NOTICE

**CERTAIN DATA
CONTAINED IN THIS
DOCUMENT MAY BE
DIFFICULT TO READ
IN MICROFICHE
PRODUCTS.**

SAND90-0445
Unlimited Release
Printed June, 1990

TECHNICAL SPECIFICATION FOR THE PRODUCT EVALUATION
MANAGEMENT INFORMATION SYSTEM (PREMIS) VERSION 1.1.2

Donna S. Eaton
Ronald C. Hall
John L. Orman
Janet Klamerus

ABSTRACT

This document contains the technical specification and implementation details for the Product Evaluation Management Information System (PREMIS) Version 1.1.2. This documentation does not include the requirements analysis or design information and is not intended as a user's guide. The INGRES Applications-by-Forms (ABF) software development tool was used to specify and define the modules and screens which comprise the PREMIS application. Several external procedures are called by the ABF procedures; these have been written in VAX/VMS DCL (Digital Command Language) and SQL (Standard Query Language). These specifications together with the PREMIS information model and corresponding database definition constitute the PREMIS Version 1.1.2 technical specification and implementation description presented herein.

MASTER

DISTRIBUTION

UNLIMITED

TECHNICAL SPECIFICATION FOR THE PRODUCT EVALUATION
MANAGEMENT INFORMATION SYSTEM (PREMIS) VERSION 1.1.2

TABLE OF CONTENTS

Section -----	Page -----
0. Introduction	0-1
1. PREMIS Application Description	1-1
2. Information Model	2-1
3. Database Definition	3-1
4. Screen Definitions	4-1
5. Frame and Procedure Specifications	5-1
6. Other Procedures (VAX/VMS DCL and INGRES SQL)	6-1
Appendix I -- PREMIS Data Dictionary	App. I-1
Appendix II -- PREMIS MAIN MENU HELP	App. II-1
Appendix III -- Screen Definition Standards	App. III-1
References	Ref-1

TECHNICAL SPECIFICATION FOR THE PRODUCT EVALUATION MANAGEMENT INFORMATION SYSTEM (PREMIS) VERSION 1.1.2

0. Introduction

This report documents implementation of the Product Evaluation Management Information System (PREMIS) Version 1.1.2. The PREMIS Version 1.1.2 has been released and is supported on the Product Test Data / Systems Evaluation (PTD/SE) VAXcluster by the Division 2818 PREMIS Database Project. The INGRES Relational Database Management System operating on VAX/VMS has been used as the implementation vehicle. The Version 5.0 INGRES Applications-by-Forms (ABF) software development tool [1] was used for the development on a VAX 8600 running the Version 4.7 SEVMS operating system. The implementation is based on Standard Query Language (SQL) queries from the normalized database definition generated from NIAM (Nijssen's Information Analysis Methodology) information modeling tools [2].

PREMIS Version 1.1.2 has been established to support management of Data Determination in Department 2810, Quality Assurance Evaluation in Department 7250 and Standard Company Codes in Department 2830. The integrated application replaces four disjoint data management systems: the Data Engineering Document Release System (DEDRS) in Division 2818, the Quality Assurance Management Information Program (QAMIP) in Division 7254, the TMS Center Database in Division 7254 and the Company Codes Management System in Division 2833. Planned future enhancements include expansion of the application to encompass other product management and evaluation functions, and also establishment of an appropriate subset of the functionality as a network application accessible from remote VAX computers having proper DECnet connectivity to the PTD/SE VAXcluster.

This document is not intended as a user's guide nor a design specification. Very little narrative information is included in the remainder of the report, its purpose being only to document technical details pertinent to the implementation. Standard commercial software and development tools were used to carry out the implementation. These tools are well documented by the respective vendors. Implementation details are best documented by specification of the input definitions or results from these tools. For example, specification of SQL SELECT statements provides much more precise descriptions of retrieval targets than narratives of various menu items or screens. INGRES

ABF is a comprehensive fourth-generation-language (4GL) environment for application development which allows direct creation, examination, and maintenance of the objects which comprise the application. Accordingly, the database definition, screen descriptions, and specifications for frames and procedures provide the technical description of the implementation. A brief description of the application precedes this technical detail.

1. PREMIS Application Description

PREMIS Version 1.1.2 has been released and is being maintained to support management of Data Determination in Department 2810, management of company codes in Department 2830, and Quality Assurance Evaluation in Department 7250. PREMIS Version 1.1.2 can be accessed by authorized PTD/SE VAXcluster users from VT100 or functionally equivalent terminals. The major purposes of the database and application are:

- (1) to insure early identification of all new components and systems potentially having quality assurance evaluation requirements and/or data requirements and to identify critical dates for these units and required documentation,
- (2) to aid the data engineering staff, project leaders, and supervisors in Department 2810 in managing their workload of generating data requirements documents in a timely fashion,
- (3) to aid the quality assurance engineers and supervisors in Department 7250 in managing their workload of planning for and carrying out quality assurance evaluations and generating the resulting documentation,
- (4) to track the required inclusion of data requirement drawings in the ML (material list) drawings,
- (5) to generate various status reports of data requirements activities for management and for the data engineers,
- (6) to generate various status reports of quality assurance activities for management and for the quality assurance engineers,
- (7) to maintain the record copy of three-character company codes which are used to identify and mark manufactured parts, as well as shorthand notation for companies on many official documents.

