value=TtlNrOfLevel1Flex+TtlNrOfLevel2Flex+TtlNrOfLevel3Flex+
    TtlNrOfLevel4Flex+TtlNrOfLevel5Flex
endmethod

```

Object : #Page2.#Box106.MatSelfAssessGrade

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
    endSwitch
  endIf
endmethod
```

Object : #Page2.#Box106.IOP_Comments

MethodName : action

```
Source : method action(var eventInfo ActionEvent)
Switch
  case propType.value="GT":
    if eventInfo.id()=fieldForward then
      disableDefault
      materialGrade.moveTo()
    endIf
  case propType.value="DSL":
    if eventInfo.id()=fieldForward then
      disableDefault
      materialGrade.moveTo()
    endIf
endSwitch
```

endmethod

Object : #Page2.#Box106.PreservationGrade

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod
```

Object : #Page2.#Box106.HighPwrDemoGrade

MethodName : changeValue

```
Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
```

```

choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box106.CleanlinessGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")

```

```

        eventInfo.setErrorCode(-1)
    endSwitch
endIf
endmethod

```

Object : #Page2.#Box106.StowageGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
            eventInfo.setErrorCode(-1)
    endSwitch
endIf
endmethod

```

Object : #Page2.#Box106.GageCalGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String

```

```

endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem", "Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
    endSwitch
  endif
endmethod

```

Object : #Page2.#Box106.CSMP_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:

```

```

otherwise:
  self.value=tempValue
  msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
  eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box106.ValveMaintGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
  case eventInfo.newValue()=choice1:
  case eventInfo.newValue()=choice2:
  case eventInfo.newValue()=choice3:
  case eventInfo.newValue()=choice4:
  case eventInfo.newValue()=choice5:
  otherwise:
    self.value=tempValue
    msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
    eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box105.INSERT_NEW_MAT_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
  action(dataInsertRecord)
endmethod

```

Object : #Page2.#Box105.MatID_Ship_ShipName_FK4

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
doDefault
if eventInfo.errorCode() = 0 then
;input accepted by Paradox
else
msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")
endif
endmethod

Object : #Page2.#Box105.HullNumber

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
disableDefault
moveTo("MatID_ExamEndDate")
endmethod

Object : OPERATIONS_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    AddPop.addText("&Locate")
    AddPop.addText("&Delete")
    examMenu.addPopUp("&Record", AddPop)
    examMenu.addText("&Quit")
    examMenu.addText("&Help")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    action(DataInsertRecord)
  endif
endmethod
```

Object : OPERATIONS_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
      case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
"+Operati_Ship_ShipName_FK3+"?") then
          deleteRecord()
        endif
      case eventInfo.menuChoice() = "&Help":
```

```

        case eventInfo.menuChoice() = "&Quit":
            reply=msgQuestion("Quit","Are you sure you want to leave this form?")
            If reply = "Yes" then
                close()
            else
                return
            endif
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box62.OperationGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box62.EccttGrade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box61.OperationID_ExamEndDate

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue AnyType
endVar

tempValue=Self.value
doDefault
if eventInfo.errorCode()=0 then
  if eventInfo.newValue() > today() then
    Self.value=tempValue
    msgStop("Problem","ExamEndDate cannot be a future date!")
    eventInfo.setErrorCode(-1)
  endif
endif
endmethod

```

Object : #Page2.#Box60.INSERT_NEW_EXAM_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
 action(dataInsertRecord)
 endmethod

Object : #Page2.#Box60.Operati_Ship_ShipName_FK3

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
 doDefault
 if eventInfo.errorCode() = 0 then
 ;input accepted by Paradox
 else
 msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")
 endif
 endmethod

Object : #Page2.#Box60.HullNumber

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
 disableDefault
 moveTo("OperationID_ExamEndDate")
 endmethod

Object : PM_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

  AddPop.addText("&Locate")
  AddPop.addText("&Delete")
  examMenu.addPopUp("&Record", AddPop)
  examMenu.addText("&Quit")
  examMenu.show()
  maximize()
  hideSpeedBar()
  edit()
  action(dataInsertRecord)
endif
endmethod
```

Object : PM_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

  Switch
    case eventInfo.menuChoice() = "&Locate":
      action(DataSearch)
    case eventInfo.menuChoice() = "&Delete":
      if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
"+PM_ID_Ship_ShipName_FK6+"?") then
        deleteRecord()
      endif
    case eventInfo.menuChoice() = "&Quit":
      reply = msgQuestion("Quit", "Are you sure you want to leave this form?")
```

```

        If reply = "Yes" then
            close()
        else
            return
        endif
    endSwitch
endif
endmethod

```

Object : #Page2.#Box77.ProgramManageGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
    TC TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, or UNS")
            eventInfo.setErrorCode(-1)
    endSwitch
endif
endmethod

```

Object : #Page2.#Box76.HeatStress

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String

```

```

choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:
self.value=tempValue
msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box76.HearingConsGrade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
tempValue anyType
choice1 String
choice2 String
choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:

```

```

        self.value=tempValue
        msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
        eventInfo.setErrorCode(-1)
    endSwitch
endIf
endmethod

```

Object : #Page2.#Box76.PM_ID_ExamEndDate

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
        var
            tempValue AnyType
        endVar

        tempValue=Self.value
        doDefault
        if eventInfo.errorCode()=0 then
            if eventInfo.newValue() > today() then
                Self.value=tempValue
                msgStop("Problem","ExamEndDate cannot be a future date!")
                eventInfo.setErrorCode(-1)
            endif
        endif
    endmethod

```

Object : #Page2.#Box76.OpLogsGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
        var
            tempValue anyType
            choice1 String
            choice2 String
            choice3 String
            choice4 String
            choice5 String
            TC TCursor
        endVar
        choice1="SAT"
        choice2="UNS"
        choice3="EXC"
        choice4="GOOD"
        choice5="NA"
        tempValue=self.value
        doDefault
        if eventInfo.errorCode()=0 then
            switch
                case eventInfo.newValue()=choice1:
                case eventInfo.newValue()=choice2:
                case eventInfo.newValue()=choice3:
                case eventInfo.newValue()=choice4:

```



```

        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.QA_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
    TC TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.OLV_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String

```

```

choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:
self.value=tempValue
msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box76.ElectSafetyGrade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
tempValue anyType
choice1 String
choice2 String
choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:

```

```

        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.TagoutGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
    TC TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.MGTESR_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String

```

```

choice3 String
choice4 String
choice5 String
TC      TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:
self.value=tempValue
msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box76.BearingRecsGrade

MethodName : changeValue

Source :

```

method changeValue(var eventInfo ValueEvent)
var
tempValue anyType
choice1 String
choice2 String
choice3 String
choice4 String
choice5 String
TC      TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:

```

```

        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.LegalRecsGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
    TC TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.DETA_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String

```

```

choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:
self.value=tempValue
msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box76.DJWTT_Grade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
tempValue anyType
choice1 String
choice2 String
choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:

```

```

        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.FOQM_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
    TC TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endIf
endmethod

```

Object : #Page2.#Box76.BWFW_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String

```

```

choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:
case eventInfo.newValue()=choice5:
otherwise:
self.value=tempValue
msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS or NA")
eventInfo.setErrorCode(-1)
endSwitch
endif
endmethod

```

Object : #Page2.#Box76.LOQM_Grade

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
tempValue anyType
choice1 String
choice2 String
choice3 String
choice4 String
choice5 String
TC    TCursor
endVar
choice1="SAT"
choice2="UNS"
choice3="EXC"
choice4="GOOD"
choice5="NA"
tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
switch
case eventInfo.newValue()=choice1:
case eventInfo.newValue()=choice2:
case eventInfo.newValue()=choice3:
case eventInfo.newValue()=choice4:

```



```

        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either EXC, GOOD, SAT, UNS, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box75.INSERT_NEW_PM_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
 action(dataInsertRecord)
 endmethod

Object : #Page2.#Box75.PM_ID_Ship_ShipName_FK6

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
 doDefault
 if eventInfo.errorCode() = 0 then
 ;input accepted by Paradox
 else
 msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")
 endif
 endmethod

Object : #Page2.#Box75.HullNumber

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
 disableDefault
 moveTo("PM_ID_ExamEndDate")
 endmethod

Object : TRAINING_INPUT_FORM

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    AddPop.addText("&Locate")
    AddPop.addText("&Delete")
    examMenu.addPopUp("&Record", AddPop)
    examMenu.addText("&Quit")
    examMenu.addText("&Help")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    action(DataInsertRecord)
  endif
endmethod
```

Object : TRAINING_INPUT_FORM

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
      case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete
        "+Trainin_Ship_ShipName_FK7+"?") then
          deleteRecord()
        endif
      case eventInfo.menuChoice() = "&Help":
```

```

        case eventInfo.menuChoice() = "&Quit":
            reply=msgQuestion("Quit","Are you sure you want to leave this form?")
            If reply = "Yes" then
                close()
            else
                return
            endif
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box86.TrainingProgramGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
    tempValue anyType
    choice1 String
    choice2 String
    choice3 String
    choice4 String
    choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
    switch
        case eventInfo.newValue()=choice1:
        case eventInfo.newValue()=choice2:
        case eventInfo.newValue()=choice3:
        case eventInfo.newValue()=choice4:
        case eventInfo.newValue()=choice5:
        otherwise:
            self.value=tempValue
            msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
            eventInfo.setErrorCode(-1)
        endSwitch
    endif
endmethod

```

Object : #Page2.#Box85.TrainingID_ExamEndDate

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)

```

var
  tempValue AnyType
endVar

tempValue=Self.value
doDefault
if eventInfo.errorCode()=0 then
  if eventInfo.newValue() > today() then
    Self.value=tempValue
    msgStop("Problem","ExamEndDate cannot be a future date!")
    eventInfo.setErrorCode(-1)
  endif
endif
endmethod

```

Object : #Page2.#Box85.NrOfSatOilKing

MethodName : action

```

Source : method action(var eventInfo ActionEvent)
Switch
  case propType.value="DSL":
    if eventInfo.id()=fieldForward then
      disableDefault
      NrOfSatENOW.moveTo()
    endif
  case propType.value="GT":
    if eventInfo.id()=fieldForward then
      disableDefault
      NrOfSatPACC.moveTo()
    endif
endSwitch
endmethod

```

Object : #Page2.#Box85.TrainingGrade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

```

```

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box85.PQS_Grade

MethodName : changeValue

```

Source : method changeValue(var eventInfo ValueEvent)
var
  tempValue anyType
  choice1 String
  choice2 String
  choice3 String
  choice4 String
  choice5 String
endVar

choice1="SAT"
choice2="UNS"
choice3="NA"
choice4="GOOD"
choice5="EXC"

tempValue=self.value
doDefault
if eventInfo.errorCode()=0 then
  switch
    case eventInfo.newValue()=choice1:
    case eventInfo.newValue()=choice2:
    case eventInfo.newValue()=choice3:
    case eventInfo.newValue()=choice4:
    case eventInfo.newValue()=choice5:
    otherwise:
      self.value=tempValue
      msgStop("Problem","Value must be either SAT, UNS, GOOD, EXC, or NA")
      eventInfo.setErrorCode(-1)
  endSwitch
endif
endmethod

```

Object : #Page2.#Box84.NrOfSatEDG_SWBD_Op

MethodName : action

Source : method action(var eventInfo ActionEvent)
if eventInfo.id()=fieldForward then
 disableDefault
 KeyPersonnelLeaving.moveTo()
endif
endmethod

Object : #Page2.#Box82.NrOfSatMsgr_EngOp

MethodName : action

Source : method action(var eventInfo ActionEvent)
if eventInfo.id()=fieldForward then
 disableDefault
 KeyPersonnelLeaving.moveTo()
endif
endmethod

Object : #Page2.#Box81.INSERT_NEW_TRAIN_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
 action(dataInsertRecord)
endmethod

Object : #Page2.#Box81.HullNumber

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
 disableDefault
 moveto("TrainingID_ExamEndDate")
endmethod

Object : #Page2.#Box81.Trainin_Ship_ShipName_FK7

MethodName : changeValue

Source : method changeValue(var eventInfo ValueEvent)
 doDefault
 if eventInfo.errorCode() = 0 then
 ;input accepted by Paradox
 else
 msgStop("Problem", "The ship you entered is not valid. You must enter the ship first!")
 endif
endmethod

Object : BOILER_FLEX_QUERY

MethodName : Const

Source : Const
ViewFlex1=301
ViewFlex2=302
ViewFlex3=303
ViewFlex4=304
ViewFlex5=305
PrintFlex1=313
PrintFlex2=314
PrintFlex3=315
PrintFlex4=316
PrintFlex5=317
PrintFlex6=318
endConst

Object : BOILER_FLEX_QUERY

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
var
tc Tcursor
examMenu, View, Print, ReportMenu Menu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewFlexPop PopUpMenu
PrintFlexPop PopUpMenu
examtype PopupMenu
endVar
if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else
; This code executes only for the form.

PrintPop.addText("Flex &1", "", PrintFlex1)
PrintPop.addText("Flex &2", "", PrintFlex2)
PrintPop.addText("Flex &3", "", PrintFlex3)
PrintPop.addText("Flex &4", "", PrintFlex4)
PrintPop.addText("Flex &5", "", PrintFlex5)
PrintPop.addText("&Graph", "", PrintFlex6)

ViewPop.addText("Flex &1", "", ViewFlex1)
ViewPop.addText("Flex &2", "", ViewFlex2)
ViewPop.addText("Flex &3", "", ViewFlex3)
ViewPop.addText("Flex &4", "", ViewFlex4)
ViewPop.addText("Flex &5", "", ViewFlex5)

examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)

```

examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endif
endmethod

```

Object : BOILER_FLEX_QUERY

MethodName : menuAction

```

Source : method menuAction(var eventInfo MenuEvent)
var
myRep Report
reply String
choiceld SmallInt
m menu
endVar
choiceld=eventInfo.id()

if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else
; This code executes only for the form.

Switch
case eventInfo.menuChoice() ="&Help":
case eventInfo.menuChoice() ="&Quit":
reply=msgQuestion("Quit","Are you sure you want to leave this form?")
If reply = "Yes" then
close()
else
return
endif
endSwitch
Switch
case choiceld =ViewFlex1:
myRep.open("Flex1",WinStyleMaximize)
hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex2:
myRep.open("Flex2",WinStyleMaximize)
hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex3:
myRep.open("Flex3",WinStyleMaximize)

```



```

hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex4:
myRep.open("Flex4",WinStyleMaximize)
hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex5:
myRep.open("Flex5", WinStyleMaximize)
hideSpeedBar()
m.addText("")
m.show()
case choiceld =PrintFlex1:
myRep.print("Flex1")
case choiceld =PrintFlex2:
myRep.print("Flex2")
case choiceld =PrintFlex3:
myRep.print("Flex3")
case choiceld =PrintFlex4:
myRep.print("Flex4")
case choiceld =PrintFlex5:
myRep.print("Flex5")
case choiceld =PrintFlex6:
myRep.print("BoilFlex")
endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

```

Source : method setFocus(var eventInfo Event)
var
examMenu, View, Print, ReportMenu Menu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewFlexPop PopUpMenu
PrintFlexPop PopUpMenu
examtype PopupMenu
endVar
if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else
; This code executes only for the form.

PrintPop.addText("Flex &1", "",PrintFlex1)
PrintPop.addText("Flex &2", "",PrintFlex2)
PrintPop.addText("Flex &3", "",PrintFlex3)
PrintPop.addText("Flex &4", "",PrintFlex4)
PrintPop.addText("Flex &5", "",PrintFlex5)

```

```
PrintPop.addText("&Graph","",PrintFlex6)
```

```
ViewPop.addText("Flex &1","",ViewFlex1)  
ViewPop.addText("Flex &2","",ViewFlex2)  
ViewPop.addText("Flex &3","",ViewFlex3)  
ViewPop.addText("Flex &4","",ViewFlex4)  
ViewPop.addText("Flex &5","",ViewFlex5)
```

```
examMenu.addPopUp("&View", ViewPop)  
examMenu.addPopUP("&Print", PrintPop)  
examMenu.addText("&Quit")  
examMenu.show()  
maximize()  
hideSpeedBar()  
endif  
endmethod
```

Object : #Page2.RESET_GRAPH_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)  
var  
tc Tcursor  
endVar  
tc.open("percent")  
tc.edit()  
tc.empty()  
tc.endEdit()  
endmethod
```

Object : #Page2.#Box23.LEVEL2_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)  
var  
tc tCursor  
tbl table  
numberOfFlexes Number  
totalNumberOfFlexes Number  
flexPercentage Number  
myQuery Query  
examDate1 Date  
examDate2 Date  
endVar  
doDefault  
examDate1=date("01/01/00")  
examDate2=date("12/31/99")  
examDate1.view("Enter start date (I.E. 01/01/95)")  
examDate2.view("Enter stop date (I.E. 12/12/99)")  
  
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
| _EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE OR =REOPPE|
```

```
SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =STM|
```

```
MATERIAL.DB | MatID_Ship_ShipName_FK4 | TtlNrOfLevel2Flex |
| _EG02 | Check |
```

```
MATERIAL.DB | TtlNrOfBoilersFlexed | BoilerFlexComments |
| Check | Check |
```

EndQuery

```
doDefault
empty("flex2")
executeQBE(myQuery, "flex2.db")
tbl.attach("flex2")
numberOfFlexes=tbl.cSum("TtlNrOfLevel2Flex")
msgInfo("Level Two Boiler Flex","The total number of Level Two Flexes are "
+strVal(NumberOfFlexes))
TotalNumberOfFlexes=tbl.cSum("TtlNrOfBoilersFlexed")
if totalNumberOfFlexes <> 0 then
msgInfo("Level Two Boiler Flex","The total number of Boiler Flexes are "
+strVal(totalNumberOfFlexes))
FlexPercentage=(numberOfFlexes/totalNumberOfFlexes)*100
msgInfo("Level Two Boiler Flex","The Level Two Boiler Flex percentage is "
+strVal(FlexPercentage))
else
msgStop("Problem","The TOTAL NUMBER of boiler flexes is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=flexPercentage
tc("PropType")="Lvl2"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box23.LEVEL5_LIST_SHIPS_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar
newView.open("Flex5")
```

endmethod

Object : #Page2.#Box23.LEVEL4_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar
newView.open("Flex4")
endmethod

Object : #Page2.#Box23.LEVEL3_LISTS_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar
newView.open("Flex3")
endmethod

Object : #Page2.#Box23.LEVEL2_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar
newView.open("Flex2")
endmethod

Object : #Page2.#Box23.LEVEL1_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar
newView.open("Flex1")
endmethod

Object : #Page2.#Box23.LEVEL4_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var

```

tc tCursor
tbl table
numberOfFlexes Number
totalNumberOfFlexes Number
flexPercentage Number
myQuery Query
examDate1 Date
examDate2 Date
endVar
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 12/12/99)")

```

```
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE OR REOPPE|

```

```

SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =STM|

```

```

MATERIAL.DB | MatID_Ship_ShipName_FK4 | TtlNrOfLevel4Flex |
|_EG02 | Check |

```

```

MATERIAL.DB | TtlNrOfBoilersFlexed | BoilerFlexComments |
| Check | Check |

```

```
EndQuery
```

```

doDefault
empty("flex4")
executeQBE(myQuery, "flex4.db")
tbl.attach("flex4")
numberOfFlexes=tbl.cSum("TtlNrOfLevel4Flex")
msgInfo("Level Four Boiler Flex", "The total number of Level Four Flexes are "
+strVal(NumberOfFlexes))
TotalNumberOfFlexes=tbl.cSum("TtlNrOfBoilersFlexed")
if totalNumberOfFlexes <> 0 then
msgInfo("Level Four Boiler Flex", "The total number of Boiler Flexes are "
+strVal(totalNumberOfFlexes))
FlexPercentage=(numberOfFlexes/totalNumberOfFlexes)*100
msgInfo("Level Four Boiler Flex", "The Level Four Boiler Flex percentage is "
+strVal(FlexPercentage))
else
msgStop("Problem", "The total number of flexes is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()

```

```

tc.("Percentage")=flexPercentage
tc.("PropType")="Lvl4"
tc.("examDate1")=examDate1
tc.("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box23.LEVEL5_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc tCursor
tbl table
numberOfFlexes Number
totalNumberOfFlexes Number
flexPercentage Number
myQuery Query
examDate1 Date
examDate2 Date
endVar
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 12/12/99)")

myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE OR REOPPE|

```

```

SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =STM|

```

```

MATERIAL.DB | MatID_Ship_ShipName_FK4 | TtlNrOfLevel5Flex |
|_EG02 | Check |

```

```

MATERIAL.DB | TtlNrOfBoilersFlexed | BoilerFlexComments |
| Check | Check |

```

EndQuery

```

doDefault
empty("flex5")
executeQBE(myQuery, "flex5.db")
tbl.attach("flex5")
numberOfFlexes=tbl.cSum("TtlNrOfLevel5Flex")
msgInfo("Level Five Boiler Flex","The total number of Level Five Flexes are "
+strVal(NumberOfFlexes))

```

```

TotalNumberOfFlexes=tbl.cSum("TtlNrOfBoilersFlexed")
if totalNumberOfFlexes <> 0 then
  msgInfo("Level Five Boiler Flex","The total number of Boiler Five are "
    +strVal(totalNumberOfFlexes))
  FlexPercentage=(numberOfFlexes/totalNumberOfFlexes)*100
  msgInfo("Level Five Boiler Flex","The Level Five Boiler Flex percentage is "
    +strVal(FlexPercentage))
else
  msgStop("Problem","The total number of flexes is 0, you cannot divide by 0!")
  return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=flexPercentage
tc("PropType")="Lvl5"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box23.LEVEL1_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
  tc tCursor
  tbl table
  numberOfFlexes Number
  totalNumberOfFlexes Number
  flexPercentage Number
  myQuery Query
  myQuery1 Query
  examDate1 Date
  examDate2 Date
  PropType String
  ExamType String
endVar
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 12/12/99)")

```

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE OR =REOPPE|

```

```

SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =STM|

```

```

MATERIAL.DB | MatID_Ship_ShipName_FK4 | TtlNrOfLevel1Flex |
| _EG02 | Check |

```

```

MATERIAL.DB | TtlNrOfBoilersFlexed | BoilerFlexComments |
| Check | Check |

```

```

EndQuery
doDefault
empty("flex1")
executeQBE(myQuery, "flex1.db")
tbl.attach("flex1")
numberOfFlexes=tbl.cSum("TtlNrOfLevel1Flex")
msgInfo("Level One Boiler Flex","The total number of Level One Flexes are "
+strVal(NumberOfFlexes))
TotalNumberOfFlexes=tbl.cSum("TtlNrOfBoilersFlexed")
if totalNumberOfFlexes <> 0 then
    msgInfo("Level One Boiler Flex","The total number of Boiler Flexes are "
    +strVal(totalNumberOfFlexes))
    FlexPercentage=(numberOfFlexes/totalNumberOfFlexes)*100
    msgInfo("Level One Boiler Flex","The Level One Boiler Flex percentage is "
    +strVal(FlexPercentage))
else
    msgStop("Problem","The TOTAL NUMBER of boiler flexes is 0, you cannot divide by 0!")
    return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=flexPercentage
tc("PropType")="Lvl1"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box23.LEVEL3_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
numberOfFlexes Number
totalNumberOfFlexes Number
flexPercentage Number
myQuery Query
examDate1 Date
examDate2 Date
endVar
doDefault
examDate1=date("01/01/00")

```



```

examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 12/12/99)")

```

```

myQuery=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE OR =REOPPE|

```

```

SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =STM|

```

```

MATERIAL.DB | MatID_Ship_ShipName_FK4 | TtlNrOfLevel3Flex |
|_EG02 | Check |

```

```

MATERIAL.DB | TtlNrOfBoilersFlexed | BoilerFlexComments |
| Check | Check |

```

```

EndQuery

```

```

doDefault
empty("flex3")
executeQBE(myQuery, "flex3.db")
tbl.attach("flex3")
numberOfFlexes=tbl.cSum("TtlNrOfLevel3Flex")
msgInfo("Level Three Boiler Flex","The total number of Level Three Flexes are "
+strVal(NumberOfFlexes))
TotalNumberOfFlexes=tbl.cSum("TtlNrOfBoilersFlexed")
if totalNumberofFlexes <> 0 then
msgInfo("Level Three Boiler Flex","The total number of Boiler Flexes are "
+strVal(totalNumberOfFlexes))
FlexPercentage=(numberOfFlexes/totalNumberOfFlexes)*100
msgInfo("Level Three Boiler Flex","The Level Three Boiler Flex percentage is "
+strVal(FlexPercentage))
else
msgStop("Problem","The TOTAL NUMBER of boiler flexes is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=flexPercentage
tc("PropType")="Lvl3"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : ECCTT_QUERY

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  tc Tcursor
  examMenu Menu
  ReportPop PopUpMenu
  ViewPop PopUpMenu
  PrintPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    PrintPop.addText("U&nsat ECCTT Report")
    PrintPop.addText("S&at ECCTT Report")
    PrintPop.addText("&Graph")
    ViewPop.addText("&Unsat ECCTT Report")
    ViewPop.addText("&Sat ECCTT Report")
    ReportPop.addPopUp("&View", ViewPoP)
    ReportPop.addPopUp("&Print", PrintPoP)
    examMenu.addPopUp("&Report", ReportPoP)
    examMenu.addText("&Quit")
    examMenu.show()
    maximize()
    hideSpeedBar()
    tc.open("percent")
    tc.edit()
    tc.empty()
    tc.endEdit()
endif
endmethod
```