PREMIS Version 1.1.2 is an initial unclassified release, and it is intended that future enhancements will follow. Nevertheless it does represent a baseline to provide stable capability and support permanent data entry. Data entry accomplished under Version 1.1.2 will not have to be repeated for Version 2.0, or for subsequent versions of PREMIS released in the future. Further design and specification for a classified release of PREMIS Version 2.0 will commence soon after the beginning of CY 1990. User experience and feedback regarding the Version 1.1.2 software will be an input to that design. Beyond that Version 2.0 objectives are as follows:

- (1) PREMIS Version 2.0 will include enhanced documentation. Both system-level and user documentation will be developed to address particular needs which are identified as a result of experience with the Version 1.1.2 software.
- (2) The need for a facility to generate reports with variable field selections has been identified. These specifications will be completed and fully implemented as a part of PREMIS Version 2.0.
- (3) The need for a facility to include long text fields in report with word wrap capability has been identified and will be included in a future version.
- (4) Experience and feedback regarding PREMIS Version 1.1.2 will identify the need for extensions and enhancements. Several areas are currently identified for special consideration during Version 2.0 design activities. PREMIS Version 1.1.2 is unclassified. Version 2.0 is planned to be a classified implementation to accommodate input of classified backlog data and new data pertaining to the W89 weapon. PREMIS Version 2.0 will contain enhancements to support expanded features for additional terminal types (in addition to VT100-like devices) if a need for this support is determined to exist. A goal is to treat data as a corporate resource. The need will be evaluated for subsystems to accomplish automated population and maintenance of certain validation tables. Examples include validation of nomenclatures and identifications for components, and automated control and maintenance of personnel data. Enhanced validity checks are also planned to further insure continuing integrity of the PREMIS database. Finally, enhanced application functionality will be developed to address particular user needs which may be identified through use of Version 1.1.2.

Complete input screens are included in PREMIS Version 1.1.2 to support data entry into all application-level tables. Comprehensive query capabilities based upon SQL SELECT statements are embedded in the application. The top-level application structure is shown in Figures 1 through 3. These figures illustrate the hierarchy of frames (which associate a screen with appropriate software procedures) to carry out the functions selectable through the menus. TOPFRAME provides a high-level help function to acquaint new users with the system. TOPFRAME also provides the entry paths into one of three main sections of the application: DATA to access the main menu for data engineering, COMPMAN to access the main menu for maintenance and retrieval of company codes, and QAMAIN to access the main menu for quality assurance. QAMAIN is further subdivided into QARMAIN to accomplish database retrievals and QAEMAIN to accomplish data entry. Beyond this, the user can traverse a path through the hierarchy of frames to accomplish whatever retrieval or update functions are required and authorized. The structure and organization of the implemented database and application used to support this functionality constitute the focus of the remaining description contained in this report.

PREMIS

Application Functional Flow

FIGURE 1

PREMIS

Application Functional Flow

FIGURE 2

PREMIS

Application Functional Flow

FIGURE 3

2. Information Model

The Information Model is represented by an Information Structure Diagram. This model allows the analyst to represent data and its relationships to other data in a format that is easily understood and evaluated.

Formal data modeling techniques have been employed in the PREMIS Project in order to develop a meaningful and usable database with the least amount of labor. This section depicts the various labels (lexical object types, i.e. the data occurrences), the entities (nonlexical object types, i.e. the abstract data attributes), and the relationships among them. The entities are shown as circular shapes made with solid lines. The labels are shown as circles with dotted lines. A variety of relationships are represented with lines between the entities, other entities, and labels.

The complete model, drawing number R12818, Issue C is shown in Figure 4 with its component parts labeled. Each component part appears in a larger drawing, Figures 5-14, and Figure 15 lists all of the repeated entity types and their numbers for reference purposes.

Figure 4. PREMIS Information Structure Diagram, Drawing R12818, Issue C

Figure 6. Weapon

Figure 7. Component

Figure 8. Person

Figure 9. Company

Figure 10. TMS (Tool-Made Sample)

Figure 11. Equipment

Figure 12. QER (Qualification Evaluation Release)

Figure 13. Data Document

Figure 14. Characteristics and Development

Figure 15. REPEATED ENTITY TYPES

(1)	PERSON
(2)	DATA DOCUMENT
(3)	MFG INST
(4)	ENGR EVAL RELS
(5)	QUAL EVAL RELS
(6)	MANUFACTURER
(7)	PA DATA ENGR
(8)	UNIT
(9)	COMPONENT
(10)	DA EMPL
(11)	DA ENGR
(12)	PA ENGR
(13)	NWC AGENCY
(14)	WEAPON
(15)	WEAPON PGM
(16)	EQUIPMENT
(17)	EQ ACTIVITY
(18)	ISSUE
(19)	DOCUMENT
(20)	DATE

3. Database Definition

The formal data model presented in Section 2 is translated into the database definition contained in this section. This relational definition, produced directly from the data model, is in fifth normal form. Fifth normal form implies that the structure is elementary because the same information content (data and relationships) cannot be reconstructed from smaller record types derived from the original (fifth normal form) representation. Such an elementary representation is particularly productive in the PREMIS environment because of the extreme flexibility it offers when making enhancements and releasing new expanded versions to accommodate additional requirements. Our plan is to release future enhancements which can be integrated to access the database as a corporate resource. These extensions can be made without rendering prior functionality inoperable.

The database definition has been used to create an INGRES relational database. The SQL definitions produce the tables which comprise the database. The PREMIS model was implemented using 60 tables, 27 of which are validation tables. The validation tables contain the set of legitimate data values which can be input to the database to represent occurrences of data for which there is a known and restricted domain. Following are the SQL statements which define and create the PREMIS database.

/*

PREMIS.SQL

*/

```

CREATE TABLE ADDRESS (ADDRESS_ID          varchar(9) ,
                        STRT_ADDR1          varchar(34) ,
                        STRT_ADDR2          varchar(34) ,
                        CITY_NAME           varchar(20) ,
                        STATE_CODE          varchar(2) ,
                        ZIP_CODE            varchar(9) ,
                        COUNTRY_NAME        varchar(20)); \P\G\R

CREATE TABLE C_DA_PRD_EGR (UNIT_DA_SUF     varchar(3) ,
                             UNIT_STEM      varchar(7) ,
                             DA_PROD_ENGR    varchar(6) ); \P\G\R

CREATE TABLE C_PA_DTA_EGR (UNIT_DA_SUF     varchar(3) ,
                             UNIT_STEM      varchar(7) ,
                             PA_DATA_ENGR    varchar(9) ); \P\G\R

CREATE TABLE CHAR_DEV_RPT (DOC_NUM         varchar(8) ,
                             DOC_TYPE       varchar(4) ,
                             DOC_ISS_NUM    varchar(2) ,
                             PERCENT_COMP   number(3) ,
                             PUBLISHED_DT   DATE ,
                             EST_DRAFT_DT   DATE ,
                             EST_PUB_DT     DATE ,
                             FINL_DRFT_DT   DATE ); \P\G\R

CREATE TABLE COMPONENT (UNIT_DA_SUF     varchar(3) ,
                          UNIT_STEM      varchar(7) ,
                          COM_NOM_NUM     varchar(8) ,
                          COM_NOM_TYPE    varchar(2) ); \P\G\R

CREATE TABLE COMPANY (MFG_CODE          varchar(3) ,
                        COMPANY_NAME      varchar(34) ,
                        PHONE_NUM         varchar(10) ,
                        ALT_MFG_CODE      varchar(2) ,
                        COMP_REQ          varchar(9) ,
                        ENTRY_DATE        date ,
                        SOURCE             varchar(34) ,
                        COMPANY_NAM2      varchar(34) ,
                        NOTES             varchar(34) ); \P\G\R

CREATE TABLE COMP_HIST (CURRENT_CODE     varchar(3) ,
                         NEW-MFG_CODE     varchar(3) ,
                         OLD_MFG_CODE     varchar(3) ,
                         OLD_NAME         varchar(34) ); \P\G\R

CREATE TABLE STATE (STATE_CODE          varchar(2) ); \P\G\R
CREATE TABLE PROVINCE (PROV_CODE        varchar(2) ); \P\G\R
CREATE TABLE DA_DESN_ENGR (EMPL_NUM     varchar(6) ); \P\G\R
CREATE TABLE DA_EMPL (EMPL_NUM          varchar(6) ,
                        ID_NUM            varchar(9) ); \P\G\R

CREATE TABLE DA_ENGR (EMPL_NUM          varchar(6) ); \P\G\R
CREATE TABLE DA_ELEC_ENGR (EMPL_NUM     varchar(6) ); \P\G\R
CREATE TABLE DA_MECH_ENGR (EMPL_NUM     varchar(6) ); \P\G\R
CREATE TABLE DA_N_SF_ENGR (DA_N_SF_ENGR varchar(6) ); \P\G\R
CREATE TABLE DA_PROD_ENGR (EMPL_NUM     varchar(6) ); \P\G\R
CREATE TABLE DA_QA_ENGR (EMPL_NUM       varchar(6) ); \P\G\R
CREATE TABLE DA_REL_ENGR (EMPL_NUM      varchar(6) ); \P\G\R
CREATE TABLE DATA_DOCUMNT (CER_DT      DATE ,
                              DTER_DT     DATE ,
                              ML_DT       DATE ,
                              DOC_NUM     varchar(8) ,
                              DOC_TYPE    varchar(4) ,
                              DOC_ISS_NUM varchar(2) ); \P\G\R

CREATE TABLE DESIGN_AGENCY (MFG_CODE    varchar(3) ); \P\G\R