Object : ECCTT_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
  m menu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.
```

Switch

```
    case eventInfo.menuChoice() = "&Unsat ECCTT Report":
        myRep.open("ECCTT2", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
    case eventInfo.menuChoice() = "&Sat ECCTT Report":
        myRep.open("ECCTT", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
    case eventInfo.menuChoice() = "&Graph":
        myRep.print("ECCTT1")
    case eventInfo.menuChoice() = "U&nsat ECCTT Report":
        myRep.print("ECCTT2")
    case eventInfo.menuChoice() = "S&at ECCTT Report":
        myRep.print("ECCTT")
    case eventInfo.menuChoice() = "&Help":

    case eventInfo.menuChoice() = "&Quit":
        reply=msgQuestion("Quit", "Are you sure you want to leave this form?")
        If reply = "Yes" then
            close()
        else
            return
        endif
    endSwitch
endif
endmethod
```

Object : #Page2

MethodName : setFocus

```
Source : method setFocus(var eventInfo Event)
var
    examMenu Menu
    ReportPop PopUpMenu
    ViewPop PopUpMenu
    PrintPop PopUpMenu
endVar
PrintPop.addText("U&nsat ECCTT Report")
PrintPop.addText("S&at ECCTT Report")
PrintPop.addText("&Graph")
ViewPop.addText("&Unsat ECCTT Report")
ViewPop.addText("&Sat ECCTT Report")
ReportPop.addPopUp("&View", ViewPop)
ReportPop.addPopUp("&Print", PrintPop)
examMenu.addPopUp("&Report", ReportPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
endmethod
```

Object : #Page2.RESET_GRAPH_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc Tcursor
endVar
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

Object : #Page2.LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
choice string
endVar
choice="SAT"
choice.view("Enter SAT or UNS to view the list of ships")
switch
case choice="SAT": newView.open("ECCTT")
case choice="UNS": newView.open("ECCTT2")
otherwise:
msgStop("Problem","The choices for ECCTT grade are SAT, or UNS only!")
return
endSwitch
endmethod

Object : #Page2.ECCTT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatECCTT Number
totalNumberOfECCTT Number
satPercentageECCTT Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
propType String
examType String

```

endVar
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

if propType="ALL" then
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check_EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | EccttGrade | EccttComments |
| Check =SAT | Check |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check_EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | EccttGrade | EccttComments |
| Check | Check |

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

```

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | EccttGrade | EccttComments |
| Check =UNS | Check |

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

SHIP.DB | ShipName | PropType |
|_EG01 | Check =~PropType|

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | EccttGrade | EccttComments |
| Check =SAT | Check |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

SHIP.DB | ShipName | PropType |
|_EG01 | Check =~PropType|

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | EccttGrade | EccttComments |
| Check | Check |

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

SHIP.DB | ShipName | PropType |
|_EG01 | Check =~PropType|

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG01 | Check >=~examDate1, <=~examDate2|

```

OPERATIO.DB | EccttGrade | EccttComments |
| Check =UNS | Check      |

```

```

EndQuery
endif

```

```

empty("ECCTT")
empty("ECCTT1")
empty("ECCTT2")
executeQBE(myQuery, "ECCTT.db")
executeQBE(myQuery1, "ECCTT1.db")
executeQBE(myQuery2, "ECCTT2.db")
tbl.attach("ECCTT")
tbl1.attach("ECCTT1")
numberOfSatECCTT=tbl.cCount("EccttGrade")
msgInfo("ECCTT","The total number of sats are "
+strVal(NumberOfSatECCTT))
TotalNumberOfECCTT=tbl1.cCount("EccttGrade")
if TotalNumberOfECCTT <> 0 then
  msgInfo("ECCTT","The total number is "
+strVal(totalNumberOfECCTT))
  SatPercentageECCTT=(numberOfSatECCTT/totalNumberOfECCTT)*100
  msgInfo("ECCTT","The sat percentage is "
+strVal(satPercentageECCTT))
else
  msgStop("Problem","The total number of ECCTT grades is 0, you cannot divide by 0!")
  return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageECCTT
tc("PropType")=PropType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : FIRE_DRILL_QUERY

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  tc Tcursor
  examMenu, View, Print, ReportMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
  PrintPop PopUpMenu
  ViewPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    ReportPop.addPopUp("&View", ViewPop)
    ReportPop.addPopUp("&Print", PrintPop)
    PrintPop.addText("FireDrill &1")
    PrintPop.addText("FireDrill &2")
    PrintPop.addText("FireDrill &3")
    PrintPop.addText("&Graph")

    ViewPop.addText("Fire Drill &1")
    ViewPop.addText("Fire Drill &2")
    ViewPop.addText("Fire Drill &3")

    examMenu.addPopUp("&View", ViewPop)
    examMenu.addPopUP("&Print", PrintPop)
    examMenu.addText("&Quit")
    examMenu.show()
    maximize()
    hideSpeedBar()
    tc.open("percent")
    tc.edit()
    tc.empty()
    tc.endEdit()
  endif
endmethod
```

Object : FIRE_DRILL_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
  m menu
endVar
```



```

if eventInfo.isPreFilter()
then
    ; This code executes for each object on the form.

else
    ; This code executes only for the form.
Switch
    case eventInfo.menuChoice() = "&Locate":
        action(DataSearch)
    case eventInfo.menuChoice() = "&Delete":
        if "Yes" = msgQuestion("Delete Record", "Are you sure you want to delete this RECORD?") then
            deleteRecord()
        endif
    case eventInfo.menuChoice() = "Fire Drill &1":
        myRep.open("FireDrl1", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
    case eventInfo.menuChoice() = "Fire Drill &2":
        myRep.open("FireDrl2", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
    case eventInfo.menuChoice() = "Fire Drill &3":
        myRep.open("FireDrl3", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
    case eventInfo.menuChoice() = "FireDrill &1":
        myRep.print("FireDrl1")
    case eventInfo.menuChoice() = "FireDrill &2":
        myRep.print("FireDrl2")
    case eventInfo.menuChoice() = "FireDrill &3":
        myRep.print("FireDrl3")
    case eventInfo.menuChoice() = "&Graph":
        myRep.print("FireDril")
    case eventInfo.menuChoice() = "&Help":

    case eventInfo.menuChoice() = "&Quit":
        reply = msgQuestion("Quit", "Are you sure you want to leave this form?")
        If reply = "Yes" then
            close()
        else
            return
        endif
endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

Source : method setFocus(var eventInfo Event)

```

var
  examMenu, View, Print, ReportMenu Menu
  ReportPop PopUpMenu
  AddPop PopUpMenu
  PrintPop PopUpMenu
  ViewPop PopUpMenu
endVar
ReportPop.addPopUp("&View", ViewPop)
ReportPop.addPopUp("&Print", PrintPop)
PrintPop.addText("FireDrill &1")
PrintPop.addText("FireDrill &2")
PrintPop.addText("FireDrill &3")
PrintPop.addText("&Graph")

ViewPop.addText("Fire Drill &1")
ViewPop.addText("Fire Drill &2")
ViewPop.addText("Fire Drill &3")

examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUp("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
endmethod

```

Object : #Page2.#Box15.FD3_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
  newView tableView
endVar
newView.open("fireDr13")
endmethod

```

Object : #Page2.#Box15.FD2_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
  newView tableView
endVar
newView.open("fireDr12")
endmethod

```

Object : #Page2.#Box15.FD1_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
  newView tableView
endVar
newView.open("fireDr11")
endmethod

```

Object : #Page2.#Box15.FIRE_DRILL_3_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
  tc tCursor
  tbl table
  tbl1 table
  tbl2 table
  numberOfSats Number
  totalNumberOfDrills Number
  satPercentage Number
  myQuery Query
  myQuery1 Query
  examDate1 Date
  examDate2 Date
  PropType String
  ExamType String
endVar
empty("temp3")
tbl2.attach("temp3.db")
doDefault
empty("firedr13")
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter end date (I.E. 01/01/95)")
propType.view("Enter (GT/STM/DSL)")
switch
  case propType="GT":
  case propType="STM":
  case propType="DSL":
  case propType="ALL":
  otherwise:
    msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, ALL only!")
    return
endSwitch
examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
  case examType="OPPE":
  case examType="REOPPE":
  case examType="LOE":
  case examType="RELOE":
  otherwise:
    msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")

```

```
return  
endSwitch
```

```
if propType="ALL" then  
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |  
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~examType |
```

```
SHIP.DB | ShipName | PropType |  
| Check _EG02, _EG01 | Check |
```

```
FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill3_Grade |FireDrillComments|  
|_EG02 | Check =SAT |Check|
```

```
EndQuery
```

```
myQuery1=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |  
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~ExamType |
```

```
SHIP.DB | ShipName | PropType |  
| Check _EG02, _EG01 | Check |
```

```
FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade | FireDrill2_Grade |  
|_EG02 | Check | Check |
```

```
FIREFIGH.DB | FireDrill3_Grade |FireDrillComments|  
| Check =SAT OR =UNS|Check|
```

```
EndQuery
```

```
else  
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |  
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~examType |
```

```
SHIP.DB | ShipName | PropType |  
| Check _EG02, _EG01 | Check =~propType|
```

```
FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill3_Grade |FireDrillComments|  
|_EG02 | Check =SAT |Check|
```

```
EndQuery
```

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~ExamType |

SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =~proptype|

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade | FireDrill2_Grade |
|_EG02 | Check | Check |

FIREFIGH.DB | FireDrill3_Grade | FireDrillComments|
| Check =SAT OR =UNS|Check|

EndQuery
endif

```
empty("fireDr3")
executeQBE(myQuery, "temp3.db")
executeQBE(myQuery1, "fireDrB.db")
tbl2.add("firedr3.db", "True", "False")
tbl.attach("fireDr3")
tbl1.attach("fireDrB")
numberOfSats=tbl.cCount("FireDrill3_Grade")
msgInfo("Fire Drill Three", "The total number of sat drills are "
+strVal(NumberOfSats))
TotalNumberOfDrills=tbl1.cCount("FireDrill3_Grade")
if TotalNumberOfDrills <> 0 then
    msgInfo("Fire Drill Three", "The total number of fire drills are "
+strVal(totalNumberOfDrills))
    SatPercentage=(numberOfSats/totalNumberOfDrills)*100
    msgInfo("Fire Drill Three", "The sat fire drill percentage is "
+strVal(satPercentage))
else
    msgStop("Problem", "The total number of drills is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentage
tc("PropType")=propType
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box15.FIRE_DRILL_2_BUTTON

MethodName : pushButton

```
Source :      method pushButton(var eventInfo Event)
              var
              tc tCursor
              tbl table
              tbl1 table
              tbl2 table
              numberOfSats Number
              totalNumberOfDrills Number
              satPercentage Number
              myQuery Query
              myQuery1 Query
              examDate1 Date
              examDate2 Date
              PropType String
              ExamType String
              endVar
              doDefault
              empty("temp2")
              tbl2.attach("temp2.db")
              empty("firedrl2")
              examDate1=date("01/01/00")
              examDate2=date("12/31/99")
              propType="GT"
              examType="OPPE"
              examDate1.view("Enter start date (I.E. 01/01/95)")
              examDate2.view("Enter stop date (I.E. 01/01/95)")
              propType.view("Enter (GT/STM/DSL)")
              switch
              case propType="GT":
              case propType="STM":
              case propType="DSL":
              case propType="ALL":
              otherwise:
              msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
              return
              endSwitch
              examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
              switch
              case examType="OPPE":
              case examType="REOPPE":
              case examType="LOE":
              case examType="RELOE":
              otherwise:
              msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
              return
              endSwitch

              if propType="ALL" then
              myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate

|ExamType |

```

|_EG01 | Check >=~examDate1, <=~examDate2|Check =~examType |
SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check |
FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill2_Grade |FireDrillComments|
|_EG02 | Check =SAT |Check |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~ExamType |
SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check |
FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade | FireDrill2_Grade |
|_EG02 | Check | Check |
FIREFIGH.DB | FireDrill3_Grade |FireDrillComments|
| Check |Check |

```

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~examType |
SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =~propType|
FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill2_Grade |FireDrillComments|
|_EG02 | Check =SAT |Check |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2|Check =~ExamType |
SHIP.DB | ShipName | PropType |

```

| Check _EG02, _EG01 | Check =~proptype|

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade | FireDrill2_Grade |
| _EG02 | Check | Check |

FIREFIGH.DB | FireDrill3_Grade | FireDrillComments|
| Check | Check |

```
EndQuery
endif
executeQBE(myQuery, "temp2.db")
executeQBE(myQuery1, "fireDriA.db")
tbl2.add("firedri2.db", "True", "False")
tbl.attach("fireDri2")
tbl1.attach("fireDriA")
numberOfSats=tbl.cCount("FireDrill2_Grade")
msgInfo("Fire Drill Two", "The total number of sat drills are "
+strVal(NumberOfSats))
TotalNumberOfDrills=tbl1.cCount("FireDrill2_Grade")
if TotalNumberOfDrills <> 0 then
msgInfo("Fire Drill Two", "The total number of fire drills are "
+strVal(totalNumberOfDrills))
SatPercentage=(numberOfSats/totalNumberOfDrills)*100
msgInfo("Fire Drill Two", "The sat fire drill percentage is "
+strVal(satPercentage))
else
msgStop("Problem", "The total number of drills is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentage
tc("PropType")=propType
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box15.FIRE_DRILL_1_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
tbl2 table
numberOfSats Number
totalNumberOfDrills Number
satPercentage Number


```

myQuery Query
myQuery1 Query
examDate1 Date
examDate2 Date
PropType String
ExamType String
endVar
doDefault
empty("temp1")
tbl2.attach("temp1.db")
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 12/12/99)")
propType.view("Enter (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch
examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch
if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2| Check =~examType|

```

```

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade |
|_EG02,_EG01 | Check =SAT |

```

```

FIREFIGH.DB | FireDrillComments |
| Check |

```

```

SHIP.DB | ShipName | PropType |
| Check _EG02 | Check |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2| Check =~examType|

SHIP.DB | ShipName | PropType|
| Check _EG02, _EG01 | Check |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade | FireDrill2_Grade |
|_EG02 | Check | Check |

FIREFIGH.DB | FireDrill3_Grade | FireDrillComments |
| Check | Check |

EndQuery

else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2| Check =~examType|

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade |
|_EG02, _EG01 | Check =SAT |

FIREFIGH.DB | FireDrillComments |
| Check |

SHIP.DB | ShipName | PropType |
| Check _EG02 | Check =~propType|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
|_EG01 | Check >=~examDate1, <=~examDate2| Check =~examType|

SHIP.DB | ShipName | PropType |
| Check _EG02, _EG01 | Check =~propType|

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireDrill1_Grade | FireDrill2_Grade |
|_EG02 | Check | Check |

FIREFIGH.DB | FireDrill3_Grade | FireDrillComments |
| Check | Check |

```

EndQuery
endif
doDefault
empty("fireDr11")
executeQBE(myQuery,"temp1.db")
executeQBE(myQuery1, "fireDr1A.db")
tbl2.add("fireDr11","True","False")
tbl.attach("fireDr11")
tbl1.attach("fireDr1A")
numberOfSats=tbl.cCount("FireDrill1_Grade")
msgInfo("Fire Drill One","The total number of sat drills are "
+strVal(NumberOfSats))
TotalNumberOfDrills=tbl1.cCount("FireDrill1_Grade")
if TotalNumberOfDrills <> 0 then
msgInfo("Fire Drill One","The total number of fire drills are "
+strVal(totalNumberOfDrills))
SatPercentage=(numberOfSats/totalNumberOfDrills)*100
msgInfo("Fire Drill One","The sat fire drill percentage is "
+strVal(satPercentage))
else
msgStop("Problem","The total number of drills is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentage
tc("PropType")=propType
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.RESET_GRAPH_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc Tcursor
endVar
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

```

Object : HIGH_POWER_DEMO_QUERY

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  tc Tcursor
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
  PrintPop PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    PrintPop.addText("&High Power Report")
    PrintPop.addText("&Graph")
    ReportPop.addText("&View")
    ReportPop.addPopUp("&Print",PrintPoP)
    examMenu.addPopUp("&Report", ReportPoP)
    examMenu.addText("&Quit")
    examMenu.show()
    maximize()
    hideSpeedBar()
    tc.open("percent")
    tc.edit()
    tc.empty()
    tc.endEdit()
endif
endmethod
```

Object : HIGH_POWER_DEMO_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
  m menu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() ="&View":
```

```

        myRep.open("HighPwr", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
    case eventInfo.menuChoice() = "&High Power Report":
        myRep.print("HighPwr")
    case eventInfo.menuChoice() = "&Graph":
        myRep.print("HiPwr")
    case eventInfo.menuChoice() = "&Help":

    case eventInfo.menuChoice() = "&Quit":
        reply=msgQuestion("Quit", "Are you sure you want to leave this form?")
        If reply = "Yes" then
            close()
        else
            return
        endif
    endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

```

Source : method setFocus(var eventInfo Event)
var
    examMenu Menu
    ReportPop PopUpMenu
    AddPoP PopUpMenu
    PrintPop PopUpMenu
endVar

PrintPop.addText("&High Power Report")
PrintPop.addText("&Graph")
ReportPop.addText("&View")
ReportPop.addPopUp("&Print", PrintPoP)
examMenu.addPopUp("&Report", ReportPoP)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
endmethod

```

Object : #Page2.#Box7.LIST_SHIPS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
    newView tableView
endVar

newView.open("Highpwr")

```

endmethod

Object : #Page2.#Box7.HIGH_PWR_DEMO_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
  tc tCursor
  tbl table
  tbl1 table
  numberOfSats Number
  totalNumberOfGrades Number
  SatPercentage Number
  myQuery Query
  myQuery1 Query
  examDate1 Date
  examDate2 Date
  PropType String
endvar
dodefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter the stop date (I.E. 01/01/95)")
propType.view("Enter the prop type (GT/STM/DSL)")
Switch
  case propType="GT":
  case propType="STM":
  case propType="DSL":
  case propType="ALL":
  otherwise:
    msgStop("Problem", "Your choice must be either GT, STM, DSL or ALL.")
  return
endSwitch

if propType="ALL" then
  myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate | HighPwrDemoGrade |
HighPwrComments |
  | _EG02 | Check >=~examDate1, <=~examDate2| Check =SAT | Check |
```

```
SHIP.DB | ShipName | PropType|
| Check _EG02 | Check |
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate | HighPwrDemoGrade |
HighPwrComments |
    | _EG02 | Check >=~examDate1, <=~examDate2| Check | Check |
```

```
SHIP.DB | ShipName | PropType|
    | Check _EG02 | Check |
```

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```
MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate | HighPwrDemoGrade |
HighPwrComments |
    | _EG02 | Check >=~examDate1, <=~examDate2| Check =SAT | Check |
```

```
SHIP.DB | ShipName | PropType |
    | Check _EG02 | Check =~propType|
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate | HighPwrDemoGrade |
HighPwrComments |
    | _EG02 | Check >=~examDate1, <=~examDate2| Check | Check |
```

```
SHIP.DB | ShipName | PropType |
    | Check _EG02 | Check =~propType|
```

EndQuery

endif

```
empty("Highpwr")
empty("Highpwr1")
executeQBE(myQuery, "Highpwr.db")
executeQBE(myQuery1, "Highpwr1.db")
```

```
tbl.attach("Highpwr")
tbl1.attach("Highpwr1")
numberOfSats=tbl.cCount("HighPwrDemoGrade")
msgInfo("Number of Sats", "The total number of sat high power demos are "
    +strVal(numberOfSats))
totalNumberOfGrades=tbl1.cCount("HighPwrDemoGrade")
if TotalNumberOfGrades <> 0 then
    msgInfo("Total Number of Grades", "The total number of high power demos are "
```

```

+strVal(totalNumberOfGrades)).
SatPercentage=(NumberOfSats/TotalNumberOfGrades)*100
msgInfo("Sat Percentage","The high power demo sat percentage is "
+strVal(SatPercentage))
else
msgStop("Problem","The total number of grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentage
tc("PropType")=propType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box7.RESET_GRAPH_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc Tcursor
endVar
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

```


Object : OPPE_LOE_AREA_QUERY

MethodName : Const

Source : Const
ViewUnsatOPPE=301
ViewUnsatOperation=302
ViewUnsatFireFighting=303
ViewUnsatMaterial=304
ViewUnsatTraining=305
ViewUnsatProgramManagment=306
ViewUnsatLOE=325
ViewSatOPPE=307
ViewSatOperation=308
ViewSatFireFighting=309
ViewSatMaterial=310
ViewSatTraining=311
ViewSatProgramManagement=312
ViewSatLOE=326
PrintUnsatOPPE=313
PrintUnsatOperation=314
PrintUnsatFireFighting=315
PrintUnsatMaterial=316
PrintUnsatTraining=317
PrintUnsatProgramManagment=318
PrintUnsatLOE=327
PrintSatOPPE=319
PrintSatOperation=320
PrintSatFireFighting=321
PrintSatMaterial=322
PrintSatTraining=323
PrintSatProgramManagement=324
PrintSatLOE=328
endConst

Object : OPPE_LOE_AREA_QUERY

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
var
tc Tcursor
examMenu, View, Print, ReportMenu Menu
ReportPop PopUpMenu
AddPoP PopUpMenu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewSatPop PopUpMenu
ViewUnsatPop PopUpMenu
PrintSatPop PopUpMenu
PrintUnsatPop PopUpMenu
examtype PopupMenu
endVar
if eventInfo.isPreFilter()

then
; This code executes for each object on the form.

else
; This code executes only for the form.

```
ViewUnsatPop.addText("&OPPE Percentage","",ViewUnsatOppe)
ViewUnsatPop.addText("Op&eration Percentage","",ViewUnsatOperation)
ViewUnsatPop.addText("&Fire Fighting Percentage","",ViewUnsatFireFighting)
ViewUnsatPop.addText("&Material Percentage","",ViewUnsatMaterial)
ViewUnsatPop.addText("&Training Percentage","",ViewUnsatTraining)
ViewUnsatPop.addText("&Program Management Percentage","",ViewUnsatProgramManagement)
ViewUnsatPop.addText("&LOE Percentage","",ViewUnsatLOE)
ViewSatPop.addText("&OPPE Percentage","",ViewSatOppe)
ViewSatPop.addText("Op&eration Percentage","",ViewSatOperation)
ViewSatPop.addText("&Fire Fighting Percentage","",ViewSatFireFighting)
ViewSatPop.addText("&Material Percentage","",ViewSatMaterial)
ViewSatPop.addText("&Training Percentage","",ViewSatTraining)
ViewSatPop.addText("&Program Management Percentage","",ViewSatProgramManagement)
ViewSatPop.addText("&LOE Percentage","",ViewSatLOE)
```

```
PrintUnsatPop.addText("&OPPE Percentage","",PrintUnsatOppe)
PrintUnsatPop.addText("Op&eration Percentage","",PrintUnsatOperation)
PrintUnsatPop.addText("&Fire Fighting Percentage","",PrintUnsatFireFighting)
PrintUnsatPop.addText("&Material Percentage","",PrintUnsatMaterial)
PrintUnsatPop.addText("&Training Percentage","",PrintUnsatTraining)
PrintUnsatPop.addText("&Program Management Percentage","",PrintUnsatProgramManagement)
PrintUnsatPop.addText("&LOE Percentage","",PrintUnsatLOE)
PrintSatPop.addText("&OPPE Percentage","",PrintSatOppe)
PrintSatPop.addText("Op&eration Percentage","",PrintSatOperation)
PrintSatPop.addText("&Fire Fighting Percentage","",PrintSatFireFighting)
PrintSatPop.addText("&Material Percentage","",PrintSatMaterial)
PrintSatPop.addText("&Training Percentage","",PrintSatTraining)
PrintSatPop.addText("&Program Management Percentage","",PrintSatProgramManagement)
PrintSatPop.addText("&LOE Percentage","",PrintSatLOE)
```

```
PrintPop.addText("&Graph")
PrintPop.addPopUp("&Sat",PrintSatPop)
PrintPop.addPopUp("&Unsat",PrintUnsatPop)
```

```
ViewPop.addPopUp("&Sat", ViewSatPop)
ViewPop.addPopUp("&Unsat",ViewUnsatPop)
```

```
examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endif
```

endmethod

Object : OPPE_LOE_AREA_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
  choiceld SmallInt
  m menu
endVar
choiceld=eventInfo.id()

if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

Switch
  case eventInfo.menuChoice() ="&Graph":
    myRep.print("OppeSum")
  case eventInfo.menuChoice() ="&Help":
  case eventInfo.menuChoice() ="&Quit":
    reply=msgQuestion("Quit","Are you sure you want to leave this form?")
    If reply = "Yes" then
      close()
    else
      return
    endif
endSwitch
Switch
  case choiceld =ViewSatOppe:
    myRep.open("oppeperc", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
  case choiceld =ViewSatOperation:
    myRep.open("operperc", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
  case choiceld =ViewSatFireFighting:
    myRep.open("FireFigh", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
  case choiceld =ViewSatMaterial:
    myRep.open("Matperc", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
```