```

CREATE TABLE DIVISION
CREATE TABLE DOC_ROUGH_DT

CREATE TABLE DOC_RQ_ML_DT

CREATE TABLE DOCUMENT

CREATE TABLE EER_STATUS
CREATE TABLE ENG_EVAL_REL

CREATE TABLE EQ_ACT_EER

CREATE TABLE EQ_ACTIVITY

CREATE TABLE EQ_INST

CREATE TABLE EQ_PQ_TEAM

CREATE TABLE EQUIPMENT

CREATE TABLE FILE_CABINET
CREATE TABLE GAGE

CREATE TABLE GOZINTO

CREATE TABLE MANUFACTURER
CREATE TABLE MFG_EQP_INST

CREATE TABLE MFG_INST

(ORGN_NUM
(R_DRAFT_DT
DOC_NUM
DOC_TYPE
DOC_ISS_NUM
(REQ_ML_DT
DOC_NUM
DOC_TYPE
DOC_ISS_NUM
(DOC_NUM
DOC_TYPE
DOC_ISS_NUM
EST_COMP_DT
REQUIRED_DT
(EER_STATUS
(RSLT_RPT_FL
DOC_NUM
DOC_TYPE
DOC_ISS_NUM
(HRD_SFT_FLAG
DOC_NUM
DOC_TYPE
DOC_ISS_NUM
EQP_INST_NUM
(QA_ACTV_TYPE
HRD_SFT_FLAG
EQP_INST_NUM
(DOC_ISS_NUM
DOC_NUM
DOC_TYPE
HRD_SFT_FLAG
EQP_INST_NUM
(MEMBER_TITLE
ID_NUM
HRD_SFT_FLAG
EQP_INST_NUM
(EQUIP_NUM
EQUIP_TYPE
SERIAL_NUM
(FILE_CAB_LOC
(EQUIP_NUM
EQUIP_TYPE
SERIAL_NUM
(UNIT_DA_SUF
UNIT_STEM
PART_DA_SUF
PART_STEM
(MFG_CODE
(EQUIP_NUM
EQUIP_TYPE
SERIAL_NUM
EQP_INST_NUM
MFG_INST_NUM
(UNIT_DA_SUF
UNIT_STEM
MFG_CODE
WPN_MOD_NUM
WPN_PGM_CODE
FRST_PROD_DT
MFG_INST_NUM

vchar(6)); |P|G|R
DATE ,
vchar(8) ,
vchar(4) ,
vchar(2)); |P|G|R
DATE ,
vchar(8) ,
vchar(4) ,
vchar(2)); |P|G|R
vchar(8) ,
vchar(4) ,
vchar(2) ,
DATE ,
DATE); |P|G|R
vchar(4)); |P|G|R
vchar(1) ,
vchar(8) ,
vchar(4) ,
vchar(2)); |P|G|R
vchar(1) ,
vchar(8) ,
vchar(4) ,
vchar(2) ,
vchar(4)); |P|G|R
vchar(4) ,
vchar(1) ,
vchar(4)); |P|G|R
vchar(2) ,
vchar(8) ,
vchar(4) ,
vchar(1) ,
vchar(4)); |P|G|R
vchar(16) ,
vchar(9) ,
vchar(1) ,
vchar(4)); |P|G|R
vchar(4) ,
vchar(2) ,
vchar(2)); |P|G|R
vchar(9)); |P|G|R
vchar(4) ,
vchar(2) ,
vchar(2)); |P|G|R
vchar(3) ,
vchar(7) ,
vchar(3) ,
vchar(7)); |P|G|R
vchar(3)); |P|G|R
vchar(4) ,
vchar(2) ,
vchar(2) ,
vchar(4) ,
vchar(4)); |P|G|R
vchar(3) ,
vchar(7) ,
vchar(3) ,
vchar(2) ,
vchar(4) ,
DATE ,
vchar(4)); |P|G|R

CREATE TABLE NWC_AGENCY	(MFG_CODE	vchar(3)); \P\G\R
CREATE TABLE PA_DATA_ENGR	(ID_NUM	vchar(9)); \P\G\R
CREATE TABLE PA_EMPL	(ID_NUM	vchar(9)); \P\G\R
CREATE TABLE PA_ENGR	(ID_NUM	vchar(9)); \P\G\R
CREATE TABLE PA_PROD_ENGR	(ID_NUM	vchar(9)); \P\G\R
CREATE TABLE PA_QC_ENGR	(ID_NUM	vchar(9)); \P\G\R
CREATE TABLE PART_NUM	(UNIT_DA_SUF	vchar(3) ,
	UNIT_STEM	vchar(7)); \P\G\R
CREATE TABLE PERSON	(FIRST_NAME	vchar(12) ,
	ID_NUM	vchar(9) ,
	LAST_NAME	vchar(15) ,
	MAIL_STOP	vchar(6) ,
	MIDDLE_INIT	vchar(1) ,
	PHONE_NUM	vchar(10) ,
	MFG_CODE	vchar(3) ,
	ASND_ORGN	vchar(6)); \P\G\R
CREATE TABLE PROD_AGENCY	(MFG_CODE	vchar(3)); \P\G\R
CREATE TABLE PROGRAM	(WPN_PGM_CODE	vchar(4)); \P\G\R
CREATE TABLE QA_ENGR_RELS	(MESSAGE_NUM	vchar(5) ,
	TWX_RECEIPT	vchar(1) ,
	COM_CTR_CNT	vchar(6) ,
	ISSUE_DT	DATE ,
	TRANS_COM_DT	DATE ,
	FILE_CAB_LOC	vchar(9) ,
	M11_DIR_NAME	vchar(6) ,
	DOC_NUM	vchar(8) ,
	DOC_TYPE	vchar(4) ,
	DOC_ISS_NUM	vchar(2) ,
	MFG_INST_NUM	vchar(4) ,
	STATUS_CODE	vchar(4)); \P\G\R
CREATE TABLE QER_PROBLEMS	(PROBLEM_DESC	vchar(30) ,
	SOLVED_DT	DATE) ,
	DOC_NUM	vchar(8) ,
	DOC_TYPE	vchar(4) ,
	DOC_ISS_NUM	vchar(2)); \P\G\R
CREATE TABLE QL_EVAL_RELS	(DOC_NUM	vchar(8) ,
	DOC_TYPE	vchar(4) ,
	DOC_ISS_NUM	vchar(2)); \P\G\R
CREATE TABLE SUPPLIER	(MFG_CODE	vchar(3)); \P\G\R
CREATE TABLE TESTER	(EQUIP_NUM	vchar(4) ,
	EQUIP_TYPE	vchar(2) ,
	SERIAL_NUM	vchar(2)); \P\G\R
CREATE TABLE TMS_ACTIVITY	(DOC_NUM	vchar(8) ,
	DOC_TYPE	vchar(4) ,
	DOC_ISS_NUM	vchar(2) ,
	QA_ACTV_TYPE	vchar(4) ,
	HRD_SFT_FLAG	vchar(1) ,
	MFG_INST_NUM	vchar(4)); \P\G\R
CREATE TABLE TMS_INST	(DOC_ISS_NUM	vchar(2) ,
	DOC_NUM	vchar(8) ,
	DOC_TYPE	vchar(4) ,
	HRD_SFT_FLAG	vchar(1) ,
	MFG_INST_NUM	vchar(4)); \P\G\R
CREATE TABLE TMS_PQ_TEAM	(MEMBER_TITLE	vchar(16) ,
	ID_NUM	vchar(9) ,
	HRD_SFT_FLAG	vchar(1) ,
	MFG_INST_NUM	vchar(4)); \P\G\R
CREATE TABLE UNIT	(CONFIG	vchar(4) ,
	UNIT_DA_SUF	vchar(3) ,
	UNIT_STEM	vchar(7) ,

CREATE TABLE UNIT_DESC

CREATE TABLE UNIT_DOC

CREATE TABLE WEAPON

CREATE TABLE WEAPON_NOM

CREATE TABLE WEAPON_PGM

DA_DATA ENGR
DA_QA ENGR
DA_REL ENGR
PA_PROD ENGR
PA_QC ENGR
DSN_MFG_CODE
PRD_MFG_CODE
DOC_NUM
DOC_TYPE
DOC_ISS_NUM
DA_DESN ENGR
DA_N_SF ENGR
(UNIT_DESC
UNIT_DA_SUF
UNIT_STEM
(DOC_NUM
DOC_TYPE
UNIT_DA_SUF
UNIT_STEM
DOC_ISS_NUM
(UNIT_DA_SUF
UNIT_STEM
WPN_PGM_CODE
E_DESN ENGR
M_DESN ENGR
WPN_MOD_NUM
WPN_TEST_CFG
(WPN_PGM_CODE
WPN_MOD_NUM
WPN_TEST_CFG
(WPN_MOD_NUM
WPN_MI_NUM
WPN_PGM_CODE

vchar(6) ,
vchar(6) ,
vchar(6) ,
vchar(9) ,
vchar(9) ,
vchar(3) ,
vchar(3) ,
vchar(8) ,
vchar(4) ,
vchar(2) ,
vchar(6) ,
vchar(6)); \P\G\R
vchar(30) ,
vchar(3) ,
vchar(7)); \P\G\R
vchar(8) ,
vchar(4) ,
vchar(3) ,
vchar(7) ,
vchar(2)); \P\G\R
vchar(3) ,
vchar(7) ,
vchar(4) ,
vchar(6) ,
vchar(6) ,
vchar(2) ,
vchar(7)); \P\G\R
vchar(4) ,
vchar(2) ,
vchar(7)); \P\G\R
vchar(2) ,
vchar(10) ,
vchar(4)); \P\G\R

PREMIS Database Requirements

A description of each table and its columns follows. In addition, the screen or frame where the values are entered by a customer is shown as the update/entry screen. If tables are not populated by data entered by customers in the PREMIS application (but are entered by the DBA or other sources), this is indicated by "----" for the update/entry screen. Note also that some tables are not yet used by the PREMIS application, but could be included in referential integrity enforcement in the future.