```

m.show()
case choiceld =ViewSatTraining :
    myRep.open("Training", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewSatProgramManagement:
    myRep.open("ProgramM", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
    case choiceld =ViewSatLOE:
        myRep.open("LoeSum", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
case choiceld =ViewUnsatOPPE:
    myRep.open("oppeper1", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewUnsatOperation:
    myRep.open("operper1", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewUnsatFireFighting:
    myRep.open("FireFig1", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewUnsatMaterial:
    myRep.open("Matper1", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewUnsatTraining:
    myRep.open("Trainin1", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld = ViewUnsatProgramManagment:
    myRep.open("Program1", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
    case choiceld = ViewUnsatLOE:
        myRep.open("LoeSum1", WinStyleMaximize)
        hideSpeedBar()
        m.addText("")
        m.show()
case choiceld =PrintSatOppe:
    myRep.print("oppeperc")
case choiceld =PrintSatOperation:
    myRep.print("operperc")

```

```

case choiceld =PrintSatFireFighting:
    myRep.print("FireFigh")
case choiceld =PrintSatMaterial:
    myRep.print("Matperc")
case choiceld =PrintSatTraining :
    myRep.print("Training")
case choiceld =PrintSatProgramManagement:
    myRep.print("ProgramM")
case choiceld =PrintSatLOE:
    myRep.print("LoeSum")
case choiceld =PrintUnsatOPPE:
    myRep.print("oppeper1")
case choiceld =PrintUnsatOperation:
    myRep.print("operper1")
case choiceld =PrintUnsatFireFighting:
    myRep.print("FireFig1")
case choiceld =PrintUnsatMaterial:
    myRep.print("Matper1")
case choiceld =PrintUnsatTraining:
    myRep.print("Trainin1")
case choiceld = PrintUnsatProgramManagment:
    myRep.print("Program1")
case choiceld = PrintUnsatLOE:
    myRep.print("LoeSum1")
endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

Source : method setFocus(var eventInfo Event)

```

var
examMenu, View, Print, ReportMenu Menu
ReportPop PopUpMenu
AddPoP PopUpMenu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewSatPop PopUpMenu
ViewUnsatPop PopUpMenu
PrintSatPop PopUpMenu
PrintUnsatPop PopUpMenu
examtype PopupMenu
endVar

```

```

ViewUnsatPop.addText("&OPPE Percentage","",ViewUnsatOppe)
ViewUnsatPop.addText("Op&eration Percentage","",ViewUnsatOperation)
ViewUnsatPop.addText("&Fire Fighting Percentage","",ViewUnsatFireFighting)
ViewUnsatPop.addText("&Material Percentage","",ViewUnsatMaterial)
ViewUnsatPop.addText("&Training Percentage","",ViewUnsatTraining)
ViewUnsatPop.addText("&Program Management Percentage","",ViewUnsatProgramManagment)
ViewUnsatPop.addText("&LOE Percentage","",ViewUnsatLOE)
ViewSatPop.addText("&OPPE Percentage","",ViewSatOppe)

```

```

ViewSatPop.addText("Op&eration Percentage", "", ViewSatOperation)
ViewSatPop.addText("&Fire Fighting Percentage", "", ViewSatFireFighting)
ViewSatPop.addText("&Material Percentage", "", ViewSatMaterial)
ViewSatPop.addText("&Training Percentage", "", ViewSatTraining)
ViewSatPop.addText("&Program Management Percentage", "", ViewSatProgramManagement)
ViewSatPop.addText("&LOE Percentage", "", ViewSatLOE)

PrintUnsatPop.addText("&OPPE Percentage", "", PrintUnsatOppe)
PrintUnsatPop.addText("Op&eration Percentage", "", PrintUnsatOperation)
PrintUnsatPop.addText("&Fire Fighting Percentage", "", PrintUnsatFireFighting)
PrintUnsatPop.addText("&Material Percentage", "", PrintUnsatMaterial)
PrintUnsatPop.addText("&Training Percentage", "", PrintUnsatTraining)
PrintUnsatPop.addText("&Program Management Percentage", "", PrintUnsatProgramManagment)
PrintUnsatPop.addText("&LOE Percentage", "", PrintUnsatLOE)
PrintSatPop.addText("&OPPE Percentage", "", PrintSatOppe)
PrintSatPop.addText("Op&eration Percentage", "", PrintSatOperation)
PrintSatPop.addText("&Fire Fighting Percentage", "", PrintSatFireFighting)
PrintSatPop.addText("&Material Percentage", "", PrintSatMaterial)
PrintSatPop.addText("&Training Percentage", "", PrintSatTraining)
PrintSatPop.addText("&Program Management Percentage", "", PrintSatProgramManagement)
PrintSatPop.addText("&LOE Percentage", "", PrintSatLOE)

PrintPop.addText("&Graph")
PrintPop.addPopUp("&Sat", PrintSatPop)
PrintPop.addPopUp("&Unsat", PrintUnsatPop)

ViewPop.addPopUp("&Sat", ViewSatPop)
ViewPop.addPopUp("&Unsat", ViewUnsatPop)

examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
endmethod

```

Object : #Page2.RESET_GRAPH_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
  tc Tcursor
endVar
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

```

Object : #Page2.#Box22.#Button34

MethodName : pushButton

```
Source :      method pushButton(var eventInfo Event)
              var
              newView tableView
              choice string
              endVar
              choice="SAT"
              choice.view("Enter SAT or UNS to view the list of ships")
              switch
              case choice="SAT": newView.open("oppeSum")
              case choice="UNS": newView.open("oppeSum2")
              otherwise:
              msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
              return
              endSwitch
              endmethod
```

Object : #Page2.#Box22.#Button32

MethodName : pushButton

```
Source :      method pushButton(var eventInfo Event)
              var
              tc tCursor
              tbl table
              tbl1 table
              numberOfSatsOppe Number
              totalNumberOfOppe Number
              satPercentageOppe Number
              myQuery Query
              myQuery1 Query
              myQuery2 Query
              examDate1 Date
              examDate2 Date
              propType String
              examType String
              endVar
              doDefault
              examDate1=date("01/01/00")
              examDate2=date("12/31/99")
              propType="GT"
              examDate1.view("Enter start date (I.E. 01/01/95)")
              examDate2.view("Enter stop date (I.E. 01/01/95)")
              propType.view("Enter prop type (GT/STM/DSL)")
              switch
              case propType="GT":
              case propType="STM":
              case propType="DSL":
              case propType="ALL":
              otherwise:
              msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL or ALL only!")
              return
              endSwitch
```

```
if propType="ALL" then
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =LOE | Check =SAT OR
=EXC OR =GOOD|
```

```
SHIP.DB | ShipName | PropType |
|_EG01 | Check|
```

```
EndQuery
```

```
myQuery1=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =LOE | Check =SAT OR
=UNS OR =GOOD OR =EXC|
```

```
SHIP.DB | ShipName | PropType |
|_EG01 | Check|
```

```
EndQuery
```

```
myQuery2=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =LOE | Check =UNS |
```

```
SHIP.DB | ShipName | PropType |
|_EG01 | Check|
```

```
EndQuery
else
```

```
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =LOE | Check =SAT OR
=EXC OR =GOOD|
```



```
SHIP.DB | ShipName | PropType |
      |_EG01 | Check =~propType|
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
      | Check _EG01 | Check >=~examDate1, <=~examDate2|Check =LOE | Check =SAT OR
=UNS OR =GOOD OR =EXC|
```

```
SHIP.DB | ShipName | PropType |
      |_EG01 | Check =~propType|
```

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
      | Check _EG01 | Check >=~examDate1, <=~examDate2|Check =LOE | Check =UNS |
```

```
SHIP.DB | ShipName | PropType |
      |_EG01 | Check =~propType|
```

EndQuery

endif

```
empty("OppeSum")
empty("OppeSum1")
empty("OppeSum2")
executeQBE(myQuery, "OppeSum.db")
executeQBE(myQuery1, "OppeSum1.db")
executeQBE(myQuery2, "OppeSum2.db")
tbl.attach("OppeSum")
tbl1.attach("OppeSum1")
numberOfSatsOppe=tbl.cCount("OverallFinding")
msgInfo("LOE","The total number of sats are "
      +strVal(NumberOfSatsOppe))
TotalNumberOfOppe=tbl1.cCount("OverallFinding")
if TotalNumberOfOppe <> 0 then
  msgInfo("LOE","The total number is "
        +strVal(totalNumberOfOppe))
  SatPercentageOppe=(numberOfSatsOppe/totalNumberOfOppe)*100
  msgInfo("LOE","The sat percentage is "
        +strVal(satPercentageOppe))
```

```

else
  msgStop("Problem","The total number of LOE's is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageOppe
tc("PropType")="LOE"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box22.PM_LIST_SHIPS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
  newView tableView
  choice string
endVar
choice="SAT"
choice.view("Enter SAT or UNS to view the list of ships")
switch
  case choice="SAT": newView.open("Program2")
  case choice="UNS": newView.open("Program3")
  otherwise:
    msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
    return
  endSwitch
endmethod

```

Object : #Page2.#Box22.OP_LIST_SHIPS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
  newView tableView
  choice string
endVar
choice="SAT"
choice.view("Enter SAT or UNS to view the list of ships")
switch
  case choice="SAT": newView.open("opSum")
  case choice="UNS": newView.open("opSum2")
  otherwise:
    msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
    return
  endSwitch
endmethod

```

Object : #Page2.#Box22.OPPE_LIST_SHIPS_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
  newView tableView
  choice string
endVar
choice="SAT"
choice.view("Enter SAT or UNS to view the list of ships")
switch
  case choice="SAT": newView.open("oppeSum")
  case choice="UNS": newView.open("oppeSum2")
  otherwise:
    msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
  return
endSwitch
endmethod
```

Object : #Page2.#Box22.PM_PERCENT_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
  tc tCursor
  tbl table
  tbl1 table
  numberOfSatsProgramGrades Number
  totalNumberOfProgramGrades Number
  satPercentageProgramGrades Number
  myQuery Query
  myQuery1 Query
  myQuery2 Query
  examDate1 Date
  examDate2 Date
  propType String
  examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
  case propType="GT":
  case propType="STM":
  case propType="DSL":
  case propType="ALL":
```

```

otherwise:
  msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
  return
endSwitch
examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Propulsion Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

if propType="ALL" then
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate
|ProgramManageGrade      |
      | Check_EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate
|ProgramManageGrade      |
      | Check_EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT OR =UNS|

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 |Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate
|ProgramManageGrade|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =UNS      |

```

```

EndQuery
else

```

```

myQuery=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate
|ProgramManageGrade|
      | Check _EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT|

```

```

EndQuery

```

```

myQuery1=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate
|ProgramManageGrade|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT OR =UNS|

```

```

EndQuery

```

```

myQuery2=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName  |PropType  |
      | _EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate
|ProgramManageGrade|
      | Check _EG02                | Check >=~examDate1, <=~examDate2|Check =UNS      |

```

```

EndQuery
endif

```

```

empty("Program1")
empty("Program2")
empty("Program3")
executeQBE(myQuery, "Program2.db")
executeQBE(myQuery1, "Program1.db")
executeQBE(myQuery2, "Program3.db")
tbl.attach("Program2")
tbl1.attach("Program1")
numberOfSatsProgramGrades=tbl.cCount("ProgramManageGrade")
msgInfo("Program Management","The total number of sats are "
      +strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("ProgramManageGrade")
if totalNumberOfProgramGrades <> 0 then
  msgInfo("Program Management","The total number of grades are "
      +strVal(totalNumberOfProgramGrades))
  SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
  msgInfo("Program Management","The sat percentage is "
      +strVal(satPercentageProgramGrades))
else
  msgStop("Problem","The total number of programGrades is 0, you cannot divide by 0!")
  return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc.("Percentage")=SatPercentageProgramGrades
tc.("PropType")="PGM"
tc.("examDate1")=examDate1
tc.("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box22.TRAIN_LIST_SHIPS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
choice string

```

endVar
choice="SAT"
choice.view("Enter SAT or UNS to view the list of ships")
switch
case choice="SAT": newView.open("train")
case choice="UNS": newView.open("train2")
otherwise:
msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
return
endSwitch
endmethod

```

Object : #Page2.#Box22.TRAINING_PERCENT_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsTrainGrades Number
totalNumberOfTrainGrades Number
satPercentageTrainGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
ExamType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (I.E. 01/01/95)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch
examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":

```

```

case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      |_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |
      |_EG02,_EG01 | Check |

```

```

TRAINING.DB | Trainin_Ship_ShipName_FK7 | TrainingID_ExamEndDate |
      | Check _EG02          | Check >=~examDate1, <=~examDate2|

```

```

TRAINING.DB | TrainingProgramGrade |
      | Check =SAT OR =GOOD OR =EXC|

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      |_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |
      |_EG02,_EG01 | Check |

```

```

TRAINING.DB | Trainin_Ship_ShipName_FK7 | TrainingID_ExamEndDate |
      | Check _EG02          | Check >=~examDate1, <=~examDate2|

```

```

TRAINING.DB | TrainingProgramGrade |
      | Check          |

```

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      |_EG01          | Check =~examType|

```


SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

TRAINING.DB | Trainin_Ship_ShipName_FK7 | TrainingID_ExamEndDate |
| Check _EG02 | Check >=~examDate1, <=~examDate2|

TRAINING.DB | TrainingProgramGrade |
| Check =UNS |

EndQuery
else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG01 | Check =~examType|

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType |

TRAINING.DB | Trainin_Ship_ShipName_FK7 | TrainingID_ExamEndDate |
| Check _EG02 | Check >=~examDate1, <=~examDate2|

TRAINING.DB | TrainingProgramGrade |
| Check =SAT OR =GOOD OR =EXC|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|

TRAINING.DB | Trainin_Ship_ShipName_FK7 | TrainingID_ExamEndDate |
| Check _EG02 | Check >=~examDate1, <=~examDate2|

TRAINING.DB | TrainingProgramGrade |
| Check |

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check =~propType|

TRAINING.DB | Trainin_Ship_ShipName_FK7 | TrainingID_ExamEndDate |
| Check _EG02 | Check >=~examDate1, <=~examDate2|

TRAINING.DB | TrainingProgramGrade |
| Check =UNS |

EndQuery
endif

```
empty("Train")
empty("Train1")
empty("Train2")
executeQBE(myQuery, "Train.db")
executeQBE(myQuery1, "Train1.db")
executeQBE(myQuery2, "Train2.db")
tbl.attach("Train")
tbl1.attach("Train1")
numberOfSatsTrainGrades=tbl.cCount("TrainingProgramGrade")
msgInfo("Training Program","The total number of sats are "
+strVal(NumberOfSatsTrainGrades))
TotalNumberOfTrainGrades=tbl1.cCount("TrainingProgramGrade")
if totalNumberOfTrainGrades <> 0 then
    msgInfo("Training Program","The total number of grades are "
+strVal(totalNumberOfTrainGrades))
    SatPercentageTrainGrades=(numberOfSatsTrainGrades/totalNumberOfTrainGrades)*100
    msgInfo("Training program","The sat percentage of is "
+strVal(satPercentageTrainGrades))
else
    msgStop("Problem","The total number of training grades is 0, you cannot divide by 0!")
    return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageTrainGrades
tc("PropType")="TRAIN"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box22.MATERIAL_LIST_SHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
 var
 newView tableView
 choice string
 endVar
 choice="SAT"
 choice.view("Enter SAT or UNS to view the list of ships")
 switch
 case choice="SAT": newView.open("Mat2")
 case choice="UNS": newView.open("Mat3")
 otherwise:
 msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
 return
 endSwitch
 endmethod

Object : #Page2.#Box22.MATERIAL_PERCENT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
 var
 tc tCursor
 tbl table
 tbl1 table
 numberOfSatsMatGrades Number
 totalNumberOfMatGrades Number
 satPercentageMatGrades Number
 myQuery Query
 myQuery1 Query
 myQuery2 Query
 examDate1 Date
 examDate2 Date
 PropType String
 examType String
 endVar

 doDefault
 examDate1=date("01/01/00")
 examDate2=date("12/31/99")
 propType="GT"
 examType="OPPE"
 examDate1.view("Enter start date (I.E. 01/01/95)")
 examDate2.view("Enter stop date (I.E. 01/01/95)")
 propType.view("Enter prop type (GT/STM/DSL)")
 switch
 case propType="GT":
 case propType="STM":
 case propType="DSL":
 case propType="ALL":
 otherwise:
 msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
 return
 endSwitch

```

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      |_EG01              | Check =~examType|

MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate      |MaterialGrade |
      | Check _EG02,_EG01   | Check >=~examDate1, <=~examDate2|Check =SAT OR =EXC OR
=GOOD|

SHIP.DB | ShipName | PropType      |
      |_EG02      | Check |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      |_EG01              | Check =~examType|

MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate      |MaterialGrade
|
      | Check _EG02,_EG01   | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=EXC OR =GOOD|

SHIP.DB | ShipName | PropType      |
      |_EG02      | Check |

```

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      |_EG01              | Check =~examType|

MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate      |MaterialGrade|

```

```

        | Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|Check =UNS |

SHIP.DB | ShipName | PropType |
        | _EG02 | Check |

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
        | _EG01 | Check =~examType|

MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate |MaterialGrade |
        | Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|Check =SAT OR =EXC OR
=GOOD|

SHIP.DB | ShipName | PropType |
        | _EG02 | Check =~propType|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
        | _EG01 | Check =~examType|

MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate |MaterialGrade |
        | Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=EXC OR =GOOD|

SHIP.DB | ShipName | PropType |
        | _EG02 | Check =~propType|

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
        | _EG01 | Check =~examType|

MATERIAL.DB | MatID_Ship_ShipName_FK4 | MatID_ExamEndDate |MaterialGrade |
        | Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|Check =UNS |

SHIP.DB | ShipName | PropType |

```

|_EG02 | Check =~propType|

EndQuery
endif

```
empty("Mat2.db")
empty("Mat1.db")
empty("Mat3.db")
executeQBE(myQuery, "Mat2.db")
executeQBE(myQuery1, "Mat1.db")
executeQBE(myQuery2, "Mat3.db")
tbl.attach("Mat2")
tbl1.attach("Mat1")
numberOfSatsMatGrades=tbl.cCount("MaterialGrade")
msgInfo("Material","The total number of sats are "
+strVal(NumberOfSatsMatGrades))
TotalNumberOfMatGrades=tbl1.cCount("MaterialGrade")
if totalNumberOfMatGrades <> 0 then
msgInfo("Material","The total number of grades are "
+strVal(totalNumberOfMatGrades))
SatPercentageMatGrades=(numberOfSatsMatGrades/totalNumberOfMatGrades)*100
msgInfo("Material","The sat percentage of is "
+strVal(satPercentageMatGrades))
else
msgStop("Problem","The total number of material grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageMatGrades
tc("PropType")="MAT"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box22.FIREFIGHT_PERCENT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsFireGrades Number
totalNumberOfFireGrades Number
satPercentageFireGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date

```

examDate2 Date
PropType String
examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
  msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
  return
endSwitch
examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

if propType="ALL" then
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      |_EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
      |_EG03,_EG01 | Check |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireFighting|_ExamEndDate |
      | Check _EG03 | Check >=~examDate1, <=~examDate2|

FIREFIGH.DB | FireFightingGrade |
      | Check =SAT OR =EXC OR =GOOD|

EndQuery

myQuery1=Query

```

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
|_EG03,_EG01 | Check |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireFightingI_ExamEndDate |
| Check _EG03 | Check >=~examDate1, <=~examDate2|

FIREFIGH.DB | FireFightingGrade |
| Check =SAT OR =UNS OR =EXC OR =GOOD|

EndQuery
myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
|_EG03,_EG01 | Check |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireFightingI_ExamEndDate |
| Check _EG03 | Check >=~examDate1, <=~examDate2|

FIREFIGH.DB | FireFightingGrade |
| Check =UNS|

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
|_EG03,_EG01 | Check =~propType |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireFightingI_ExamEndDate |
| Check _EG03 | Check >=~examDate1, <=~examDate2|

FIREFIGH.DB | FireFightingGrade |
| Check =SAT OR =EXC OR =GOOD|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
| _EG03, _EG01 | Check =~propType |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireFightingI_ExamEndDate |
| Check _EG03 | Check >=~examDate1, <=~examDate2|

FIREFIGH.DB | FireFightingGrade |
| Check =SAT OR =UNS OR =EXC OR =GOOD|

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG01 | Check =~examType |

SHIP.DB | ShipName | PropType |
| _EG03, _EG01 | Check =~propType |

FIREFIGH.DB | FireFig_Ship_ShipName_FK5 | FireFightingI_ExamEndDate |
| Check _EG03 | Check >=~examDate1, <=~examDate2|

FIREFIGH.DB | FireFightingGrade |
| Check =UNS|

EndQuery

endif

empty("Fire1")

empty("Fire2")

empty("Fire3")

executeQBE(myQuery, "Fire1.db")

executeQBE(myQuery1, "Fire2.db")

executeQBE(myQuery2, "Fire3.db")

tbl.attach("Fire1")

tbl1.attach("Fire2")

numberOfSatsFireGrades=tbl.cCount("FireFightingGrade")

msgInfo("Fire Fighting","The total number of sats are "

+strVal(NumberOfSatsFireGrades))

TotalNumberOfFireGrades=tbl1.cCount("FireFightingGrade")

if TotalNumberOfFireGrades <> 0 then

msgInfo("Fire Fighting","The total number of grades are "

+strVal(totalNumberOfFireGrades))

SatPercentageFireGrades=(numberOfSatsFireGrades/totalNumberOfFireGrades)*100

msgInfo("Fire Fighting","The sat percentage is "

```

        +strVal(satPercentageFireGrades))
    else
        msgStop("Problem","The total number of fire fighting grades is 0, you cannot divide by 0!")
        return
    endif
    tc.open("percent")
    TC.edit()
    tc.insertRecord()
    tc.("Percentage")=SatPercentageFireGrades
    tc.("PropType")="F/F"
    tc.("examDate1")=examDate1
    tc.("examDate2")=examDate1
    tc.endEdit()
endmethod

```

Object : #Page2.#Box22.FF_LIST_SHIPS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
    newView tableView
    choice string
endVar
choice="SAT"
choice.view("Enter SAT or UNS to view the list of ships")
switch
    case choice="SAT": newView.open("fire1")
    case choice="UNS": newView.open("fire3")
    otherwise:
        msgStop("Problem","The choices for Exam grade are SAT, or UNS only!")
        return
    endSwitch
endmethod

```

Object : #Page2.#Box22.OPERATION_PERCENT_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
    tc tCursor
    tbl table
    tbl1 table
    numberOfSatsOperationGrades Number
    totalNumberOfOperationGrades Number
    satPercentageOperationGrades Number
    myQuery Query
    myQuery1 Query
    myQuery2 Query
    examDate1 Date
    examDate2 Date
    PropType String
endVar

```

```

examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
doDefault
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      | _EG01 | Check =OPPE |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
      | Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | OperationGrade |
      | Check =SAT OR =GOOD OR =EXC|

SHIP.DB | ShipName | PropType |
      | _EG02 | Check |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      | _EG01 | Check =OPPE |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
      | Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | OperationGrade |
      | Check |

SHIP.DB | ShipName | PropType |
      | _EG02 | Check |

```

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =OPPE |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | OperationGrade |
| Check =UNS |

SHIP.DB | ShipName | PropType |
|_EG02 | Check |

EndQuery

else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =OPPE |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | OperationGrade |
| Check =SAT OR =GOOD OR =EXC|

SHIP.DB | ShipName | PropType |
|_EG02 | Check =~propType|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =OPPE |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | OperationGrade |
| Check |

SHIP.DB | ShipName | PropType |

```

|_EG02 | Check =~propType|

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =OPPE |

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate |
| Check _EG02, _EG01 | Check >=~examDate1, <=~examDate2|