DATASET: ADDRESS (Store the addresses of companies stored in the DATABASE)
UPDATE/ENTRY SCREEN: COMPF0RM3

Variable Name	data type	size
KEY ADDRESS_ID	character	09
STRT_ADDR1	character	34
STRT_ADDR2	character	34
CITY_NAME	character	20
STATE_CODE	character	02
ZIP_CODE	character	09
COUNTRY_NAME	character	20

DATASET: C_DA_PRD_EGR (Show the Sandia Product Engineer for the component)
(NOT USED)

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
DA_PROD_ENGR	character	06

DATASET: C_PA_DTA_EGR (Show the production agency data engineer for the unit)
(NOT USED)

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
PA_DATA_ENGR	character	09

DATASET: CHAR_DEV_RPT (Characteristic and Development Report)
UPDATE/ENTRY SCREEN: CHARDEV

Variable Name	data type	size
KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
EST_DRAFT_DT	date	
FINL_DRFT_DT	date	
EST_PUB_DT	date	
PUBLISHED_DT	date	
PERCENT_COMP	integer	02

note: a two byte integer can store a number as large as 127.

DATASET: COMPANY (Company File)
 UPDATE/ENTRY SCREEN: COMPFORM3

Variable Name	data type	size
KEY MFG_CODE	character	03
COMPANY_NAME	character	34
PHONE_NUM	character	10
ALT MFG_CODE	character	02
COMP_REQ	character	09
ENTRY_DATE	date	
SOURCE	character	34
COMPANY_NAME2	character	34
NOTES	character	34

DATASET: COMPANY_HIST (Company History File)
 UPDATE/ENTRY SCREEN: COMPFORM3

Variable Name	data type	size
KEY CURRENT_CODE	character	03
NEW_MFG_CODE	character	03
OLD_MFG_CODE	character	03
OLD_NAME	character	34

DATASET: COMPANY_SYN (Company Synonym File)
 UPDATE/ENTRY SCREEN: ----

Variable Name	data type	size
KEY MFG_CODE	character	03
COMPANY_NAME	character	34

DATASET: COMPONENT (Component Level Unit)
 UPDATE/ENTRY SCREEN: ENTER, NEWPART(MENU CHOICE)

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
COM_NOM_TYPE	character	02
COM_NOM_NUM	character	08

DATASET: DATA DOCUMNT (Data Documents)
 UPDATE/ENTRY SCREEN: NEWDOC, DELETEDOC

Variable Name	data type	size
KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
CER_DT	date	
DTER_DT	date	
ML_DT	date	

DATASET: DDC_ROUGH_DT (Show the rough draft date for documents)

UPDATE/ENTRY SCREEN: NEWDOC, DELETEDOC

Variable Name	data type	size
KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
R_DRAFT_DT	date	

DATASET: DDC_RQ_ML_DT (Show the date we requested the document to be placed on the material list.)

UPDATE/ENTRY SCREEN: NEWDOC, DELETEDOC

Variable Name	data type	size
KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
REQ_ML_DT	date	

DATASET: DOCUMENT

UPDATE/ENTRY SCREEN: NEWDOC, DELETEDOC

Variable Name	data type	size
KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
EST_COMP_DT	date	
REQUIRED_DT	date	

DATASET: ENG_EVAL_REL (Engineering Evaluation Release Documents)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
RSLT_RPT_FL	character	01

DATASET: EQ_ACT_EER (Shows which eq activity is planned in the EER)

UPDATE/ENTRY SCREEN: -----

Variable Name	data type	size
KEY EQP_INST_NUM	character	04
KEY HRD_SFT_FLAG	character	01
DOC_TYPE	character	04
DOC_NUM	character	08
DOC_ISS_NUM	character	02

DATASET: EQ ACTIVITY (Equipment Qualification Activity)

UPDATE/ENTRY SCREEN: ----

Variable Name	data type	size
KEY EQP_INST_NUM	character	04
KEY HRD_SFT_FLAG	character	01
QA_ACTV_TYPE	character	04

DATASET: EQ_INST (Equipment Instance)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY DOC_ISS_NUM	character	02
KEY HRD_SFT_FLAG	character	01
KEY EQP_INST_NUM	character	04
DOC_TYPE	character	04
DOC_NUM	character	08

DATASET: EQ_PQ_TEAM (Ad hoc members of an Equipment Qualification PQ Team)

UPDATE/ENTRY SCREEN: PQTEAM

Variable Name	data type	size
KEY ID_NUM	character	09
KEY HRD_SFT_FLAG	character	01
KEY EQP_INST_NUM	character	04
MEMBER_TITLE	character	16

DATASET: EQUIPMENT (Pieces of Equipment usually a gage or a tester)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY EQUIP_TYPE	character	02
KEY EQUIP_NUM	character	04
KEY SERIAL_NUM	character	02

DATASET: GOZINTO (Shows which components are contained in each system)

UPDATE/ENTRY SCREEN: NEWPART

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
KEY PART_STEM	character	07
KEY PART_DA_SUF	character	03

DATASET: MFG_EQP_INST (An instance of a piece of equipment manufactured by a
 UPDATE/ENTRY SCREEN: PQTEAM, TMSCNTR
 manufacturer)

Variable Name	data type	size
KEY EQP_INST_NUM	character	04
EQUIP_TYPE	character	02
EQUIP_NUM	character	04
SERIAL_NUM	character	02
MFG_INST_NUM	character	04

DATASET: MFG_INST (An instance of a component being manufactured by a specific
 Manufacturer).