OPERATIO.DB | OperationGrade |
| Check =UNS |

SHIP.DB | ShipName | PropType |
|_EG02 | Check =~propType|

EndQuery
endif

empty("OpSum")
empty("OpSum1")
empty("OpSum2")
executeQBE(myQuery, "OpSum.db")
executeQBE(myQuery1, "OpSum1.db")
executeQBE(myQuery2, "OpSum2.db")
tbl.attach("OpSum")
tbl1.attach("OpSum1")
numberOfSatsOperationGrades=tbl.cCount("OperationGrade")
msgInfo("Operations","The total number of sats are "
+strVal(NumberOfSatsOperationGrades))
TotalNumberOfOperationGrades=tbl1.cCount("OperationGrade")
if TotalNumberOfOperationGrades <> 0 then
msgInfo("Operations","The total number grades are "
+strVal(totalNumberOfOperationGrades))

SatPercentageOperationGrades=(numberOfSatsOperationGrades/totalNumberOfOperationGrades)*100
msgInfo("Operations","The sat percentage is "
+strVal(satPercentageOperationGrades))
else
msgStop("Problem","The total number of operation grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageOperationGrades
tc("PropType")="OPS"
tc("examDate1")=examDate1
tc("examDate2")=examDate2

```

```
tc.endEdit()
endmethod
```

Object : #Page2.#Box22.OPPE_PERCENTAGE_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tc tCursor
tbl table
tbl1 table
numberOfSatsOppe Number
totalNumberOfOppe Number
satPercentageOppe Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
propType String
examType String
endVar
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

if propType="ALL" then
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =SAT OR
=EXC OR =GOOD|
```

```
SHIP.DB | ShipName | PropType |
|_EG01 | Check |
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =SAT OR
=UNS OR =GOOD OR =EXC|

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =UNS |

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

EndQuery

else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =SAT OR
=EXC OR =GOOD|

SHIP.DB | ShipName | PropType |
|_EG01 | Check =~propType|

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =SAT OR
=UNS OR =GOOD OR =EXC|

```
SHIP.DB | ShipName | PropType |
      |_EG01 | Check =~propType|
```

EndQuery

myQuery2=Query

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
      | Check _EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =UNS |
```

```
SHIP.DB | ShipName | PropType |
      |_EG01 | Check =~propType|
```

EndQuery

endif

```
empty("OppeSum")
empty("OppeSum1")
empty("OppeSum2")
executeQBE(myQuery, "OppeSum.db")
executeQBE(myQuery1, "OppeSum1.db")
executeQBE(myQuery2, "OppeSum2.db")
tbl.attach("OppeSum")
tbl1.attach("OppeSum1")
numberOfSatsOppe=tbl.cCount("OverallFinding")
msgInfo("OPPE","The total number of sats are "
      +strVal(NumberOfSatsOppe))
TotalNumberOfOppe=tbl1.cCount("OverallFinding")
if TotalNumberOfOppe <> 0 then
  msgInfo("OPPE","The total number is "
      +strVal(totalNumberOfOppe))
  SatPercentageOppe=(numberOfSatsOppe/totalNumberOfOppe)*100
  msgInfo("OPPE","The sat percentage is "
      +strVal(satPercentageOppe))
else
  msgStop("Problem","The total number of OPPE's is 0, you cannot divide by 0!")
  return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageOppe
tc("PropType")="OPPE"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```


Object : PROGRAM_STATS_QUERY

MethodName : Const

Source : Const
ViewProgram1=301
ViewProgram2=302
ViewProgram3=303
ViewProgram4=304
ViewProgram5=305
ViewProgram6=306
ViewProgram7=307
ViewProgram8=308
ViewProgram9=309
ViewProgram10=310
ViewProgram11=311
ViewProgram12=312
ViewProgram13=313
ViewProgram14=314
PrintProgram1=315
PrintProgram2=316
PrintProgram3=317
PrintProgram4=318
PrintProgram5=319
PrintProgram6=320
PrintProgram7=321
PrintProgram8=322
PrintProgram9=323
PrintProgram10=324
PrintProgram11=325
PrintProgram12=326
PrintProgram13=327
PrintProgram14=328
PrintProgram15=329
endConst

Object : PROGRAM_STATS_QUERY

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
var
tc Tcursor
examMenu, View, Print, ReportMenu Menu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewProgramPop PopUpMenu
PrintProgramPop PopUpMenu
examtype PopupMenu
endVar
if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else

; This code executes only for the form.

```
PrintPop.addText("&Bearing &Records", "", PrintProgram1)
PrintPop.addText("&BWFW", "", PrintProgram2)
PrintPop.addText("&DETA", "", PrintProgram3)
PrintPop.addText("&D&JWTT", "", PrintProgram4)
PrintPop.addText("&Electrical Safety", "", PrintProgram5)
PrintPop.addText("&FOQM", "", PrintProgram6)
PrintPop.addText("&Hearing Conservation", "", PrintProgram7)
PrintPop.addText("&Le&gal Records", "", PrintProgram8)
PrintPop.addText("&LOQM", "", PrintProgram9)
PrintPop.addText("&MGTESR", "", PrintProgram10)
PrintPop.addText("&OLV", "", PrintProgram11)
PrintPop.addText("&O&perating Logs", "", PrintProgram12)
PrintPop.addText("&QA", "", PrintProgram13)
PrintPop.addText("&Tag Out", "", PrintProgram14)
PrintPop.addText("&Graph", "", PrintProgram15)
```

```
ViewPop.addText("&Bearing &Records", "", ViewProgram1)
ViewPop.addText("&BWFW", "", ViewProgram2)
ViewPop.addText("&DETA", "", ViewProgram3)
ViewPop.addText("&D&JWTT", "", ViewProgram4)
ViewPop.addText("&Electrical Safety", "", ViewProgram5)
ViewPop.addText("&FOQM", "", ViewProgram6)
ViewPop.addText("&Hearing Conservation", "", ViewProgram7)
ViewPop.addText("&Le&gal Records", "", ViewProgram8)
ViewPop.addText("&LOQM", "", ViewProgram9)
ViewPop.addText("&MGTESR", "", ViewProgram10)
ViewPop.addText("&OLV", "", ViewProgram11)
ViewPop.addText("&O&perating Logs", "", ViewProgram12)
ViewPop.addText("&QA", "", ViewProgram13)
ViewPop.addText("&Tag Out", "", ViewProgram14)
```

```
examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endif
endmethod
```

Object : PROGRAM_STATS_QUERY

MethodName : menuAction

Source : method menuAction(var eventInfo MenuEvent)
var

```

myRep Report
reply String
choiceId SmallInt
m menu
endVar
choiceId=eventInfo.id()

if eventInfo.isPreFilter()
    then
        ; This code executes for each object on the form.

    else
        ; This code executes only for the form.

Switch
    case eventInfo.menuChoice() = "&Help":
case eventInfo.menuChoice() = "&Quit":
    reply=msgQuestion("Quit", "Are you sure you want to leave this form?")
    If reply = "Yes" then
        close()
    else
        return
    endif
endSwitch
Switch
case choiceId =ViewProgram1:
    myRep.open("Bearing")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceId =ViewProgram2:
    myRep.open("BFWF")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceId =ViewProgram3:
    myRep.open("DETA")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceId =ViewProgram4:
    myRep.open("DJWTT")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceId =ViewProgram5:
    myRep.open("Electric")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceId =ViewProgram6:
    myRep.open("FOQM")
    hideSpeedBar()
    m.addText("")
    m.show()

```

```

case choiceld =ViewProgram7:
    myRep.open("Hearing")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram8:
    myRep.open("LegalRec")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram9:
    myRep.open("LOQM")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram10:
    myRep.open("MGTESR")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram11:
    myRep.open("OLV")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram12:
    myRep.open("OPLOGS")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram13:
    myRep.open("QA")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =ViewProgram14:
    myRep.open("TagOut")
    hideSpeedBar()
    m.addText("")
    m.show()
case choiceld =PrintProgram1:
    myRep.print("Bearing")
case choiceld =PrintProgram2:
    myRep.print("BWFW")
case choiceld =PrintProgram3:
    myRep.print("DETA")
case choiceld =PrintProgram4:
    myRep.print("DJWTT")
case choiceld =PrintProgram5:
    myRep.print("Electric")
case choiceld =PrintProgram6:
    myRep.print("FOQM")
case choiceld =PrintProgram7:
    myRep.print("Hearing")
case choiceld =PrintProgram8:

```

```

        myRep.print("LegalRec")
    case choiceld =PrintProgram9:
        myRep.print("LOQM")
    case choiceld =PrintProgram10:
        myRep.print("MGTESR")
    case choiceld =PrintProgram11:
        myRep.print("OLV")
    case choiceld =PrintProgram12:
        myRep.print("OPLOGS")
    case choiceld =PrintProgram13:
        myRep.print("QA")
    case choiceld =PrintProgram14:
        myRep.print("TagOut")
    case choiceld =PrintProgram15:
        myRep.print("ProgStat")
endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

Source : method setFocus(var eventInfo Event)

```

var
    examMenu, View, Print, ReportMenu Menu
    PrintPop PopUpMenu
    ViewPop PopUpMenu
    ViewProgramPop PopUpMenu
    PrintProgramPop PopUpMenu
    examtype PopUpMenu
endVar

PrintPop.addText("Bearing &Records","",PrintProgram1)
PrintPop.addText("&BFWF","",PrintProgram2)
PrintPop.addText("&DETA","",PrintProgram3)
PrintPop.addText("D&JWTT","",PrintProgram4)
PrintPop.addText("&Electrical Safety","",PrintProgram5)
PrintPop.addText("&FOQM","",PrintProgram6)
PrintPop.addText("&Hearing Conservation","",PrintProgram7)
PrintPop.addText("Le&gal Records","",PrintProgram8)
PrintPop.addText("&LOQM","",PrintProgram9)
PrintPop.addText("&MGTESR","",PrintProgram10)
PrintPop.addText("&OLV","",PrintProgram11)
PrintPop.addText("O&perating Logs","",PrintProgram12)
PrintPop.addText("&QA","",PrintProgram13)
PrintPop.addText("&Tag Out","",PrintProgram14)
PrintPop.addText("&Graph","",PrintProgram15)

ViewPop.addText("Bearing &Records","",ViewProgram1)
ViewPop.addText("&BFWF","",ViewProgram2)
ViewPop.addText("&DETA","",ViewProgram3)
ViewPop.addText("D&JWTT","",ViewProgram4)
ViewPop.addText("&Electrical Safety","",ViewProgram5)

```

```

ViewPop.addText("&FOQM","",ViewProgram6)
ViewPop.addText("&Hearing Conservation","",ViewProgram7)
ViewPop.addText("Le&gal Records","",ViewProgram8)
ViewPop.addText("&LOQM","",ViewProgram9)
ViewPop.addText("&MGTESR","",ViewProgram10)
ViewPop.addText("&OLV","",ViewProgram11)
ViewPop.addText("O&perating Logs","",ViewProgram12)
ViewPop.addText("&QA","",ViewProgram13)
ViewPop.addText("&Tag Out","",ViewProgram14)

```

```

examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
endmethod

```

Object : #Page2.#Box3.RESET_GRAPH_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc Tcursor
endVar
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.HEAT_STRESS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

```

```

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName    |PropType      |
      | _EG02, _EG01 |Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HeatStress      |
      | Check _EG02                | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT|
EndQuery

```

```

myQuery1=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName    |PropType      |
      | _EG02, _EG01 |Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HeatStress
|
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT OR =UNS|
EndQuery

```

```

else
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
|_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName   |PropType      |
|_EG02,_EG01 |Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HeatStress      |
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT|
EndQuery

```

```

myQuery1=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
|_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName   |PropType      |
|_EG02,_EG01 |Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HeatStress
|
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT OR =UNS|
EndQuery
endif

```

```

empty("HEATSTRS")
empty("HEATSTR1")
executeQBE(myQuery, "HEATSTRS.db")
executeQBE(myQuery1, "HEATSTR1.db")
tbl.attach("HEATSTRS")
tbl1.attach("HEATSTR1")
numberOfSatsProgramGrades=tbl.cCount("HeatStress")
msgInfo("Heat Stress","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("HeatStress")
if totalNumberOfProgramGrades <> 0 then
msgInfo("HeatStress","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("Heat Stress","The sat percentage is "
+strVal(satPercentageProgramGrades))

```



```

else
    msgStop("Problem","The total number of Heat Stress grades is 0, you cannot divide by 0!")
    return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="HEAT"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.BEARING_RECORDS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
    tc tCursor
    tbl table
    tbl1 table
    numberOfSatsProgramGrades Number
    totalNumberOfProgramGrades Number
    satPercentageProgramGrades Number
    myQuery Query
    myQuery1 Query
    myQuery2 Query
    examDate1 Date
    examDate2 Date
    PropType String
    examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
    msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL or ALL only!")
    return
endSwitch

```

```

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

if propType="ALL" then
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | BearingRecsGrade
      | Check _EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | BearingRecsGrade|
      | Check _EG02          | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC      |

EndQuery
else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

```

```
SHIP.DB | ShipName | PropType |
      | _EG02, _EG01 | Check = ~propType |
```

```
PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | BearingRecsGrade |
      | Check _EG02 | Check >= ~examDate1, <= ~examDate2 | Check = EXC OR = GOOD OR
= SAT |
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
      | _EG01 | Check = ~examType |
```

```
SHIP.DB | ShipName | PropType |
      | _EG02, _EG01 | Check = ~propType |
```

```
PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | BearingRecsGrade |
      | Check _EG02 | Check >= ~examDate1, <= ~examDate2 | Check = SAT OR = UNS OR
= GOOD OR = EXC |
```

EndQuery

endif

```
empty("BEARREC")
empty("BEARREC1")
executeQBE(myQuery, "BEARREC.db")
executeQBE(myQuery1, "BEARREC1.db")
tbl.attach("BEARREC")
tbl1.attach("BEARREC1")
numberOfSatsProgramGrades=tbl.cCount("BearingRecsGrade")
msgInfo("Bearing Records", "The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("BearingRecsGrade")
if totalNumberOfProgramGrades <> 0 then
  msgInfo("Bearing Records", "The total number of grades are "
+strVal(totalNumberOfProgramGrades))
  SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
  msgInfo("Bearing Records", "The sat percentage is "
+strVal(satPercentageProgramGrades))
else
  msgStop("Problem", "The total number of Bearing Records Grades is 0, you cannot divide by 0!")
  return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="BRG"
tc("examDate1")=examDate1
```

```

tc.("examDate2")=examDate2
tc.("propType1")=examType
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.LEGAL_RECORDS_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

```

```
if propType="ALL" then  
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |  
      | _EG01                | Check =~examType|
```

```
SHIP.DB | ShipName   | PropType      |  
      | _EG02, _EG01 | Check |
```

```
PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | LegalRecsGrade  
|  
      | Check _EG02                | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR  
=SAT |
```

```
EndQuery
```

```
myQuery1=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |  
      | _EG01                | Check =~examType|
```

```
SHIP.DB | ShipName   | PropType      |  
      | _EG02, _EG01 | Check |
```

```
PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | LegalRecsGrade |  
      | Check _EG02                | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR  
=GOOD OR =EXC      |
```

```
EndQuery
```

```
else  
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |  
      | _EG01                | Check =~examType|
```

```
SHIP.DB | ShipName   | PropType      |  
      | _EG02, _EG01 | Check =~propType|
```

```
PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | LegalRecsGrade  
|  
      | Check _EG02                | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR  
=SAT |
```

```
EndQuery
```

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName | PropType |
|_EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | LegalRecsGrade |
| Check _EG02 | Check >=~examDate1, <=~examDate2 | Check =SAT OR =UNS OR
=GOOD OR =EXC |

EndQuery
endif

empty("LEGALREC")
empty("LGALREC1")
executeQBE(myQuery, "LEGALREC.db")
executeQBE(myQuery1, "LGALREC1.db")
tbl.attach("LEGALREC")
tbl1.attach("LGALREC1")
numberOfSatsProgramGrades=tbl.cCount("LegalRecsGrade")
msgInfo("Legal Records", "The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("LegalRecsGrade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("Legal Records", "The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("Legal Records", "The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem", "The total number of Legal Records Grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="LGL"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

Object : #Page2.#Box3.BWFW_BUTTON

MethodName : pushButton

```

Source :      method pushButton(var eventInfo Event)
              var
              tc tCursor
              tbl table
              tbl1 table
              numberOfSatsProgramGrades Number
              totalNumberOfProgramGrades Number
              satPercentageProgramGrades Number
              myQuery Query
              myQuery1 Query
              myQuery2 Query
              examDate1 Date
              examDate2 Date
              PropType String
              examType String
              endVar

              doDefault
              examDate1=date("01/01/00")
              examDate2=date("12/31/99")
              propType="GT"
              examType="OPPE"
              examDate1.view("Enter start date (I.E. 01/01/95)")
              examDate2.view("Enter stop date (I.E. 01/01/95)")
              propType.view("Enter prop type (GT/STM/DSL)")
              switch
              case propType="GT":
              case propType="STM":
              case propType="DSL":
              case propType="ALL":
              otherwise:
              msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
              return
              endSwitch

              examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
              switch
              case examType="OPPE":
              case examType="REOPPE":
              case examType="LOE":
              case examType="RELOE":
              otherwise:
              msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
              return
              endSwitch

              if propType="ALL" then
              myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                      | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |

```

```

|_EG02,_EG01|Check|
PROGRAMM.DB|PM_ID_Ship_ShipName_FK6|PM_ID_ExamEndDate|BFWF_Grade
|
|Check_EG02|Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT|

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB|ExamID_Ship_ShipName_FK2|ExamType|
|_EG01|Check =~examType|

```

```

SHIP.DB|ShipName|PropType|
|_EG02,_EG01|Check|

```

```

PROGRAMM.DB|PM_ID_Ship_ShipName_FK6|PM_ID_ExamEndDate|BFWF_Grade|
|Check_EG02|Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC|

```

EndQuery
else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB|ExamID_Ship_ShipName_FK2|ExamType|
|_EG01|Check =~examType|

```

```

SHIP.DB|ShipName|PropType|
|_EG02,_EG01|Check =~propType|

```

```

PROGRAMM.DB|PM_ID_Ship_ShipName_FK6|PM_ID_ExamEndDate|BFWF_Grade
|
|Check_EG02|Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT|

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB|ExamID_Ship_ShipName_FK2|ExamType|
|_EG01|Check =~examType|

```

```

SHIP.DB|ShipName|PropType|
|_EG02,_EG01|Check =~propType|

```



```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |BFWF Grade|
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC|

```

```

EndQuery
endif

```

```

empty("BFWF")
empty("BFWF1")
executeQBE(myQuery, "BFWF.db")
executeQBE(myQuery1, "BFWF1.db")
tbl.attach("BFWF")
tbl1.attach("BFWF1")
numberOfSatsProgramGrades=tbl.cCount("BFWF_Grade")
msgInfo("BFWF","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("BFWF_Grade")
if totalNumberOfProgramGrades <> 0 then
    msgInfo("BFWF","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
    SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
    msgInfo("BFWF","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
    msgStop("Problem","The total number of BFWF Grades is 0, you cannot divide by 0!")
    return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="BFWF"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.LOQM_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query

```

```

myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

```

```

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
  msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
  return
endSwitch

```

```

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName  | PropType      |
      | _EG02, _EG01 | Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | LOQM_Grade
|
| Check _EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

```

```

EndQuery

```

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | LOQM_Grade |
| Check_EG02 | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC|

EndQuery

else

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | LOQM_Grade
|
| Check_EG02 | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | LOQM_Grade |
| Check_EG02 | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC|

EndQuery

endif

empty("LOQM")

```

empty("LOQM1")
executeQBE(myQuery, "LOQM.db")
executeQBE(myQuery1, "LOQM1.db")
tbl.attach("LOQM")
tbl1.attach("LOQM1")
numberOfSatsProgramGrades=tbl.cCount("LOQM_Grade")
msgInfo("LOQM","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("LOQM_Grade")
if totalNumberOfProgramGrades <> 0 then
    msgInfo("LOQM","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
    SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
    msgInfo("LOQM","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
    msgStop("Problem","The total number of LOQM Grades is 0, you cannot divide by 0!")
    return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="LOQM"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.DETA_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
    tc tCursor
    tbl table
    tbl1 table
    numberOfSatsProgramGrades Number
    totalNumberOfProgramGrades Number
    satPercentageProgramGrades Number
    myQuery Query
    myQuery1 Query
    myQuery2 Query
    examDate1 Date
    examDate2 Date
    PropType String
    examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")

```

```

propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
  msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
  return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

```

```

SHIP.DB | ShipName | PropType      |
      | _EG02, _EG01 | Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | DETA_Grade
|
| Check _EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
= SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

```

```

SHIP.DB | ShipName | PropType      |
      | _EG02, _EG01 | Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |DETA_Grade |
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

EndQuery

```

else
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

```

```

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |DETA_Grade
|
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

```

```

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |DETA_Grade |
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

```

EndQuery
endif

```

```

empty("DETA")
empty("DETA1")
executeQBE(myQuery, "DETA.db")
executeQBE(myQuery1, "DETA1.db")
tbl.attach("DETA")
tbl1.attach("DETA1")
numberOfSatsProgramGrades=tbl.cCount("DETA_Grade")
msgInfo("DETA","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("DETA_Grade")
if totalNumberOfProgramGrades <> 0 then

```

```

msgInfo("DETA","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("DETA","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of DETA Grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="DETA"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.MGTESR_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":

```

```

otherwise:
  msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, ALL only!")
  return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

if propType="ALL" then
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |MGTESR_Grade
|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |MGTESR_Grade|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC|

EndQuery

else
myQuery=Query

```


ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |MGTESR_Grade
|
| Check _EG02 | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |MGTESR_Grade|
| Check _EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC|

EndQuery

endif

```
empty("MGTESR")
empty("MGTESR1")
executeQBE(myQuery, "MGTESR.db")
executeQBE(myQuery1, "MGTESR1.db")
tbl.attach("MGTESR")
tbl1.attach("MGTESR1")
numberOfSatsProgramGrades=tbl.cCount("MGTESR_Grade")
msgInfo("MGTESR","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("MGTESR_Grade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("MGTESR","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("MGTESR","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of MGTESR Grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
```

```

TC.edit()
tc.insertRecord()
tc.("Percentage")=SatPercentageProgramGrades
tc.("PropType")="MGTESR"
tc.("PropType1")=examType
tc.("examDate1")=examDate1
tc.("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.HEARING_CONSERVE_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

```

```

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem", "The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

```

```

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":

```

```

case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HearingConsGrade
|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT|
EndQuery

```

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HearingConsGrade
|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT OR =UNS|
EndQuery

```

```

else
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |HearingConsGrade
|
      | Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT|
EndQuery

```

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |HearingConsGrade
|
| Check _EG02 | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT OR =UNS|
EndQuery
endif

```
empty("HEARCON")
empty("HEARCON1")
executeQBE(myQuery, "HEARCON.db")
executeQBE(myQuery1, "HEARCON1.db")
tbl.attach("HEARCON")
tbl1.attach("HEARCON1")
numberOfSatsProgramGrades=tbl.cCount("HearingConsGrade")
msgInfo("Hearing Conservation","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("HearingConsGrade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("Hearing Conservation","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("Hearing Conservation","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of Hearing Conservation grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="HC"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box3.TAGOUT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
  tc tCursor
  tbl table
  tbl1 table
  numberOfSatsProgramGrades Number
  totalNumberOfProgramGrades Number
  satPercentageProgramGrades Number
  myQuery Query
  myQuery1 Query
  myQuery2 Query
  examDate1 Date
  examDate2 Date
  PropType String
  examType String
endVar

doDefault
  examDate1=date("01/01/00")
  examDate2=date("12/31/99")
  propType="GT"
  examType="OPPE"
  examDate1.view("Enter start date (I.E. 01/01/95)")
  examDate2.view("Enter stop date (I.E. 01/01/95)")
  propType.view("Enter prop type (GT/STM/DSL)")
  switch
    case propType="GT":
    case propType="STM":
    case propType="DSL":
    case propType="ALL":
    otherwise:
      msgStop("Problem","The choices for Propulsion Type are GT, STM, or DSL only!")
      return
  endSwitch

  examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
  switch
    case examType="OPPE":
    case examType="REOPPE":
    case examType="LOE":
    case examType="RELOE":
    otherwise:
      msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
      return
  endSwitch

  if propType="ALL" then
    myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      |_EG01                | Check =~examType|

SHIP.DB | ShipName |PropType      |
      |_EG02,_EG01 |Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |TagoutGrade
|
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
|_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName   |PropType      |
|_EG02,_EG01 |Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |TagoutGrade|
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC      |

```

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
|_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName   |PropType      |
|_EG02,_EG01 |Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |TagoutGrade
|
| Check_EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
|_EG01          | Check =~examType|

```

```

SHIP.DB | ShipName   |PropType      |
|_EG02,_EG01 |Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | TagoutGrade|
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

```

EndQuery
endif

```

```

empty("TAGOUT")
empty("TAGOUT")
executeQBE(myQuery, "TAGOUT.db")
executeQBE(myQuery1, "TAGOUT1.db")
tbl.attach("TAGOUT")
tbl1.attach("TAGOUT1")
numberOfSatsProgramGrades=tbl.cCount("TagOutGrade")
msgInfo("Tag Out","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("TagOutGrade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("Tag Out","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("Tag Out","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of Tag Out grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="TAGOUT"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.FOQM_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query

```

```

examDate1 Date
examDate2 Date
PropType String
examType String
endVar

```

```

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

```

```

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName | PropType      |
      | _EG02, _EG01 | Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | FOQM_Grade
|
| Check _EG02                | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |FOQM_Grade|
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC|

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |FOQM_Grade
|
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |FOQM_Grade|
| Check_EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC|

EndQuery
endif

empty("FOQM")
empty("FOQM1")

```

executeQBE(myQuery, "FOQM.db")
executeQBE(myQuery1, "FOQM1.db")
tbl.attach("FOQM")
tbl1.attach("FOQM1")
numberOfSatsProgramGrades=tbl.cCount("FOQM_Grade")
msgInfo("FOQM","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("FOQM_Grade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("FOQM","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("FOQM","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of FOQM Grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="FOQM"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.QA_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"

```

```

examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
  msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
  return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

if propType="ALL" then
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 |Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |QA_Grade
|
| Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 |Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |QA_Grade|

```

```

| Check _EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

EndQuery

```

else
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG01 | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |QA_Grade
|
| Check _EG02 | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG01 | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate |QA_Grade|
| Check _EG02 | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

```

EndQuery
endif

```

```

empty("QA")
empty("QA1")
executeQBE(myQuery, "QA.db")
executeQBE(myQuery1, "QA1.db")
tbl.attach("QA")
tbl1.attach("QA1")
numberOfSatsProgramGrades=tbl.cCount("QA_Grade")
msgInfo("Quality Assurance","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("QA_Grade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("Quality Assurance","The total number of grades are "

```

```

        +strVal(totalNumberOfProgramGrades))
    SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
    msgInfo("Quality Assurance","The sat percentage is "
        +strVal(satPercentageProgramGrades))
    else
        msgStop("Problem","The total number of Quality Assurance grades is 0, you cannot divide by 0!")
    return
    endif
    tc.open("percent")
    TC.edit()
    tc.insertRecord()
    tc("Percentage")=SatPercentageProgramGrades
    tc("PropType")="QA"
    tc("PropType1")=examType
    tc("examDate1")=examDate1
    tc("examDate2")=examDate2
    tc.endEdit()
    endmethod

```

Object : #Page2.#Box3.ELECTRICAL_SAFETY_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
    tc tCursor
    tbl table
    tbl1 table
    numberOfSatsProgramGrades Number
    totalNumberOfProgramGrades Number
    satPercentageProgramGrades Number
    myQuery Query
    myQuery1 Query
    myQuery2 Query
    examDate1 Date
    examDate2 Date
    PropType String
    examType String
endVar

doDefault
    examDate1=date("01/01/00")
    examDate2=date("12/31/99")
    propType="GT"
    examType="OPPE"
    examDate1.view("Enter start date (I.E. 01/01/95)")
    examDate2.view("Enter stop date (I.E. 01/01/95)")
    propType.view("Enter prop type (GT/STM/DSL)")
    switch
        case propType="GT":
        case propType="STM":
        case propType="DSL":
        case propType="ALL":
        otherwise:

```

```

msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

if propType="ALL" then
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | ElectSafetyGrade
|
| Check _EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | ElectSafetyGrade|
| Check _EG02          | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC      |

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | ElectSafetyGrade
|
      | Check _EG02          | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01          | Check =~examType|

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | ElectSafetyGrade|
      | Check _EG02          | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC      |

```

EndQuery
endif

```

empty("ELECSAF")
empty("ELECSAF1")
executeQBE(myQuery, "ELECSAF.db")
executeQBE(myQuery1, "ELECSAF1.db")
tbl.attach("ELECSAF")
tbl1.attach("ELECSAF1")
numberOfSatsProgramGrades=tbl.cCount("ElectSafetyGrade")
msgInfo("Electrical Safety","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("ElectSafetyGrade")
if totalNumberOfProgramGrades <> 0 then
  msgInfo("Electrical Safety","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
  SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
  msgInfo("Electrical Safety","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
  msgStop("Problem","The total number of Electrical Safety Grades is 0, you cannot divide by 0!")
  return
endif
tc.open("percent")
TC.edit()

```

```

tc.insertRecord()
tc."Percentage"=SatPercentageProgramGrades
tc."PropType"="ES"
tc."PropType1"=examType
tc."examDate1"=examDate1
tc."examDate2"=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.OPERATING_LOGS_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number
satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":

```



```

otherwise:
  msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
  return
endSwitch

```

```

if propType="ALL" then
  myQuery=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 | Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |OpLogsGrade
|
| Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR
=SAT |

```

```

EndQuery

```

```

myQuery1=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 | Check |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |OpLogsGrade |
| Check _EG02          | Check >=~examDate1, <=~examDate2|Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

```

EndQuery

```

```

else
  myQuery=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName  |PropType      |
      | _EG02, _EG01 | Check =~propType|

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      |OpLogsGrade
|
| Check _EG02          | Check >=~examDate1, <=~examDate2|Check =EXC OR =GOOD OR

```

=SAT |

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG01 | Check =~examType|

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | OpLogsGrade |
| Check _EG02 | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC |

EndQuery

endif

```
empty("OPLOGS")
empty("OPLOGS1")
executeQBE(myQuery, "OPLOGS.db")
executeQBE(myQuery1, "OPLOGS1.db")
tbl.attach("OPLOGS")
tbl1.attach("OPLOGS1")
numberOfSatsProgramGrades=tbl.cCount("OpLogsGrade")
msgInfo("Operation Logs","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("OpLogsGrade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("Operation Logs","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("Operation Logs","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of Operation Logs Grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="OPLOGS"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod
```

Object : #Page2.#Box3.DJWTT_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
  tc tCursor
  tbl table
  tbl1 table
  numberOfSatsProgramGrades Number
  totalNumberOfProgramGrades Number
  satPercentageProgramGrades Number
  myQuery Query
  myQuery1 Query
  myQuery2 Query
  examDate1 Date
  examDate2 Date
  PropType String
  examType String
endVar

doDefault
  examDate1=date("01/01/00")
  examDate2=date("12/31/99")
  propType="GT"
  examType="OPPE"
  examDate1.view("Enter start date (I.E. 01/01/95)")
  examDate2.view("Enter stop date (I.E. 01/01/95)")
  propType.view("Enter prop type (GT/STM/DSL)")
  switch
    case propType="GT":
    case propType="STM":
    case propType="DSL":
    case propType="ALL":
    otherwise:
      msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, ALL only!")
      return
  endSwitch

  examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
  switch
    case examType="OPPE":
    case examType="REOPPE":
    case examType="LOE":
    case examType="RELOE":
    otherwise:
      msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
      return
  endSwitch

  if propType="ALL" then
    myQuery=Query

  ANSWER: :PRIV:ANSWER.DB
```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | DJWTT_Grade
|
      | Check _EG02                | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check |

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | DJWTT_Grade |
      | Check _EG02                | Check >=~examDate1, <=~examDate2| Check =SAT OR =UNS OR
=GOOD OR =EXC |

```

EndQuery

else
myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

SHIP.DB | ShipName   | PropType      |
      | _EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | DJWTT_Grade
|
      | Check _EG02                | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |

```

```

|_EG01          | Check =~examType|

SHIP.DB | ShipName | PropType |
|_EG02, _EG01 | Check =~propType|

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | DJWTT_Grade |
| Check_EG02 | Check >=~examDate1, <=~examDate2 | Check =SAT OR =UNS OR
=GOOD OR =EXC |

EndQuery
endif
empty("DJWTT")
empty("DJWTT1")
executeQBE(myQuery, "DJWTT.db")
executeQBE(myQuery1, "DJWTT1.db")
tbl.attach("DJWTT")
tbl1.attach("DJWTT1")
numberOfSatsProgramGrades=tbl.cCount("DJWTT_Grade")
msgInfo("DJWTT","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("DJWTT_Grade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("DJWTT","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("DJWTT","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of DJWTT Grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="DJWTT"
tc("PropType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page2.#Box3.OLV_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsProgramGrades Number
totalNumberOfProgramGrades Number

```

satPercentageProgramGrades Number
myQuery Query
myQuery1 Query
myQuery2 Query
examDate1 Date
examDate2 Date
PropType String
examType String
endVar

```

```

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examType="OPPE"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")

```

```

examType.view("Enter (OPPE/REOPPE/LOE/RELOE)")
switch
case examType="OPPE":
case examType="REOPPE":
case examType="LOE":
case examType="RELOE":
otherwise:
msgStop("Problem","The choices for Exam Type are OPPE, REOPPE, LOE, or RELOE only!")
return
endSwitch

```

```
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |
      | _EG02, _EG01 | Check =STM |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate      | OLV_Grade
|
| Check _EG02                | Check >=~examDate1, <=~examDate2| Check =EXC OR =GOOD OR
=SAT |

```

```
EndQuery
```

```
myQuery1=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG01                | Check =~examType|

```

```

SHIP.DB | ShipName | PropType |
      | _EG02, _EG01 | Check =STM |

```

```

PROGRAMM.DB | PM_ID_Ship_ShipName_FK6 | PM_ID_ExamEndDate | OLV_Grade |
| Check_EG02 | Check >= ~examDate1, <= ~examDate2 | Check = SAT OR = UNS OR
= GOOD OR = EXC |

```

EndQuery

```

empty("OLV")
empty("OLV1")
executeQBE(myQuery, "OLV.db")
executeQBE(myQuery1, "OLV1.db")
tbl.attach("OLV")
tbl1.attach("OLV1")
numberOfSatsProgramGrades=tbl.cCount("OLV_Grade")
msgInfo("Online Verification","The total number of sats are "
+strVal(NumberOfSatsProgramGrades))
TotalNumberOfProgramGrades=tbl1.cCount("OLV_Grade")
if totalNumberOfProgramGrades <> 0 then
msgInfo("Online Verification","The total number of grades are "
+strVal(totalNumberOfProgramGrades))
SatPercentageProgramGrades=(numberOfSatsProgramGrades/totalNumberOfProgramGrades)*100
msgInfo("Online Verification","The sat percentage is "
+strVal(satPercentageProgramGrades))
else
msgStop("Problem","The total number of Online Verification grades is 0, you cannot divide by 0!")
return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageProgramGrades
tc("PropType")="OLV"
tc("propType1")=examType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : OPPE_LOE_MONTHLY_QUERY

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    examMenu.addText("&Quit")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    endif
endmethod
```

Object : OPPE_LOE_MONTHLY_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Help":

      case eventInfo.menuChoice() = "&Quit":
        reply=msgQuestion("Quit", "Are you sure you want to leave this form?")
        If reply = "Yes" then
          close()
        else
          return
        endif
      endSwitch
    endif
endmethod
```


Object : #Page2.#Button3

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl tableView
myQuery Query
examDate1 Date
examDate2 Date
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
```

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check |
```

```
EXAM.DB | OverallFinding | Comments |
| Check | Check |
```

```
EndQuery
empty("Summary")
if executeQBE(myQuery, "Summary.db") then
tbl.open("Summary")
else
msgStop("Problem","Could not open summary database.")
endif
endmethod
```

Object : #Page4

MethodName : setFocus

Source : method setFocus(var eventInfo Event)

```
var
examMenu Menu
ReportPop PopUpMenu
AddPoP PopUpMenu
endVar

examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
edit()
```

endmethod

Object : #Page4.#Box8.PRINT_REPORT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
myRep Report
reply String
endVar
reply=msgQuestion("View report","Have you updated the report with the SUMMARY query?")
if reply="Yes" then
myRep.print("Summary")
else
return
endif
endmethod

Object : #Page4.#Box8.VIEW_REPORT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
myRep Report
reply String
m menu
endVar
reply=msgQuestion("View report","Have you updated the report with the SUMMARY query?")
if reply="Yes" then
myRep.open("Summary", WinStyleMaximize)
hideSpeedBar()
m.addText("")
m.Show()
message("Shift-F4 for next page, Shift-F3 for previous page, Ctrl-F4 to close report")
else
return
endif
endmethod

Object : #Page4.#Box8.SUMMARY_QUERY_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tbl tableView
myQuery Query
examDate1 Date
examDate2 Date
endVar

```
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
```

```
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check |
```

```
EXAM.DB | OverallFinding | Comments |
| Check | Check |
```

```
EndQuery
empty("Summary")
if executeQBE(myQuery, "Summary.db") then
msgInfo("Summary", "The Summary Query was successful.")
; tbl.open("Summary")
else
msgStop("Problem", "Could not open summary database.")
endif
endmethod
```

Object : EVOLUTIONS_DRILLS_QUERY

MethodName : Const

Source : Const
ViewFlex1=301
ViewFlex2=302
ViewFlex3=303
ViewFlex4=304
ViewFlex5=305
ViewFlex6=306
PrintFlex1=313
PrintFlex2=314
PrintFlex3=315
PrintFlex4=316
PrintFlex5=317
PrintFlex6=318
endConst

Object : EVOLUTIONS_DRILLS_QUERY

MethodName : arrive

Source : method arrive(var eventInfo MoveEvent)
var
tc Tcursor
examMenu, View, Print, ReportMenu Menu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewFlexPop PopUpMenu
PrintFlexPop PopUpMenu
examtype PopupMenu
endVar
if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else
; This code executes only for the form.

PrintPop.addText("Evolutions Section 1","",PrintFlex1)
PrintPop.addText("Drills Section 1","",PrintFlex4)
PrintPop.addText("Evolutions Section 2","",PrintFlex2)
PrintPop.addText("Drills Section 2","",PrintFlex5)
PrintPop.addText("Evolutions Section 3","",PrintFlex3)
PrintPop.addText("Drills Section 3","",PrintFlex6)
PrintPop.addText("&Graph")

ViewPop.addText("Evolutions Section 1","",ViewFlex1)
ViewPop.addText("Drills Section 1","",ViewFlex4)
ViewPop.addText("Evolutions Section 2","",ViewFlex2)
ViewPop.addText("Drills Section 2","",ViewFlex5)
ViewPop.addText("Evolutions Section 3","",ViewFlex3)
ViewPop.addText("Drills Section 3","",ViewFlex6)

```

examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endif
endmethod

```

Object : EVOLUTIONS_DRILLS_QUERY

MethodName : menuAction

```

Source : method menuAction(var eventInfo MenuEvent)
var
myRep Report
reply String
choiceId SmallInt
m menu
endVar
choiceId=eventInfo.id()

if eventInfo.isPreFilter()
then
; This code executes for each object on the form.

else
; This code executes only for the form.

Switch
case eventInfo.menuChoice() ="&Help":
case eventInfo.menuChoice() ="&Quit":
reply=msgQuestion("Quit","Are you sure you want to leave this form?")
If reply = "Yes" then
close()
else
return
endif
case eventInfo.menuchoice() ="&Graph":
myRep.print("taskevol")
endSwitch
Switch
case choiceId =ViewFlex1:
myRep.open("tasksec1")
hideSpeedBar()
m.addText("")
m.show()
case choiceId =ViewFlex2:
myRep.open("tasksec2")

```

```

hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex3:
myRep.open("tasksec3")
hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex4:
myRep.open("drilsec1")
hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex5:
myRep.open("drilsec2")
hideSpeedBar()
m.addText("")
m.show()
case choiceld =ViewFlex6:
myRep.open("drilsec3")
hideSpeedBar()
m.addText("")
m.show()
case choiceld =PrintFlex1:
myRep.print("tasksec1")
case choiceld =PrintFlex2:
myRep.print("tasksec2")
case choiceld =PrintFlex3:
myRep.print("tasksec3")
case choiceld =PrintFlex4:
myRep.print("drilsec1")
case choiceld =PrintFlex5:
myRep.print("drilsec2")
case choiceld =PrintFlex6:
myRep.print("drilsec3")
endSwitch
endif
endmethod

```

Object : #Page2.#Box19.#Button25

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("drilsec3")
endmethod

Object : #Page2.#Box19.#Button31

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl table
tbl1 table
numberOfSats Number
totalNumberOfEvolutions Number
satPercentage Number
myQuery Query
myQuery1 Query
examDate Date
PropType String
endVar
```

```
doDefault
examDate.view("Please enter the desired start date.")
propType.view("Please enter the desired propulsion type.")
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatDrill_2ndSec |
|_EG01 | Check >=50 |
```

```
SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE |
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
|_EG01 | Check |
```

```
OPERATIO.DB | %ofSatEvol_2ndSec | %ofSatEvol_3rdSec | %ofSatDrill_1stSec |
| Check | Check | Check |
```

```
OPERATIO.DB | %ofSatDrill_2ndSec | %ofSatDrill_3rdSec |
| Check | Check |
```

```
SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE |
```

EndQuery

```

executeQBE(myQuery, "DrilSec2.db").
executeQBE(myQuery1, "Tas1Sec1.db")

tbl.attach("DrilSec2")
tbl1.attach("Tas1Sec1")
numberOfSats=tbl.cCount("%ofSatDrill_2ndSec")
msgInfo("Section Two","The total number of sat drill sets are "
+strVal(NumberOfSats))
TotalNumberOfEvolutions=tbl1.cCount("%ofSatDrill_2ndSec")
msgInfo("Section Two","The total number of drill sets are "
+strVal(totalNumberOfEvolutions))
SatPercentage=(numberOfSats/totalNumberOfEvolutions)*100
msgInfo("Section Two","The sat drill set percentage is "
+strVal(satPercentage))
endmethod

```

Object : #Page2.#Box19.#Button30

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tbl table
tbl1 table
numberOfSats Number
totalNumberOfEvolutions Number
satPercentage Number
myQuery Query
myQuery1 Query
examDate Date
PropType String
endVar

```

```

doDefault
examDate.view("Please enter the desired start date.")
propType.view("Please enter the desired propulsion type.")
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatDrill_3rdSec |
|_EG01 | Check >=50 |

```

```

SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
|_EG01 | Check |

OPERATIO.DB | %ofSatEvol_2ndSec | %ofSatEvol_3rdSec | %ofSatDrill_1stSec |
| Check | Check | Check |

OPERATIO.DB | %ofSatDrill_2ndSec | %ofSatDrill_3rdSec |
| Check | Check |

SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate | Check =~propType |

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE |

EndQuery

executeQBE(myQuery, "DrilSec3.db")
executeQBE(myQuery1, "Tas1Sec1.db")

tbl.attach("DrilSec3")
tbl1.attach("Tas1Sec1")
numberOfSats=tbl.cCount("%ofSatDrill_3rdSec")
msgInfo("Section Three", "The total number of sat drill sets are "
+strVal(NumberOfSats))
TotalNumberOfEvolutions=tbl1.cCount("%ofSatDrill_3rdSec")
msgInfo("Section Three", "The total number of drill sets are "
+strVal(totalNumberOfEvolutions))
SatPercentage=(numberOfSats/totalNumberOfEvolutions)*100
msgInfo("Section Three", "The sat drill set percentage is "
+strVal(satPercentage))
endmethod

Object : #Page2.#Box19.#Button23

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("drilsec2")
endmethod

Object : #Page2.#Box19.#Button21

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var

```
newView tableView
endVar
```

```
newView.open("drilsec1")
endmethod
```

Object : #Page2.#Box19.#Button9

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl table
tbl1 table
numberOfSats Number
totalNumberOfEvolutions Number
satPercentage Number
myQuery Query
myQuery1 Query
examDate Date
PropType String
endVar
```

```
doDefault
examDate.view("Please enter the desired start date.")
propType.view("Please enter the desired propulsion type.")
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatDrill_1stSec |
|_EG01 | Check >=50 |
```

```
SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE |
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
|_EG01 | Check |
```

```
OPERATIO.DB | %ofSatEvol_2ndSec | %ofSatEvol_3rdSec | %ofSatDrill_1stSec |
| Check | Check | Check |
```

```
OPERATIO.DB | %ofSatDrill_2ndSec | %ofSatDrill_3rdSec |
| Check | Check |
```

```
SHIP.DB | ShipName      | ExamEndDate  | PropType |
      | Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType  |
      | _EG02                      | Check =OPPE |
```

EndQuery

```
executeQBE(myQuery, "DrilSec1.db")
executeQBE(myQuery1, "Tas1Sec1.db")
```

```
tbl.attach("DrilSec1")
tbl1.attach("Tas1Sec1")
numberOfSats=tbl.cCount("%ofSatDrill_1stSec")
msgInfo("Section One","The total number of sat drill sets are "
+strVal(NumberOfSats))
TotalNumberOfEvolutions=tbl1.cCount("%ofSatDrill_1stSec")
msgInfo("Section One","The total number of drill sets are "
+strVal(totalNumberOfEvolutions))
SatPercentage=(numberOfSats/totalNumberOfEvolutions)*100
msgInfo("Section One","The sat drill set percentage is "
+strVal(satPercentage))
endmethod
```

Object : #Page2.#Box11.#Button15

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("Tas2sec1")
endmethod
```

Object : #Page2.#Box11.#Button17

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("tas2sec3")
endmethod
```

Object : #Page2.#Box11.#Button13

MethodName : pushButton

Object : #Page2.#Box11.#Button35

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl table
tbl1 table
numberOfSats Number
totalNumberOfEvolutions Number
satPercentage Number
myQuery Query
myQuery1 Query
examDate Date
PropType String
endVar
```

```
doDefault
examDate.view("Please enter the desired start date.")
propType.view("Please enter the desired propulsion type.")
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_3rdSec |
|_EG01 | Check >=75 |
```

```
SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE |
```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
|_EG01 | Check |
```

```
OPERATIO.DB | %ofSatEvol_2ndSec | %ofSatEvol_3rdSec |
| Check | Check |
```

```
SHIP.DB | ShipName      | ExamEndDate   | PropType |
      | Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType  |
      | _EG02                      | Check =OPPE |
```

EndQuery

```
executeQBE(myQuery, "Tas2Sec3.db")
executeQBE(myQuery1, "Tas1Sec1.db")
```

```
tbl.attach("Tas2Sec3")
tbl1.attach("Tas1Sec1")
numberOfSats=tbl.cCount("%ofSatEvol_3rdSec")
msgInfo("Section Three", "The total number of sat evolution sets are "
      +strVal(NumberOfSats))
TotalNumberOfEvolutions=tbl1.cCount("%ofSatEvol_2ndSec")
msgInfo("Section Three", "The total number of evolution sets are "
      +strVal(totalNumberOfEvolutions))
SatPercentage=(numberOfSats/totalNumberOfEvolutions)*100
msgInfo("Section Three", "The sat evolution set percentage is "
      +strVal(satPercentage))
endmethod
```

Object : #Page2.#Box11.#Button36

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl table
tbl1 table
numberOfSats Number
totalNumberOfEvolutions Number
satPercentage Number
myQuery Query
myQuery1 Query
examDate Date
PropType String
endVar
```

```
doDefault
examDate.view("Please enter the desired start date.")
propType.view("Please enter the desired propulsion type.")
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_2ndSec |
      | _EG01                      | Check >=75      |
```

```
SHIP.DB | ShipName      | ExamEndDate   | PropType |
```

```

| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
| _EG01 | Check |

```

```

OPERATIO.DB | %ofSatEvol_2ndSec | %ofSatEvol_3rdSec |
| Check | Check |

```

```

SHIP.DB | ShipName | ExamEndDate | PropType |
| Check _EG02, _EG01 | Check >=~examDate| Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE |

```

EndQuery

```

executeQBE(myQuery, "Tas2Sec2.db")
executeQBE(myQuery1, "Tas1Sec1.db")

```

```

tbl.attach("Tas2Sec2")
tbl1.attach("Tas1Sec1")
numberOfSats=tbl.cCount("%ofSatEvol_2ndSec")
msgInfo("Section Two","The total number of sat evolution sets are "
+strVal(NumberOfSats))
TotalNumberOfEvolutions=tbl1.cCount("%ofSatEvol_2ndSec")
msgInfo("Section Two","The total number of evolution sets are "
+strVal(totalNumberOfEvolutions))
SatPercentage=(numberOfSats/totalNumberOfEvolutions)*100
msgInfo("Section Two","The sat evolution percentage is "
+strVal(satPercentage))
endmethod

```

Object : #Page2.#Box11.#Button3

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tbl table
tbl1 table
numberOfSats Number
totalNumberOfEvolutions Number
satPercentage Number

```

```

myQuery Query
myQuery1 Query
examDate Date
PropType String
endVar

```

```

doDefault
examDate.view("Please enter the desired start date.")
propType.view("Please enter the desired propulsion type.")
myQuery=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
              |_EG01                  | Check >=75         |

SHIP.DB | ShipName          | ExamEndDate      | PropType |
         | Check _EG02, _EG01 | Check >=~examDate| Check =~propType|

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType   |
         |_EG02                  | Check =OPPE |

```

EndQuery

myQuery1=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | %ofSatEvol_1stSec |
              |_EG01                  | Check          |

SHIP.DB | ShipName          | ExamEndDate      | PropType |
         | Check _EG02, _EG01 | Check >=~examDate| Check =~propType|

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType   |
         |_EG02                  | Check =OPPE |