UPDATE/ENTRY SCREEN: PQTEAM, TMSCNTR, NEWPART, ENTER

Variable Name	data type	size
KEY MFG_INST_NUM	character	04
MFG_CODE	character	03
WPN_PGM_CODE	character	04
WPN_MOD_NUM	character	03
UNIT_STEM	character	07
UNIT_DA_SUF	character	03
FRST_PROD_DT	date	

NWC_AGENCY (Nuclear Weapons Complex Agency)

UPDATE/ENTRY SCREEN: ----

Variable Name	data type	size
KEY MFG_CODE	character	03
DOE_CODE	character	02

DATASET: PERSON (Person's Name, Organization, and Company)

UPDATE/ENTRY SCREEN: ----

Variable Name	data type	size
KEY ID_NUM	character	09
MFG_CODE	character	03
ASND_ORGN	character	06
FIRST_NAME	character	12
MIDDL_INIT	character	01
LAST_NAME	character	15
MAIL_STOP	character	06
PHONE_NUM	character	10

DATASET: QA ENGR RELS (Quality Assurance Engineering Release)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
KEY DOC_TYPE	character	04
TWX_RECEIPT	character	01
ISSUE_DT	date	
TRANS_COM_DT	date	
FILE_CAB_LOC	character	09
M11_DIR_NAME	character	07
COM_CTR_CNT	character	06
MESSAGE_NUM	character	05
STATUS_CODE	character	04

DATASET: QER_PROBLEMS (Problems found while doing a QER which must be solved before the next release.)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
KEY DOC_TYPE	character	04
PROBLEM_DESC	character	30
SOLVED_DT	date	

DATASET: TMS_ACTIVITY (Tool Made Sample Activity)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY MFG_INST_NUM	character	04
KEY HRD_SFT_FLAG	character	01
QA_ACTV_TYPE	character	04
DOC_TYPE	character	04
DOC_NUM	character	08
DOC_ISS_NUM	character	02

DATASET: TMS_INST (An instance of a Tool Made Sample)

UPDATE/ENTRY SCREEN: TMSCNTR

Variable Name	data type	size
KEY DOC_ISS_NUM	character	02
KEY MFG_INST_NUM	character	04
KEY HRD_SFT_FLAG	character	01
DOC_TYPE	character	04
DOC_NUM	character	08

DATASET: TMS_PQ_TEAM (Ad hoc members of a Tool Made Sample PQ Team)

UPDATE/ENTRY SCREEN: PQTEAM

Variable Name	data type	size
KEY ID_NUM	character	09
KEY MFG_INST_NUM	character	04
KEY HRD_SFT_FLAG	character	01
MEMBER_TITLE	character	16

DATASET: UNIT (A component or a weapon level unit)

UPDATE/ENTRY SCREEN: ENTER, NEWPART, PQTEAM

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
DOC_NUM	character	08
DOC_ISS_NUM	character	02
DOC_TYPE	character	04
CONFIG	character	01
DA_DATA_ENGR	character	06
DA_DESN_ENGR	character	06
DA_N_SF_ENGR	character	06
DA_QA_ENGR	character	06
DA_REL_ENGR	character	06
DSN_MFG_CODE	character	03
PA_PROD_ENGR	character	09
PA_QC_ENGR	character	09
PRD_MFG_CODE	character	03

DATASET: UNIT_DESC (Shows the description of a unit)

UPDATE/ENTRY SCREEN: NEWPART

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
UNIT_DESC	character	30

DATASET: UNIT_DOC (Shows which documents are required for each unit)

UPDATE/ENTRY SCREEN: UNITDOC

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
DOC_TYPE	character	04
DOC_NUM	character	08
DOC_ISS_NUM	character	02

DATASET: WEAPON

UPDATE/ENTRY SCREEN: ---

Variable Name	data type	size
KEY UNIT_STEM	character	07
KEY UNIT_DA_SUF	character	03
E_DESN_ENGR	character	06
M_DESN_ENGR	character	06
WPN_PGM_CODE	character	04
WPN_TEST_CFG	character	07
WPN_MOD_NUM	character	02

DATASET: WEAPON_PGM (Weapon Program)

UPDATE/ENTRY SCREEN: ----

Variable Name	data type	size
KEY WPN_PGM_CODE	character	04
KEY WPN_MOD_NUM	character	02
WPN_MI_NUM	character	10

VERIFICATION TABLES

COM_NDM_TYPE (Component Type)

Variable Name	data type	size
KEY TYPE	character	06

DA_CT_DS_EGR (Design Agency Component Design Engineer)

Variable Name	data type	size
KEY EMPL_NUM	character	06

DA_DATA_ENGR (Design Agency Data Engineer)

Variable Name	data type	size
KEY EMPL_NUM	character	06

DA_DESN_ENGR (Design Agency Design Engineer)

Variable Name	data type	size
KEY EMPL_NUM	character	06

DA_DESN_ORGN (Design Agency Design Organization)