```

EndQuery

```

executeQBE(myQuery, "Tas2Sec1.db")
executeQBE(myQuery1, "Tas1Sec1.db")

```

```

tbl.attach("Tas2Sec1")
tbl1.attach("Tas1Sec1")
numberOfSats=tbl.cCount("%ofSatEvol_1stSec")
msgInfo("Section One","The total number of sat evolution sets are "
+strVal(NumberOfSats))
TotalNumberOfEvolutions=tbl1.cCount("%ofSatEvol_1stSec")
msgInfo("Section One","The total number of evolution sets are "
+strVal(totalNumberOfEvolutions))
SatPercentage=(numberOfSats/totalNumberOfEvolutions)*100
msgInfo("Section One","The sat evolution set percentage is "
+strVal(satPercentage))

```

endmethod

Object : #Page27

MethodName : setFocus

Source : method setFocus(var eventInfo Event)
var
examMenu, View, Print, ReportMenu Menu
PrintPop PopUpMenu
ViewPop PopUpMenu
ViewFlexPop PopUpMenu
PrintFlexPop PopUpMenu
examtype PopupMenu
endVar

PrintPop.addText("Evolutions Section 1","",PrintFlex1)
PrintPop.addText("Drills Section 1","",PrintFlex4)
PrintPop.addText("Evolutions Section 2","",PrintFlex2)
PrintPop.addText("Drills Section 2","",PrintFlex5)
PrintPop.addText("Evolutions Section 3","",PrintFlex3)
PrintPop.addText("Drills Section 3","",PrintFlex6)
PrintPop.addText("&Graph")

ViewPop.addText("Evolutions Section 1","",ViewFlex1)
ViewPop.addText("Drills Section 1","",ViewFlex4)
ViewPop.addText("Evolutions Section 2","",ViewFlex2)
ViewPop.addText("Drills Section 2","",ViewFlex5)
ViewPop.addText("Evolutions Section 3","",ViewFlex3)
ViewPop.addText("Drills Section 3","",ViewFlex6)

examMenu.addPopUp("&View", ViewPop)
examMenu.addPopUP("&Print", PrintPop)
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
endmethod

Object : #Page27.RESET_GRAPH_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc Tcursor
endVar
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

Object : #Page27.#Box46.DRILL_SEC3_LISTSHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("drilsec3")
endmethod

Object : #Page27.#Box46.DRILL_SEC2_LISTSHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("drilsec2")
endmethod

Object : #Page27.#Box46.DRILL_SEC1_LISTSHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("drilsec1")
endmethod

Object : #Page27.#Box46.DRILL_SET_PERCENT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsSec1 Number
totalNumberOfSatsSec1 Number
satPercentageSec1 Number
numberOfSatsSec2 Number
totalNumberOfSatsSec2 Number
satPercentageSec2 Number
numberOfSatsSec3 Number
totalNumberOfSatsSec3 Number

```

satPercentageSec3 Number
myQuery1 Query
myQuery2 Query
myQuery3 Query
myQuery4 Query
myQuery5 Query
myQuery6 Query
examDate1 Date
examDate2 Date
PropType String
endVar

```

```

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examDate1.view("Enter the start date (I.E. 01/01/95)")
examDate2.view("Enter the stop date (I.E. 01/01/95)")
propType.view("Enter the prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery1=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 |OperationID_ExamEndDate      | %ofSatDrill_1stSec |
| Check_EG01      |Check >=~examDate1, <=~examDate2| Check >=50      |

```

```

SHIP.DB | ShipName      |PropType      |
|_EG02,_EG01      |Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
|_EG02      | Check =OPPE OR =REOPPE|

```

EndQuery

```

myQuery2=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 |OperationID_ExamEndDate      | %ofSatDrill_1stSec |
| Check_EG01      |Check >=~examDate1, <=~examDate2| Check >0      |

```

```

SHIP.DB | ShipName      |PropType      |
|_EG02,_EG01      |Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG02                | Check =OPPE OR =REOPPE|

```

EndQuery

executeQBE(myQuery1, "DrilSec1.db")

executeQBE(myQuery2, "Tas1Sec1.db")

tbl.attach("DrilSec1")

tbl1.attach("Tas1Sec1")

numberOfSatsSec1=tbl.cCount("%ofSatDrill_1stSec")

msgInfo("Section One","The total number of sat drill sets are "
+strVal(NumberOfSatsSec1))

TotalNumberOfSatsSec1=tbl1.cCount("%ofSatDrill_1stSec")

if totalNumberOfSatsSec1 <> 0 then

msgInfo("Section One","The total number of drill sets are "
+strVal(totalNumberOfSatsSec1))

SatPercentageSec1=(numberOfSatsSec1/totalNumberOfSatsSec1)*100

msgInfo("Section One","The sat drill set percentage is "
+strVal(satPercentageSec1))

else

msgStop("Problem","The total number of watch sections is 0, you cannot divide by 0!")

return

endif

myQuery3=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate      | %ofSatDrill_2ndSec |
      | Check _EG01                | Check >=~examDate1, <=~examDate2| Check >=50        |

```

```

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01    | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG02                | Check =OPPE OR =REOPPE|

```

EndQuery

myQuery4=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate      | %ofSatDrill_2ndSec |
      | Check _EG01                | Check >=~examDate1, <=~examDate2| Check >0          |

```

```

SHIP.DB | ShipName      | PropType      |
      | _EG02, _EG01    | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG02                | Check =OPPE OR =REOPPE|

```

```

EndQuery
executeQBE(myQuery3, "DrilSec2.db")
executeQBE(myQuery4, "Tas1Sec1.db")

tbl.attach("DrilSec2")
tbl1.attach("Tas1Sec1")
numberOfSatsSec2=tbl.cCount("%ofSatDrill_2ndSec")
msgInfo("Section Two","The total number of sat drill sets are "
+strVal(NumberOfSatsSec2))
TotalNumberOfSatsSec2=tbl1.cCount("%ofSatDrill_2ndSec")
if totalNumberOfSatsSec2 <> 0 then
msgInfo("Section Two","The total number of drill sets are "
+strVal(totalNumberOfSatsSec2))
SatPercentageSec2=(numberOfSatsSec2/totalNumberOfSatsSec2)*100
msgInfo("Section Two","The sat drill set percentage is "
+strVal(satPercentageSec2))
else
msgStop("Problem","The total number of watch sections is 0, you cannot divide by 0!")
return
endif

```

myQuery5=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatDrill_3rdSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=50 |

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

myQuery6=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatDrill_3rdsec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

```

executeQBE(myQuery5, "DrilSec3.db")
executeQBE(myQuery6, "Tas1Sec1.db")

```

```

tbl.attach("DrilSec3")
tbl1.attach("Tas1Sec1")
numberOfSatsSec3=tbl.cCount("%ofSatDrill_3rdSec")
msgInfo("Section Three","The total number of sat drill sets are "
+strVal(NumberOfSatsSec3))
TotalNumberOfSatsSec3=tbl1.cCount("%ofSatDrill_3rdSec")
if totalNumberOfSatsSec3 <> 0 then
msgInfo("Section Three","The total number of drill sets are "
+strVal(totalNumberOfSatsSec3))
SatPercentageSec3=(numberOfSatsSec3/totalNumberOfSatsSec3)*100
msgInfo("Section Three","The sat drill set percentage is "
+strVal(satPercentageSec3))
else
msgStop("Problem","The total number of watch sections is 0, you cannot divide by 0!")
return
endif

```

```

else
myQuery1=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 |OperationID_ExamEndDate | %ofSatDrill_1stSec |
| Check_EG01 |Check >=~examDate1, <=~examDate2| Check >=50 |

```

```

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

```

myQuery2=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 |OperationID_ExamEndDate | %ofSatDrill_1stSec |
| Check_EG01 |Check >=~examDate1, <=~examDate2| Check >0 |

```

```

SHIP.DB | ShipName |PropType |
|_EG02,_EG01 |Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

```

executeQBE(myQuery1, "DrilSec1.db")
executeQBE(myQuery2, "Tas1Sec1.db")

```

```

tbl.attach("DrilSec1")
tbl1.attach("Tas1Sec1")

```

```

numberOfSatsSec1=tbl.cCount("%ofSatDrill_1stSec")
msgInfo("Section One","The total number of sat drill sets are "
+strVal(NumberOfSatsSec1))
TotalNumberOfSatsSec1=tbl1.cCount("%ofSatDrill_1stSec")
if totalNumberOfSatsSec1 <> 0 then
    msgInfo("Section One","The total number of drill sets are "
+strVal(totalNumberOfSatsSec1))
    SatPercentageSec1=(numberOfSatsSec1/totalNumberOfSatsSec1)*100
    msgInfo("Section One","The sat drill set percentage is "
+strVal(satPercentageSec1))
else
    msgStop("Problem","The total number of watch sections is 0, you cannot divide by 0!")
return
endif

```

myQuery3=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatDrill_2ndSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=50 |

```

```

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

myQuery4=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatDrill_2ndSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |

```

```

SHIP.DB | ShipName | PropType |
|_EG02,_EG01 | Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
|_EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

```

executeQBE(myQuery3, "DrilSec2.db")
executeQBE(myQuery4, "Tas1Sec1.db")

```

```

tbl.attach("DrilSec2")
tbl1.attach("Tas1Sec1")
numberOfSatsSec2=tbl.cCount("%ofSatDrill_2ndSec")
msgInfo("Section Two","The total number of sat drill sets are "
+strVal(NumberOfSatsSec2))
TotalNumberOfSatsSec2=tbl1.cCount("%ofSatDrill_2ndSec")

```

```

if totalNumberOfSatsSec2 <> 0 then
    msgInfo("Section Two","The total number of drill sets are "
        +strVal(totalNumberOfSatsSec2))
    SatPercentageSec2=(numberOfSatsSec2/totalNumberOfSatsSec2)*100
    msgInfo("Section Two","The sat drill set percentage is "
        +strVal(satPercentageSec2))
else
    msgStop("Problem","The total number of watch sections is 0, you cannot divide by 0!")
return
endif

```

myQuery5=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatDrill_3rdSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=50 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

myQuery6=Query

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatDrill_3rdsec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

```

executeQBE(myQuery5, "DrilSec3.db")
executeQBE(myQuery6, "Tas1Sec1.db")

```

```

tbl.attach("DrilSec3")
tbl1.attach("Tas1Sec1")
numberOfSatsSec3=tbl.cCount("%ofSatDrill_3rdSec")
msgInfo("Section Three","The total number of sat drill sets are "
    +strVal(NumberOfSatsSec3))
TotalNumberOfSatsSec3=tbl1.cCount("%ofSatDrill_3rdSec")
if totalNumberOfSatsSec3 <> 0 then
    msgInfo("Section Three","The total number of drill sets are "
        +strVal(totalNumberOfSatsSec3))
    SatPercentageSec3=(numberOfSatsSec3/totalNumberOfSatsSec3)*100

```

```

msgInfo("Section Three","The sat drill set percentage is "
+strVal(satPercentageSec3))
else
msgStop("Problem","The total number of watch sections is 0, you cannot divide by 0!")
return
endif
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageSec1
tc("Percentage1")=SatPercentageSec2
tc("Percentage2")=SatPercentageSec3
tc("PropType")=propType
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```

Object : #Page27.#Box28.EVOLUTION_PERCENT_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
tc tCursor
tbl table
tbl1 table
numberOfSatsSec1 Number
totalNumberOfEvolutionsSec1 Number
satPercentageSec1 Number
numberOfSatsSec2 Number
totalNumberOfEvolutionsSec2 Number
satPercentageSec2 Number
numberOfSatsSec3 Number
totalNumberOfEvolutionsSec3 Number
satPercentageSec3 Number
myQuery1 Query
myQuery2 Query
myQuery3 Query
myQuery4 Query
myQuery5 Query
myQuery6 Query
examDate1 Date
examDate2 Date
PropType String
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
propType="GT"
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")

```



```

propType.view("Enter prop type (GT/STM/DSL)")
switch
case propType="GT":
case propType="STM":
case propType="DSL":
case propType="ALL":
otherwise:
msgStop("Problem","The choices for Propulsion Type are GT, STM, DSL, or ALL only!")
return
endSwitch

```

```

if propType="ALL" then
myQuery1=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_1stSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=65 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

EndQuery

```

myQuery2=Query

```

ANSWER: :PRIV:ANSWER.DB

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_1stSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR REOPPE|

```

EndQuery

```

executeQBE(myQuery1, "Tas2Sec1.db")
executeQBE(myQuery2, "Tas1Sec1.db")

```

```

tbl.attach("Tas2Sec1")
tbl1.attach("Tas1Sec1")
numberOfSatsSec1=tbl.cCount("%ofSatEvol_1stSec")
msgInfo("Section One","The total number of sat evolution sets are "
+strVal(NumberOfSatsSec1))
TotalNumberOfEvolutionsSec1=tbl1.cCount("%ofSatEvol_1stSec")
if totalNumberOfEvolutionsSec1 <> 0 then
msgInfo("Section One","The total number of evolution sets are "

```

```

+strVal(totalNumberOfEvolutionsSec1))
SatPercentageSec1=(numberOfSatsSec1/totalNumberOfEvolutionsSec1)*100
msgInfo("Section One","The sat evolution set percentage is "
+strVal(satPercentageSec1))
else
msgStop("Problem","The total number of evolutions is 0, you cannot divide by 0!")
return
endif

```

```
myQuery3=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_2ndSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=65 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

```
EndQuery
```

```
myQuery4=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_2ndSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

```
EndQuery
```

```
executeQBE(myQuery3, "Tas2Sec2.db")
```

```
executeQBE(myQuery4, "Tas1Sec1.db")
```

```
tbl.attach("Tas2Sec2")
```

```
tbl1.attach("Tas1Sec1")
```

```
numberOfSatsSec2=tbl.cCount("%ofSatEvol_2ndSec")
```

```
msgInfo("Section Two","The total number of sat evolution sets are "
```

```
+strVal(NumberOfSatsSec2))
```

```
TotalNumberOfEvolutionsSec2=tbl1.cCount("%ofSatEvol_2ndSec")
```

```
if TotalNumberOfEvolutionsSec2 <> 0 then
```

```
msgInfo("Section Two","The total number of evolution sets are "
```

```
+strVal(totalNumberOfEvolutionsSec2))
```

```
SatPercentageSec2=(numberOfSatsSec2/totalNumberOfEvolutionsSec2)*100
```

```
msgInfo("Section Two","The sat evolution percentage is "
```

```
+strVal(satPercentageSec2))
```

```
else
```

```

msgStop("Problem","The total number of evolutions is 0, you cannot divide by 0!")
return
endif

```

```

myQuery5=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_3rdSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=65 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

```

EndQuery

```

```

myQuery6=Query

```

```

ANSWER: :PRIV:ANSWER.DB

```

```

OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_3rdSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |

```

```

SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check |

```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|

```

```

EndQuery

```

```

executeQBE(myQuery5, "Tas2Sec3.db")
executeQBE(myQuery6, "Tas1Sec1.db")

```

```

tbl.attach("Tas2Sec3")

```

```

tbl1.attach("Tas1Sec1")

```

```

numberOfSatsSec3=tbl.cCount("%ofSatEvol_3rdSec")

```

```

msgInfo("Section Three","The total number of sat evolution sets are "

```

```

+strVal(numberOfSatsSec3))

```

```

TotalNumberOfEvolutionsSec3=tbl1.cCount("%ofSatEvol_3rdSec")

```

```

if TotalNumberOfEvolutionsSec3 <> 0 then

```

```

msgInfo("Section Three","The total number of evolution sets are "

```

```

+strVal(totalNumberOfEvolutionsSec3))

```

```

SatPercentageSec3=(numberOfSatsSec3/totalNumberOfEvolutionsSec3)*100

```

```

msgInfo("Section Three","The sat evolution set percentage is "

```

```

+strVal(satPercentageSec3))

```

```

else

```

```

msgStop("Problem","The total number of evolutions is 0, you cannot divide by 0!")

```

```

return

```

```

endif

```

```
else
myQuery1=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_1stSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=65 |
```

```
SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE OR =REOPPE|
```

```
EndQuery
```

```
myQuery2=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_1stSec |
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |
```

```
SHIP.DB | ShipName | PropType |
| _EG02, _EG01 | Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |
| _EG02 | Check =OPPE |
```

```
EndQuery
```

```
executeQBE(myQuery1, "Tas2Sec1.db")
executeQBE(myQuery2, "Tas1Sec1.db")
```

```
tbl.attach("Tas2Sec1")
tbl1.attach("Tas1Sec1")
numberOfSatsSec1=tbl.cCount("%ofSatEvol_1stSec")
msgInfo("Section One","The total number of sat evolution sets are "
+strVal(NumberOfSatsSec1))
TotalNumberOfEvolutionsSec1=tbl1.cCount("%ofSatEvol_1stSec")
if totalNumberOfEvolutionsSec1 <> 0 then
msgInfo("Section One","The total number of evolution sets are "
+strVal(totalNumberOfEvolutionsSec1))
SatPercentageSec1=(numberOfSatsSec1/totalNumberOfEvolutionsSec1)*100
msgInfo("Section One","The sat evolution set percentage is "
+strVal(satPercentageSec1))
else
msgStop("Problem","The total number of evolutions is 0, you cannot divide by 0!")
return
endif
```

```
myQuery3=Query
```

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_2ndSec |  
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=65 |
```

```
SHIP.DB | ShipName | PropType |  
| _EG02, _EG01 | Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |  
| _EG02 | Check =OPPE OR =REOPPE|
```

EndQuery

myQuery4=Query

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_2ndSec |  
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >0 |
```

```
SHIP.DB | ShipName | PropType |  
| _EG02, _EG01 | Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType |  
| _EG02 | Check =OPPE OR =REOPPE|
```

EndQuery

executeQBE(myQuery3, "Tas2Sec2.db")

executeQBE(myQuery4, "Tas1Sec1.db")

tbl.attach("Tas2Sec2")

tbl1.attach("Tas1Sec1")

numberOfSatsSec2=tbl.cCount("%ofSatEvol_2ndSec")

msgInfo("Section Two","The total number of sat evolution sets are "

+strVal(NumberOfSatsSec2))

TotalNumberOfEvolutionsSec2=tbl1.cCount("%ofSatEvol_2ndSec")

if TotalNumberOfEvolutionsSec2 <> 0 then

msgInfo("Section Two","The total number of evolution sets are "

+strVal(totalNumberOfEvolutionsSec2))

SatPercentageSec2=(numberOfSatsSec2/totalNumberOfEvolutionsSec2)*100

msgInfo("Section Two","The sat evolution percentage is "

+strVal(satPercentageSec2))

else

msgStop("Problem","The total number of evolutions is 0, you cannot divide by 0!")

return

endif

myQuery5=Query

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 | OperationID_ExamEndDate | %ofSatEvol_3rdSec |  
| Check_EG01 | Check >=~examDate1, <=~examDate2| Check >=65 |
```

```
SHIP.DB | ShipName      |PropType      |
      | _EG02, _EG01    |Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG02                      |Check =OPPE OR =REOPPE|
```

EndQuery

myQuery6=Query

ANSWER: :PRIV:ANSWER.DB

```
OPERATIO.DB | Operati_Ship_ShipName_FK3 |OperationID_ExamEndDate      | %ofSatEvol_3rdSec |
      | Check _EG01                    |Check >=~examDate1, <=~examDate2| Check >0          |
```

```
SHIP.DB | ShipName      |PropType      |
      | _EG02, _EG01    |Check =~propType|
```

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamType      |
      | _EG02                      |Check =OPPE OR =REOPPE|
```

EndQuery

executeQBE(myQuery5, "Tas2Sec3.db")

executeQBE(myQuery6, "Tas1Sec1.db")

tbl.attach("Tas2Sec3")

tbl1.attach("Tas1Sec1")

numberOfSatsSec3=tbl.cCount("%ofSatEvol_3rdSec")

msgInfo("Section Three","The total number of sat evolution sets are "

+strVal(NumberOfSatsSec3))

TotalNumberOfEvolutionsSec3=tbl1.cCount("%ofSatEvol_3rdSec")

if TotalNumberOfEvolutionsSec3 <> 0 then

msgInfo("Section Three","The total number of evolution sets are "

+strVal(totalNumberOfEvolutionsSec3))

SatPercentageSec3=(numberOfSatsSec3/totalNumberOfEvolutionsSec3)*100

msgInfo("Section Three","The sat evolution set percentage is "

+strVal(satPercentageSec3))

else

msgStop("Problem","The total number of evolutions is 0, you cannot divide by 0!")

return

endif

endif

tc.open("percent")

TC.edit()

tc.insertRecord()

tc.("Percentage")=SatPercentageSec1

tc.("Percentage1")=SatPercentageSec2

tc.("Percentage2")=SatPercentageSec3

tc.("PropType")=propType

tc.("examDate1")=examDate1

tc.("examDate2")=examDate2

tc.endEdit()

endmethod

Object : #Page27.#Box28.EVOL_SEC3_LISTSHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("tas2sec3")
endmethod

Object : #Page27.#Box28.EVOL_SEC1_LISTSHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("Tas2sec1")
endmethod

Object : #Page27.#Box28.EVOL_SEC2_LISTSHIP_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)
var
newView tableView
endVar

newView.open("tas2sec2")
endmethod

Object : SPECIAL_SITUATIONS_QUERY

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    examMenu.addText("&Quit")
    examMenu.show()
    maximize()
    hideSpeedBar()
    edit()
    endif
endmethod
```

Object : SPECIAL_SITUATIONS_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

    Switch
      case eventInfo.menuChoice() = "&Help":

      case eventInfo.menuChoice() = "&Quit":
        reply=msgQuestion("Quit","Are you sure you want to leave this form?")
        If reply = "Yes" then
          close()
        else
          return
        endif
      endSwitch
    endif
endmethod
```


Object : #Page2

MethodName : setFocus

```
Source : method setFocus(var eventInfo Event)
var
  examMenu Menu
  ReportPop PopUpMenu
  AddPoP PopUpMenu
endVar

  examMenu.addText("&Quit")
  examMenu.show()
  maximize()
  hideSpeedBar()
  edit()
endmethod
```

Object : #Page2.#Box3.PRINT_REPORT_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
  myRep Report
  reply String
endVar
  reply=msgQuestion("Print the report","Have you updated the report with the queries?")
  if reply="Yes" then
    myRep.print("Special")
  else
    return
  endif
endmethod
```

Object : #Page2.#Box3.VIEW_REPORT_BUTTON

MethodName : pushButton

```
Source : method pushButton(var eventInfo Event)
var
  myRep Report
  reply String
  m menu
endVar
  reply=msgQuestion("View report","Have you updated the report with the queries?")
  if reply="Yes" then
    myRep.open("Special", WinStyleMaximize)
    hideSpeedBar()
    m.addText("")
    m.show()
    message("Shift-F4 for next page, Shift-F3 for previous page, Ctrl-F4 to close report")
  else
```

```
return
endif
endmethod
```

Object : #Page2.#Box3.OVERDUE_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl tableView
myQuery Query
examDate1 Date
endVar
doDefault
examDate1=today()
myQuery=Query
```

ANSWER: :PRIV:ANSWER.DB

```
EXAM.DB | ExamID_Ship_ShipName_FK2 | NextExamDate      |
| Check_EG01      | Check <=~examDate1,NOT BLANK|
```

EndQuery

```
empty("LateExam")
if executeQBE(myQuery, "LateExam.db") then
msgInfo("Overdue Query","The Overdue Query was successful!")
else
msgStop("Problem","Sorry, could not open LateExam database.")
endif
endmethod
```

Object : #Page2.#Box3.INCOMPLETE_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```
var
tbl tableView
myQuery Query
examDate1 Date
examDate2 Date
endVar
```

```
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")
```

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType|
| Check | Check >=~examDate1, <=~examDate2| Check |

```

```

EXAM.DB | OverallFinding| Comments | NextExamDate | Cleared |
| Check =INC | Check | Check | Check =N|

```

EndQuery

```

empty("INC")
if executeQBE(myQuery, "INC.db") then
    msgInfo("Incomplete Query","The Incomplete Query was successful!")
else
    msgStop("Problem","Sorry, could not open INCOMPLETE database.")
endif
endmethod

```

Object : #Page2.#Box3.UNSAT_BUTTON

MethodName : pushButton

Source : method pushButton(var eventInfo Event)

```

var
tbl tableView
myQuery Query
examDate1 Date
examDate2 Date
endVar

doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")

```

myQuery=Query

ANSWER: :PRIV:ANSWER.DB

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate | ExamType|
| Check | Check >=~examDate1, <=~examDate2| Check |

```

```

EXAM.DB | OverallFinding| Comments | NextExamDate | Cleared |
| Check =UNS | Check | Check | Check =N|

```

EndQuery

```

empty("Unsat")
if executeQBE(myQuery, "Unsat.db") then
    msgInfo("Unsatisfactory query","The Unsatisfactory Query was successful!")
else
    msgStop("Problem","Sorry, could not open the unsat database.")

```

```
endif  
endmethod
```

Object : CUMULATIVE_TREND_QUERY

MethodName : arrive

```
Source : method arrive(var eventInfo MoveEvent)
var
  tc Tcursor
  examMenu, View, Print, ReportMenu Menu
  PrintPop PopUpMenu
  ViewPop PopUpMenu
  ViewFlexPop PopUpMenu
  PrintFlexPop PopUpMenu
  examtype PopupMenu
endVar
if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

  examMenu.addText("&Print")
  examMenu.addText("&Quit")
  examMenu.show()
  maximize()
  hideSpeedBar()
  tc.open("percent")
  tc.edit()
  tc.empty()
  tc.endEdit()
endif
endmethod
```

Object : CUMULATIVE_TREND_QUERY

MethodName : menuAction

```
Source : method menuAction(var eventInfo MenuEvent)
var
  myRep Report
  reply String
  choiceId SmallInt
endVar
choiceId=eventInfo.id()

if eventInfo.isPreFilter()
  then
    ; This code executes for each object on the form.

  else
    ; This code executes only for the form.

  Switch
```

```

        case eventInfo.menuChoice() = "&Help":
    case eventInfo.menuChoice() = "&Quit":
        reply=msgQuestion("Quit","Are you sure you want to leave this form?")
        If reply = "Yes" then
            close()
        else
            return
        endif
    case eventInfo.menuChoice() = "&Print":
        myRep.print("CumTrend")
    endSwitch
endif
endmethod

```

Object : #Page2

MethodName : setFocus

```

Source : method setFocus(var eventInfo Event)
var
    tc Tcursor
    examMenu, View, Print, ReportMenu Menu
    PrintPop PopUpMenu
    ViewPop PopUpMenu
    ViewFlexPop PopUpMenu
    PrintFlexPop PopUpMenu
    examtype PopupMenu
endVar
examMenu.addText("&Print")
examMenu.addText("&Quit")
examMenu.show()
maximize()
hideSpeedBar()
tc.open("percent")
tc.edit()
tc.empty()
tc.endEdit()
endmethod

```

Object : #Page2.OPPE_PERCENT_BUTTON

MethodName : pushButton

```

Source : method pushButton(var eventInfo Event)
var
    tc tCursor
    tbl table
    tbl1 table
    numberOfSatsOppe Number
    totalNumberOfOppe Number
    satPercentageOppe Number
    myQuery Query
    myQuery1 Query
    myQuery2 Query

```

```

examDate1 Date
examDate2 Date
propType String
examType String
endVar
doDefault
examDate1=date("01/01/00")
examDate2=date("12/31/99")
examDate1.view("Enter start date (I.E. 01/01/95)")
examDate2.view("Enter stop date (I.E. 01/01/95)")

```

```
myQuery=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =SAT OR
=EXC OR =GOOD|

```

```

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

```

```
EndQuery
```

```
myQuery1=Query
```

```
ANSWER: :PRIV:ANSWER.DB
```

```

EXAM.DB | ExamID_Ship_ShipName_FK2 | ExamID_ExamEndDate |ExamType |
OverallFinding |
| Check_EG01 | Check >=~examDate1, <=~examDate2|Check =OPPE | Check =SAT OR
=UNS OR =GOOD OR =EXC|

```

```

SHIP.DB | ShipName | PropType |
|_EG01 | Check |

```

```
EndQuery
```

```

empty("OppeSum")
empty("OppeSum1")
executeQBE(myQuery, "OppeSum.db")
executeQBE(myQuery, "OppeSum1.db")
tbl.attach("OppeSum")
tbl1.attach("OppeSum1")
numberOfSatsOppe=tbl.cCount("OverallFinding")
msgInfo("OPPE","The total number of sats are "
+strVal(NumberOfSatsOppe))
TotalNumberOfOppe=tbl1.cCount("OverallFinding")
if TotalNumberOfOppe <> 0 then
msgInfo("OPPE","The total number is "
+strVal(totalNumberOfOppe))
SatPercentageOppe=(numberOfSatsOppe/totalNumberOfOppe)*100
msgInfo("OPPE","The sat percentage is "

```

```

        +strVal(satPercentageOppe))
else
    msgStop("Problem","The total number of OPPE's is 0, you cannot divide by 0!")
    return
endif
tc.open("percent")
TC.edit()
tc.insertRecord()
tc("Percentage")=SatPercentageOppe
tc("PropType")="OPPE"
tc("examDate1")=examDate1
tc("examDate2")=examDate2
tc.endEdit()
endmethod

```


APPENDIX H. USER'S MANUAL

A. INTRODUCTION

The purpose of this manual is to familiarize the user with the Propulsion Examining Board's Database System (PEBDS). The system uses a series of windows and pull down menus. The system is designed to be very easy to use but does require familiarity with Microsoft Windows and the use of a mouse.

B. GETTING STARTED

Before installing the PEBDS, the Paradox for Windows version 4.5 or 5.0 database management software and Microsoft Windows 3.1 or Windows 95 must be installed as specified in their respective user's manuals. The application files in the provided 3.5 inch floppy disks must be copied to the hard disk into a directory labeled PEBDS. If the PEBDS directory does not exist, you must create the directory either through DOS or the Windows Program Manager. Once the files are copied into PEBDS directory, there are two ways to initialize the PEBDS.

The first way to load the PEBDS is to initialize Paradox for Windows by double clicking on the Paradox for Windows icon. Once Paradox for Windows is loaded, you must change the working directory of the database management software to C:\PEBDS by selecting FILE from the main menu and then selecting WORKING DIRECTORY. Type in C:\PEBDS and select save, this will change the working directory of Paradox for Windows. If you are having difficulty changing the working directory, consult the user's manual for Paradox for Windows for further guidance. Once the Paradox for Windows desktop is visible on the screen, initialize the startup script (Startup.ssl) by selecting Script from the Open drop down menu. The PEBDS will be loaded with the main menu as the first screen.

The second way to load the PEBDS, which is more robust, is to load the PEBDS directly from the Windows Program Manager. You do this by providing an icon that both loads Paradox for Windows and the startup script. You first must make the PEB Database program group by selecting New from the File drop down menu. Select the Program Group radio button, type in PEB Database in the description box and hit Enter. Next select the Paradox for Windows icon and then select Copy from the File drop down menu. The Copy Program Item dialog box will give you a drop down list to select where you want to copy the icon to. Select the PEB Database program group and hit Enter. Next highlight the Paradox for Windows icon and choose Properties from the File drop down menu. Here you will change the name of the icon to PEBDS by typing PEBDS in the Description box. You can now change the look of the icon if you desire by selecting the Change Icon pushbutton. Next you must change the Command Line to load the startup script to look like:

```
C:\OFFICE\PDOXWIN\PDOXWIN.EXE  STARTUP.SSL -c -q -w C:\PEBDS
```

Next select the OK pushbutton and you will now have a separate program group and icon to load the PEBDS. From the program manager select the PEBDS icon and initialize the application (see figure 2). Once the application is loaded, the next screen to appear is the main menu for the PEBDS. The main menu, (see figure 2), allows the user to input data, process queries, backup data and quit to the Program Manager through ADD, INQUIRY, BACKUP and QUIT respectively.

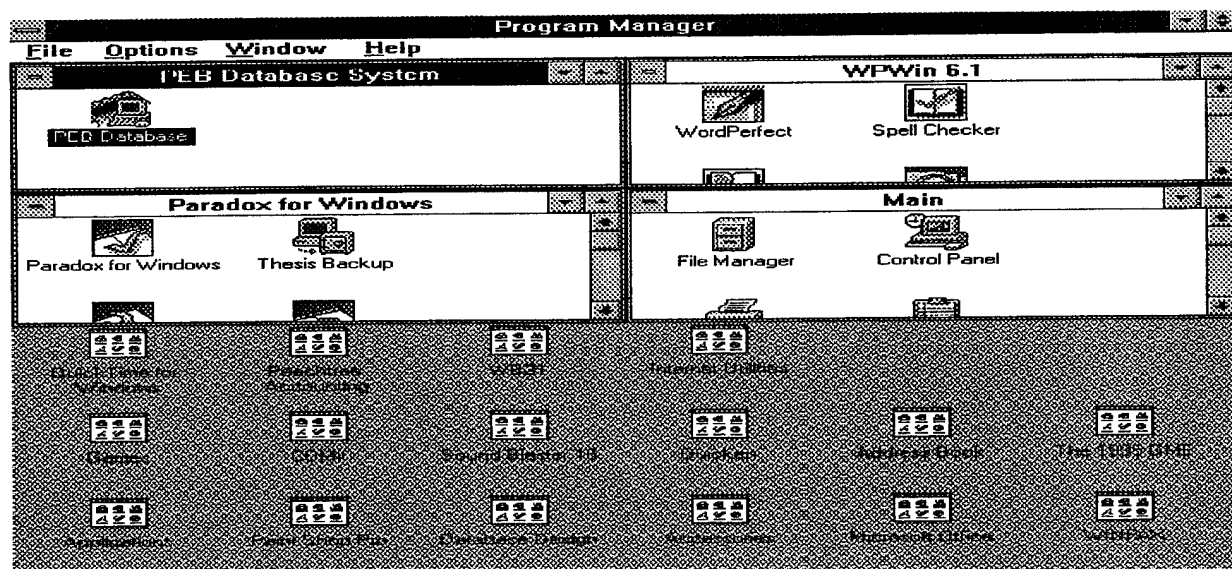


Figure 1. Program Manager screen.

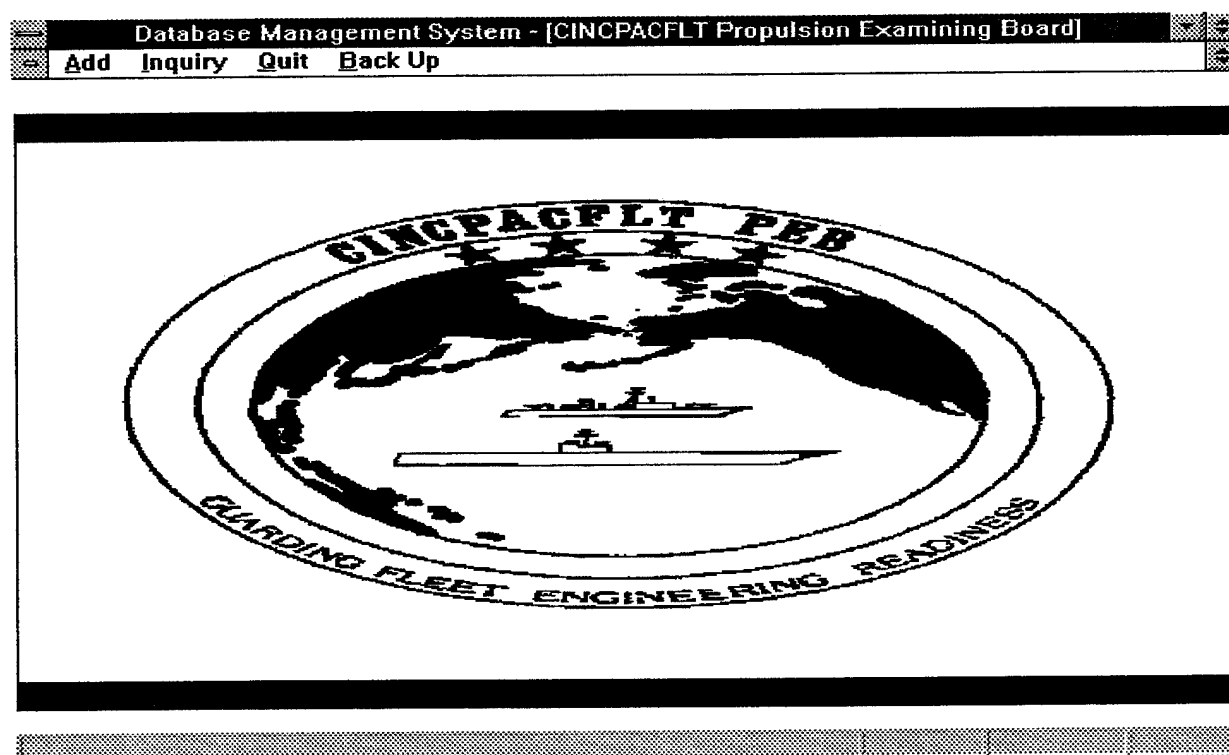


Figure 2. Main menu screen

C. MAIN MENU

1. ADD

The ADD menu selection will give the user the option to add the following: Ship, Exam, Fire

Fighting, Level of Knowledge, Material, Operations, Program Management, and Training. When an option is selected, the respective data entry screen will appear and data entry can begin immediately (see figure 3). When the last field of data is entered, the cursor will automatically scroll to the first field. In addition, there are several default values in selected fields. They have been added to decrease data entry. These default values however, can be changed if required. If the user requires to enter another record, the user will press the push button. When the push button is pressed, the process can now be repeated. If a correction is needed to the current record, the user may choose to do either of the following: one press TAB or SHIFT-TAB key(s) to move forward or backward

Figure 3. Exam input screen.

respectively or two, position and click the mouse on the field to change.

All input screens have identical main menus. The choices on the main menu are RECORD and QUIT (see figure 3). The RECORD option produces a drop down menu with the choice to LOCATE or DELETE. When the LOCATE option is used, a locating dialog box appears and the user may locate any record by any field. The default field is the ship name on all input screens. However, this default value may be changed by scrolling through the available fields for that screen. The DELETE option will delete the current record. A safeguard has been added by providing a question dialog box asking the user to verify that he or she wants to delete the current record. **Once a record is deleted, it cannot be restored unless it is reentered.** The QUIT option closes the input screen and returns you to the main menu.

The PEBDS has been designed to provide a robust and user friendly environment. However, if the user tries to enter a record that has been previously entered, a **Key Violation** error will result. If this occurs, delete the current record or change the information that is in error (this will most likely be the exam end date and ship name combination). This error results from a ship having two exams ending on the same date. All other errors are covered by the systems internal error checking mechanism.

NOTE: You must ADD the ship before any other data entry is conducted. The PEBDS is dependent on the ship and data pertaining to the ship in each input screen will not be allowed unless the ship is in the database first.

2. INQUIRY

The Inquiry selection will give the user the option to manipulate the following queries: Boiler Flexes, ECCTT, Fire Drills, High Power Demo, OPPE/LOE Area Summary, Program Statistics, Monthly OPPE/LOE Summary, and Evolutions and Drills. When an option is selected, the respective query screen will appear and manipulation of the query can begin immediately. Since the operation of the query screens vary, each of the query screens will be illustrated. However there are several features that are common to all query screens and they will be discussed first.

The similar features common to all query screens are the start date, stop date, propulsion type, and exam type input windows (See figures 4, 5, 6 and 7). Other features are the LIST SHIPS and RESET GRAPH push buttons. The start date and stop date inputs define the period you want to search. The format for the input is 01/01/95, 01-01-95, or 01.01.95. If another format is chosen, the installed error checking mechanism will signal the user that the choice was invalid. The exam type input defines which type of exam you want to search. The choices for are OPPE, REOPPE, LOE, or RELOE. The choices must be spelled exactly and must up in **uppercase** for the input to be valid. The propulsion type input defines which ship type you want to search. The choices are GT (Gas Turbine), STM (Steam), DSL (Diesel) or ALL (all ship types). Again the choices must be spelled exactly and must be in **uppercase** for the input to be valid. The exam type and propulsion type inputs are common on all but a few of the query screens. The LIST SHIPS push button displays the ships in a table format. The RESET GRAPH push button clears the graph for the user to begin another query. Caution should be exercised when pressing this push button to prevent from inadvertently clearing the graphical display.

A. Boiler Flexes Query

The Boiler Flexes query (see figure 8) enables the user to graphically display the percentage of boiler flex levels. To start the query, the user simply presses which level he or she wants to display, for example LEVEL 1. The start date and stop date input dialog boxes appear, the desired period is then entered and the query conducts the search. The next dialog boxes are informative in nature and the result is graphically displayed. To compare the different levels against each other, the user must be sure that he or she is consistent with the start and stop dates when the other level push buttons are pressed. The LIST SHIPS and RESET GRAPH push buttons were discussed earlier. The Boiler

Enter start date (I.E. 01/01/95)

01/01/95

✓ OK ✗ Cancel

Enter stop date (I.E. 12/12/99)

12/31/95

✓ OK ✗ Cancel

Enter prop type (GT/STM/DSL)

GT

✓ OK ✗ Cancel

Enter (OPPE/RE OPPE/LOE/RELOE)

OPPE

✓ OK ✗ Cancel

Figure 4.

Figure 5.

Figure 6.

Figure 7.

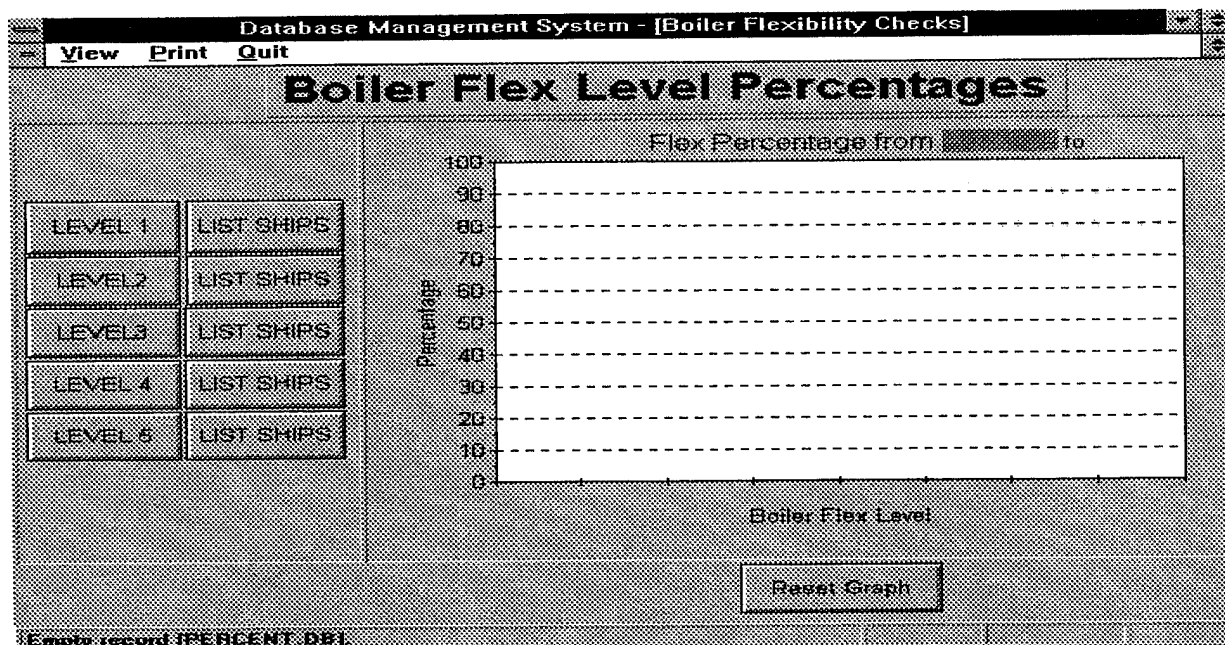


Figure 8. Boiler Flex query.

Flexes main menu has three options. The options are View, Print, and Quit. The View option displays a report that the user can use for information purposes. The Print option displays two choices. The choices are Report and Graph. The Report choice prints the report that was previously viewed and the Graph choice prints the graph that is displayed on the screen. The Quit option returns the user to the main menu.

B. ECCTT query

The ECCTT query (see figure 9) enables the user to graphically display the ECCTT satisfactory percentage. To start the query, the user simply presses the ECCTT push button. The start date, stop date, and propulsion type input dialog boxes appear. The desired period and propulsion type are entered and the query conducts the search. The next dialog boxes are informative in nature and the result is graphically displayed. To compare the ECCTTs of different propulsion types, the user must be sure that he or she is consistent with the start and stop dates. The LIST SHIPS and RESET GRAPH push buttons have been discussed earlier. The ECCTT query main menu has two options. The options are Report and Quit. The Report option displays two choices. The choices are View and Print. The View choice has two sub-choices. The sub-choices are Unsat ECCTT Report and Sat ECCTT Report. The user can view these reports for information purposes. The Print choice has two sub-choices. The sub-choices are Unsat ECCTT Report, Sat ECCTT Report, and Graph. The Print choice prints the reports that were previously reviewed and the graph that is displayed. The Quit option returns the user to the main menu.

C. Fire Drills query

The Fire Drills query (see figure 10) enables the user to graphically display the Fire Drills percentage. To start the query, the user simply presses one of three Fire Drill push buttons. The start date, stop date, propulsion type, and exam type input dialog boxes appear. The desired inputs are entered and the query conducts the search. The next dialog boxes are informative in nature

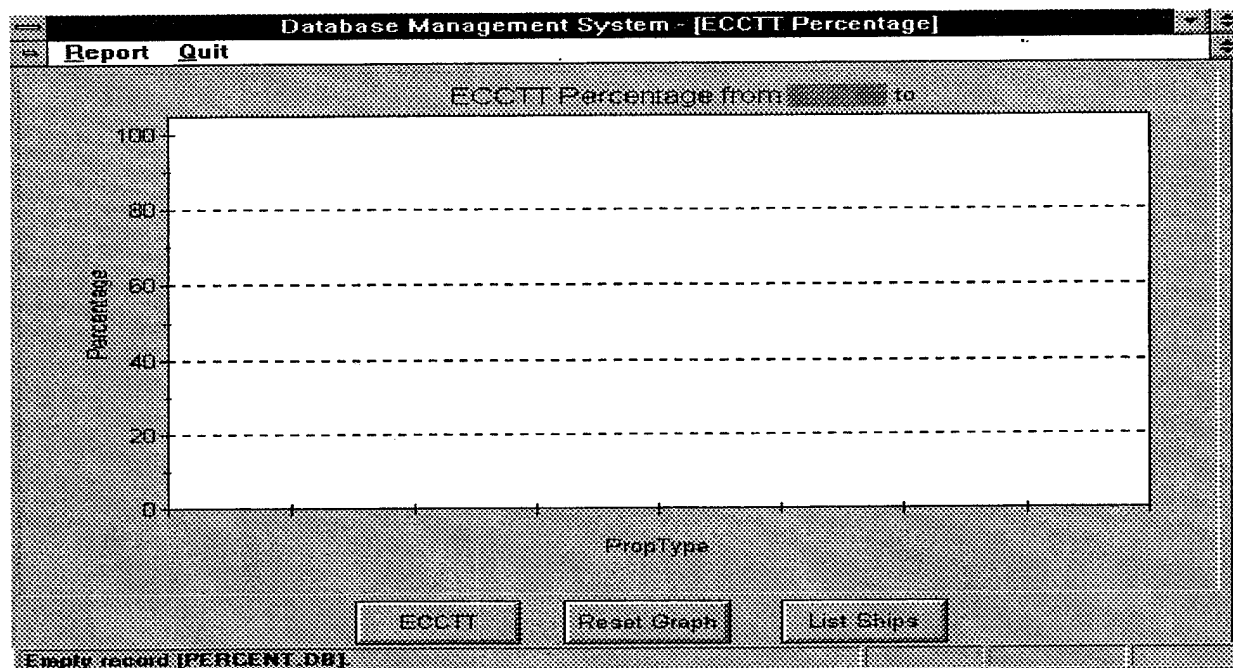


Figure 9. ECCTT query.

and the result is graphically displayed. To compare the Fire Drills of different propulsion types, the user must be sure that he or she is consistent with the start and stop dates. The LIST SHIPS and RESET GRAPH push buttons have been discussed earlier. The Fire Drills query main menu has three

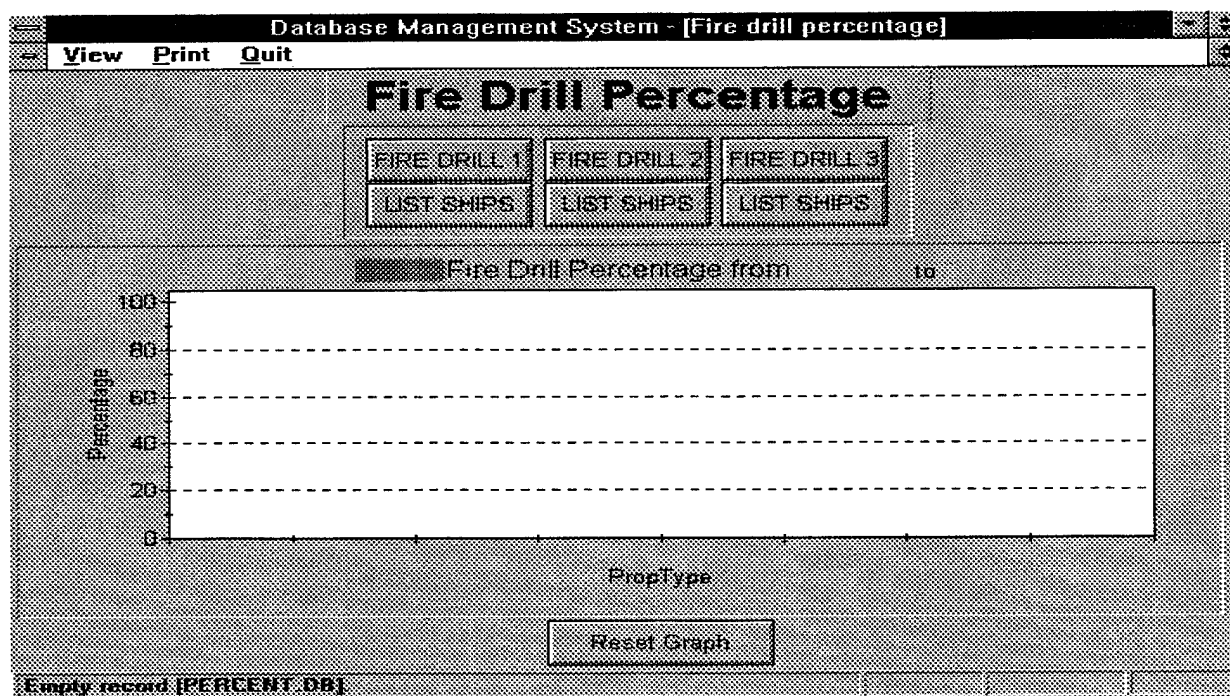


Figure 10. Fire Drills query.

options. The options are View, Print and Quit. The View option displays three choices. The choices are Fire Drill 1, FireDrill 2, and Fire Drill 3. The user can view these reports for information purposes. The Print option displays four choices. The choices are Fire Drill 1, Fire Drill 2, Fire Drill 3, and Graph. The Print option prints the reports that were previously reviewed and the graph that is displayed. The Quit option returns the user to the main menu.

D. High Power Demo query

The High Power Demo query (see figure 11) enables the user to graphically display the High Power Demo satisfactory percentage. To start the query, the user simply presses the High Power Demo push button. The start date, stop date, and propulsion type input dialog boxes appear. The desired period and propulsion types are entered and the query conducts the search. The next dialog boxes are informative in nature and the result is graphically displayed. To compare the High Power Demos of different propulsion types, the user must be sure that he or she is consistent with the start and stop dates. The LIST SHIPS and RESET GRAPH push buttons have been discussed earlier. The High Power Demo query main menu has two options. The two options are Report and Quit. The Report option displays two choices. The choices are View and Print. The View choice enables the user to view the high power demo report. The Print choice has two sub-choices. The sub-choices are High Power report and Graph. The Print choice prints the High Power report previously viewed and the Graph that is displayed. The Quit option returns the user to the main menu.

E. OPPE/LOE Area Summary query

The OPPE/LOE Area Summary query (see figure 12) enables the user to graphically display the OPPE/LOE area satisfactory percentage. To start the query, the user simply presses the desired OPPE/LOE area push button. The start date, stop date, propulsion type, and exam type input dialog boxes appear. The desired inputs are entered and the query conducts the search. The next dialog boxes are informative in nature and the result is graphically displayed.

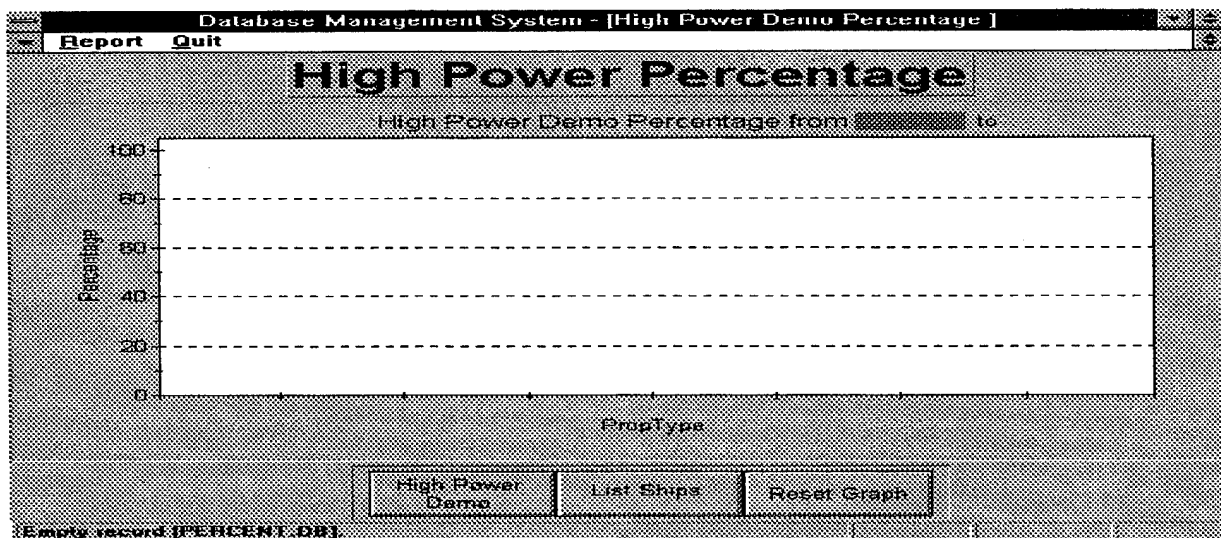


Figure 11. High Power Percentage query.

TIP: It is a good idea to press the OPPE or LOE percentage push button first, then the area push buttons to keep the graph consistent. Also, you should keep the propulsion type constant.

To compare the OPPE/LOE Areas, the user must be sure that he or she is consistent with the start and stop dates. The LIST SHIPS and RESET GRAPH push buttons have been discussed earlier. The OPPE/LOE Area Summary query main menu has three options. The three options are View, Print, and Quit. The View choice has two sub-choices. The sub-choices are Sat and Unsat. These sub-choices enable the user to view the following reports: OPPE Percentage, Operations Percentage, Fire Fighting Percentage, Material Percentage, Training Percentage, Program Management Percentage, and LOE percentage.

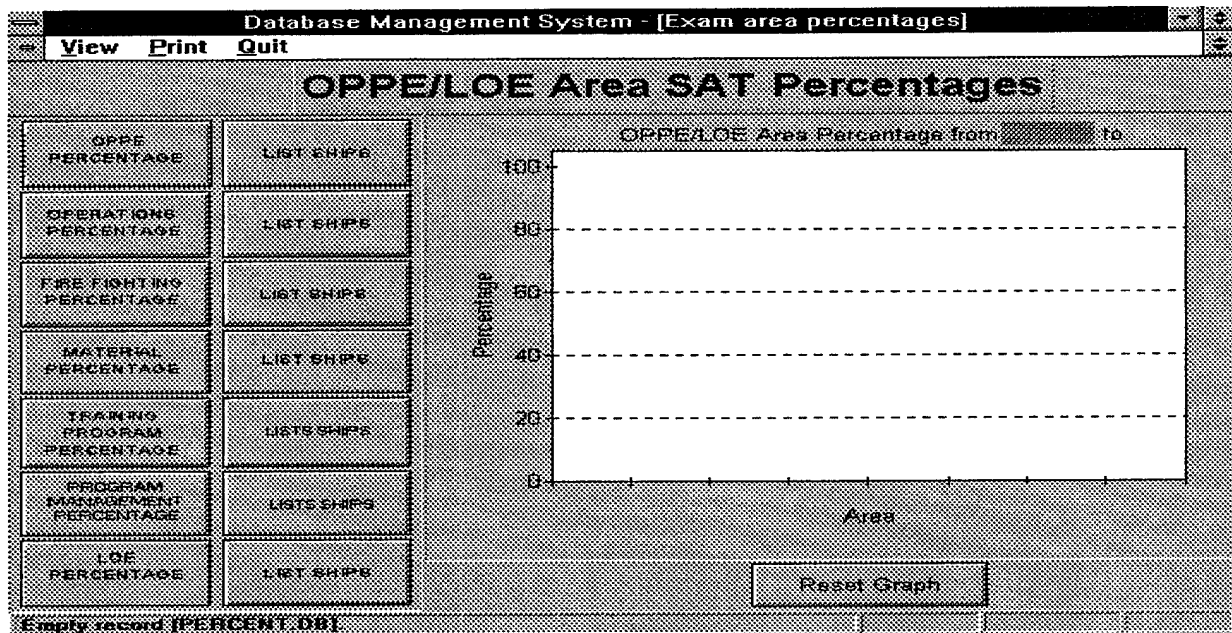


Figure 12. OPPE/LOE Area Summary query.

Percentage, and LOE percentage. The Print choice has three sub-choices. The sub-choices are Sat, Unsat, and Graph. These sub-choices enable the user to print the previously viewed report plus the displayed graph. The Quit option returns the user to the main menu.

F. Program Statistics query

The Program Statistics query (see figure 13) enables the user to graphically display each program percentage. To start the query, the user simply presses the Program push button, for example, BEARING RECORDS. The start date, stop date, propulsion type, and exam type input dialog boxes appear. The desired period, propulsion type, and exam type are entered and the query conducts the search. The next dialog boxes are informative in nature and the result is graphically displayed.

TIP: The user should keep the propulsion type, exam type and exam start and stop dates constant to get a homogenous graphical display.

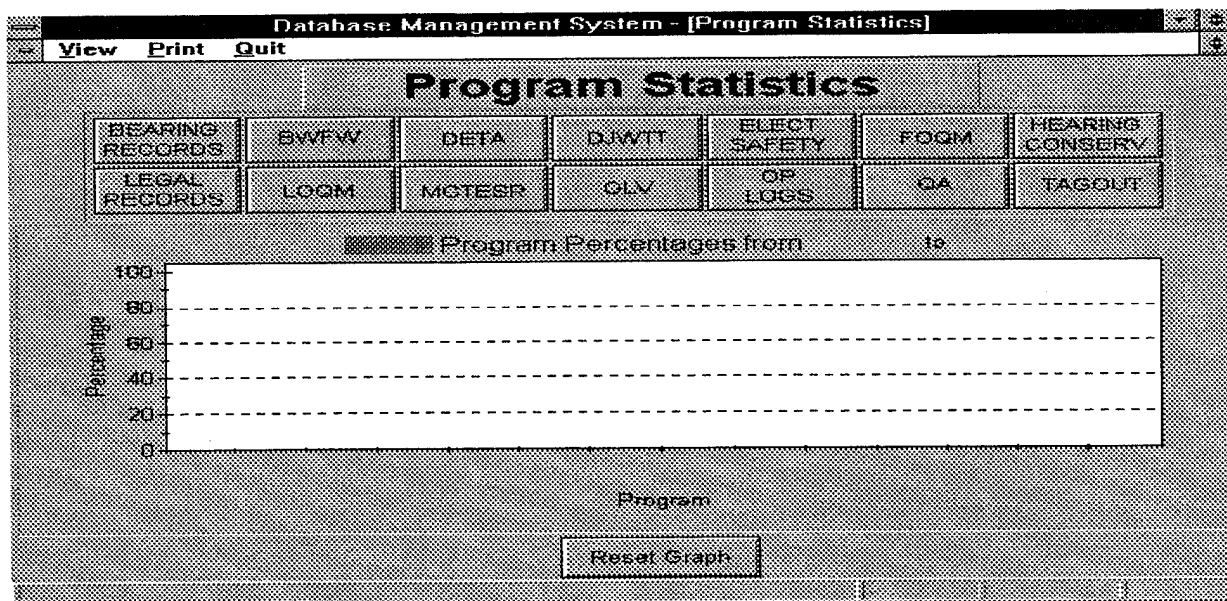


Figure 13. Program Statistics query.

The RESET GRAPH push button has been discussed earlier. The Program Statistics query main menu has three options. The three options are View, Print, and Quit. The View option has fourteen choices to view. These choices are Bearing Records, BWFW, DJWTT, Electrical Safety, FOQM, Hearing Conservation, Legal Records, LOQM, MGTESR, OLV, Operating Logs, QA, and Tagout. The Print option has fifteen choices to print. These choices are the same as the View option with the addition of printing the displayed graph. The Quit option returns the user to the main menu.

G. OPPE/LOE Monthly Summary of Activity query

The OPPE/LOE Monthly Summary of Activity query (see figure 14) enables the user to obtain information for input to the monthly reports to CINPACFLT and CNO N86P. The SUMMARY QUERY push button executes the query. The query must be executed prior to reviewing or printing the report. The user can view the report by pressing the VIEW REPORT. When the VIEW REPORT push button is pressed, a dialog box appears asking if the user has updated the Summary Query. The user can print the report by pressing the PRINT REPORT push button. Again a dialog box appears asking if the user has updated the Summary Query. The OPPE/LOE Monthly Summary of Activity query main menu has only one option. The Quit option returns the user to the main menu.

H. Evolutions and Drills query

The Evolutions and Drills query (see figure 15) enables the user to graphically display the Fire Drill satisfactory percentage. To start the query, the user simply presses the DRILL SET PERCENTAGE or the EVOLUTION SET PERCENTAGE push buttons. The start date, stop date, propulsion type, and exam type dialog boxes appear. The inputs are entered and the query conducts the search. The next dialog boxes are informative in nature and the result is graphically displayed. To compare the Evolution and Drills of different propulsion types, the user must be sure that he or she is consistent with the start and stop dates. The SHIPS and RESET GRAPH push

buttons have been discussed earlier. The Evolution and Drill query has three options. The three options are View, Print and Quit. The View option has six reports to view. The six reports are Evolutions Section 1, Drills Section 1, Evolutions Section 2, Drills Section 2, Evolutions Section 3, and Drills Section 3. The Print option has seven reports to print. The seven reports are the same as the View choices with the exception of the graph report. The Quit option returns the user to the main menu.

I. Ships with Special Situations query

The Ships with Special Situations query (see figure 16) enables the user to obtain information for input to the monthly reports to CINCPACFLT and CNO N86P. The INCOMPLETE QUERY, UNSAT QUERY, and OVERDUE QUERY push buttons execute their respective queries. The queries must be executed prior to reviewing the report or printing the report. The user can view the report by pressing the VIEW REPORT push button. When the VIEW REPORT push button is pressed, a dialog box appears asking if the user has updated the three queries. The user can print the report by pressing the PRINT REPORT push button. Again a dialog box appears asking if the user has updated the three queries. The Ships with Special Situations query main menu has only one option. The Quit option returns the user to the main menu.

The next dialog boxes are informative in nature and the result is graphically displayed. To compare the OPPE percentages, the user must ensure that he or she is consistent with the start date, stop date, and propulsion type. The RESET GRAPH push button has been discussed earlier.

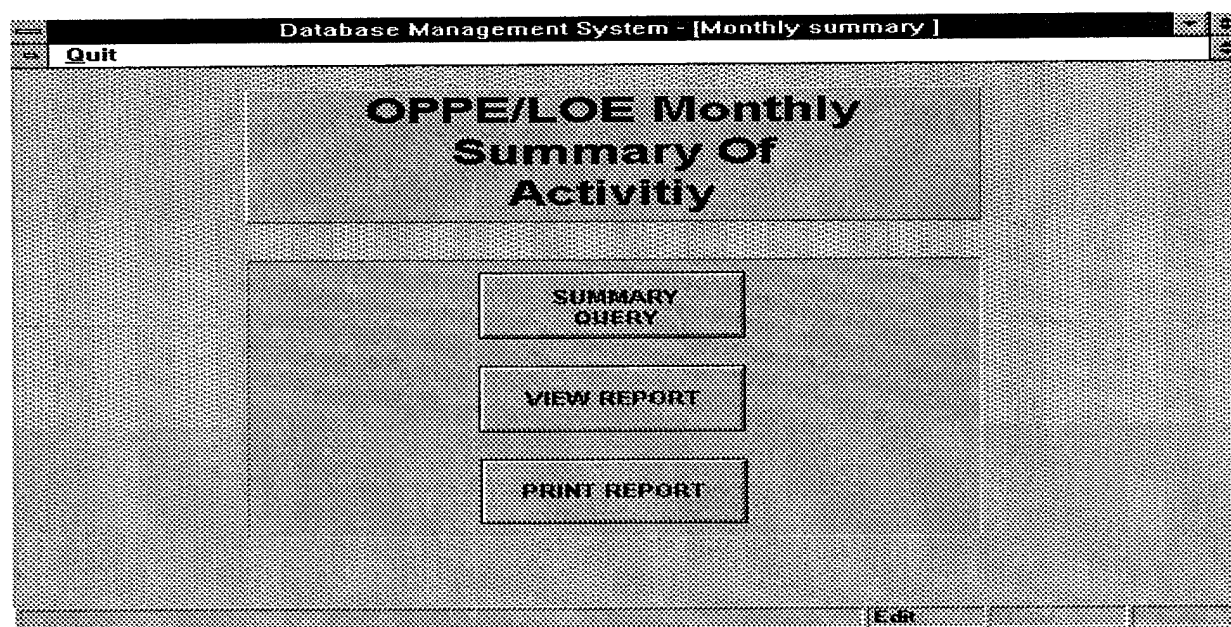


Figure 14. OPPE/LOE Monthly Summary of Activity query.

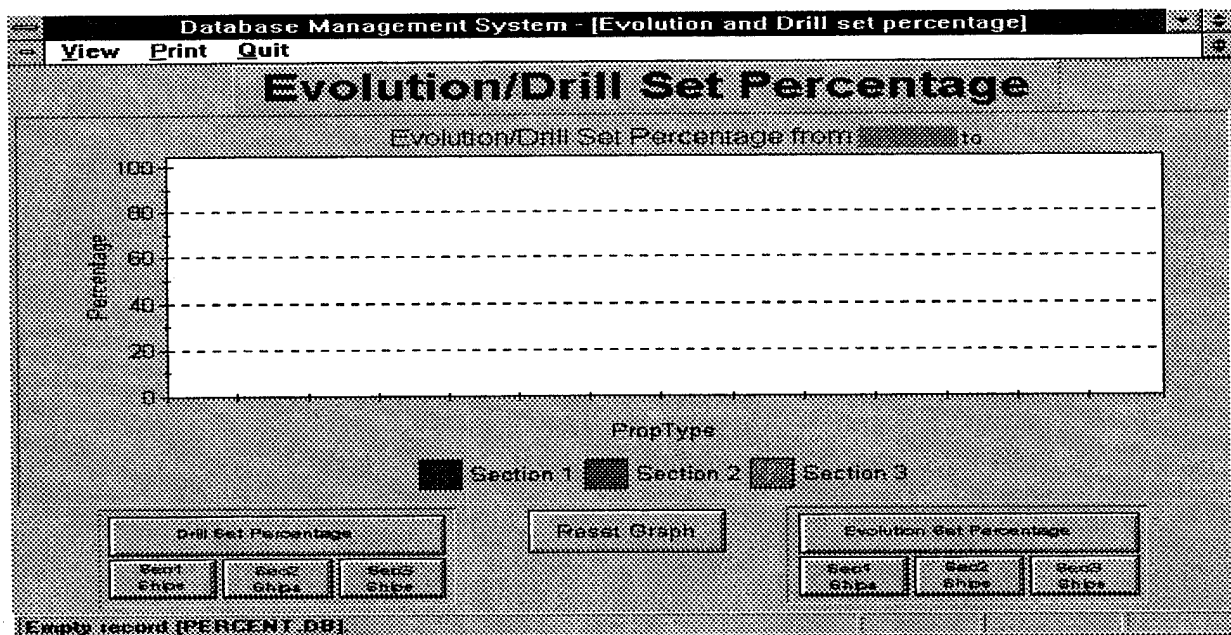


Figure 15. Evolutions and Drills query.

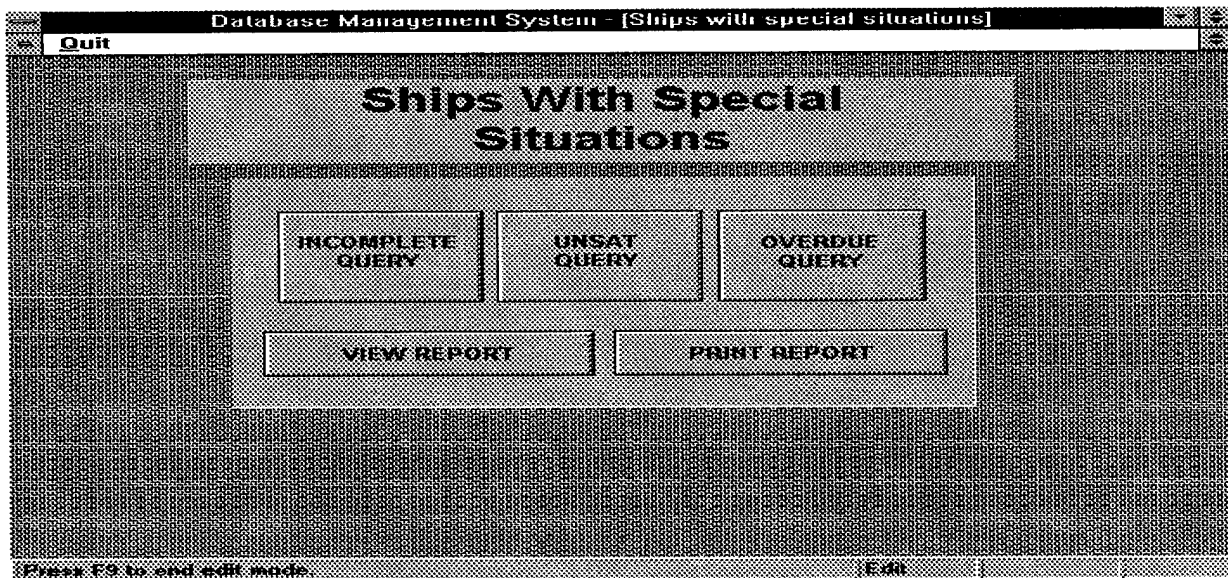


Figure 16. Ships with Special Situations query.

J. Twelve Month OPPE Cumulative Trend query

The Twelve Month OPPE Cumulative Trend query (see figure 17) enables the user to graphically display the OPPE percentages on a cumulative annual basis. To start the query, the user simply presses the OPPE Percentage push button. The start date, stop date, and propulsion type dialog boxes appear. The inputs are entered and the query conducts the search. The Twelve Month

OPPE Cumulative Trend main menu has two choices. The two choices are Print and Quit. The Print option prints the displayed graph and the Quit option returns the user to the main menu.

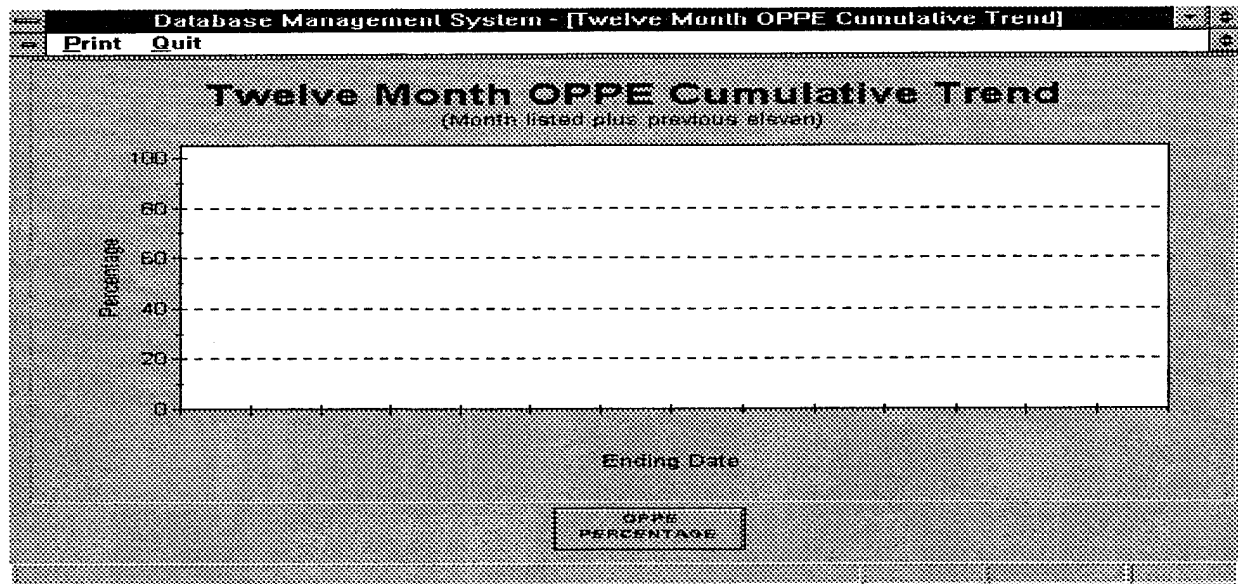


Figure 17. Twelve Month OPPE Cumulative Trend query.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2. Library, Code 052 Naval Post Graduate School Monterey, CA 93943-5002	2
3. Shu Liao, Code SM/LC Department of Systems Management Naval Post Graduate School Monterey, CA 93943-5002	1
4. Suresh Sridhar, Code SM/SR Department of Systems Management Naval Post Graduate School Monterey, CA 93943-5002	1
5. Lt. Eric Whiteman, USN CINCPACFLT Propulsion Examining Board Box 70 Naval Station San Diego, CA 92136-5296	1
6. Lt. Tony R. Encinias, USN 417 East 4th Street Walsenburg, CO 81089	1