Variable Name	data type	size
KEY DIVISION	character	06

DA_ELEC_ENGR (Design Agency Lead Electrical Engineer)

Variable Name	data type	size
KEY EMPL_NUM	character	06

DA_EMPL (Design Agency Employee)

Variable Name	data type	size
KEY EMPL_NUM	character	06
ID_NUM	character	09

DA_ENGR (Design Agency Engineer)

Variable Name	data type	size
KEY EMPL_NUM	character	06

DA_MECH_ENGR (Design Agency Lead Mechanical Engineer)

Variable Name	data type	size
KEY EMPL_NUM	character	06

DA_N_SF_ENGR (Design Agency Nuclear Safety Engineer)		
Variable Name	data type	size

KEY EMPL_NUM	character	06

DA_PROD_ENGR (Design Agency Product Engineer)		
Variable Name	data type	size

KEY EMPL_NUM	character	06

DA_QA_ENGR (Design Agency Quality Assurance Engineer)		
Variable Name	data type	size

KEY EMPL_NUM	character	06

DA_REL_ENGR (Design Agency Reliability Engineer)		
Variable Name	data type	size

KEY EMPL_NUM	character	06

DESIGN_AGENCY (Design Agency)		
Variable Name	data type	size

KEY MFG_CODE	character	03

EER_STATUS (Valid statuses)		
Variable Name	data type	size

KEY EER_STATUS	character	04

FILE_CABINET (Valid File cabinet locations)		
Variable Name	data type	size

KEY FILE_CAB_LOC	character	09

GAGE (A piece of equipment with EQUIP_TYPE = "GA" or "UA")		
Variable Name	data type	size

KEY EQUIP_TYPE	character	02
KEY EQUIP_NUM	character	04
KEY SERIAL_NUM	character	02

MANUFACTURER (Can be a production agency, a supplier or in rare cases a design agency)		
Variable Name	data type	size

KEY MFG_CODE	character	03

PA_DATA_ENGR (Production Agency Data Engineer)		
Variable Name	data type	size

KEY ID_NUM	character	09

PA_EMPL (Production Agency Employee)		
Variable Name	data type	size

KEY ID_NUM	character	09
PA_PROD_ENGR (Production Agency Product Engineer)		
Variable Name	data type	size

KEY ID_NUM	character	09
PA_QC_ENGR (Production Agency Quality Control Engineer)		
Variable Name	data type	size

KEY ID_NUM	character	09
PROD_AGENCY (Production Agency)		
Variable Name	data type	size

KEY MFG_CODE	character	03
PROVINCE (Canadian Province Codes)		
Variable Name	data type	size

KEY PROV_CODE	character	04
QL_EVAL_RELS (Quality Evaluation Release)		
Variable Name	data type	size

KEY DOC_TYPE	character	04
KEY DOC_NUM	character	08
KEY DOC_ISS_NUM	character	02
STATE (Table of valid state codes)		
Variable Name	data type	size

KEY STATE_CODE	character	02
SUPPLIER		
Variable Name	data type	size

KEY MFG_CODE	character	03
TESTER (A piece of equipment with EQUIP_TYPE = "PT")		
Variable Name	data type	size

KEY EQUIP_TYPE	character	02
KEY EQUIP_NUM	character	06
KEY SERIAL_NUM	character	02
WEAPON_NOM (Weapon Nomenclature)		
Variable Name	data type	size

KEY WPN_PGM_CODE	character	04
KEY WPN_TEST_CFG	character	07
KEY WPN_MOD_NUM	character	02

VIEWS

NOMEN_VIEW (Used to report QA information sorted by NOMENCLATURE, PART_NUMBER)

View Name	data type	size	Database Name
NOMENCLATURE	character	10	COM_NOM_TYPE + COM_NOM_NUM
PART_NUMBER	character	07	UNIT_STEM
QA_ORGN	character	06	ASND_ORGN
QA_ENGINEER	character	15	LAST_NAME
PA	character	02	DOE_CODE
WPN_PGM	character	08	WPN_PGM_CODE + '-' + WPN_MOD_NUM
DESCRIPTION	character	30	UNIT_DESC

NWC_COM_VIEW (Used to associate DOE Codes with Company names)

View Name	data type	size	Database Name
COMPANY_NAME	character	34	COMPANY_NAME
MFG_CODE	character	03	MFG_CODE
DOE_CODE	character	02	DOE_CODE
CITY_NAME	character	20	CITY_NAME
STATE	character	02	STATE_CODE

PART_NO_VIEW (Used to report QA information sorted by PART_NUMBER)

View Name	data type	size	Database Name
PART_NUMBER	character	07	UNIT_STEM
NOMENCLATURE	character	10	COM_NOM_TYPE + COM_NOM_NUM
QA_ORGN	character	06	ASND_ORGN
QA_ENGINEER	character	15	LAST_NAME
PA	character	02	DOE_CODE
WPN_PGM	character	08	WPN_PGM_CODE + '-' + WPN_MOD_NUM
DESCRIPTION	character	30	UNIT_DESC

QA_ORGN_VIEW (Used to report QA information sorted by QA_ORGN and QA_ENGR_NAME)

View Name	data type	size	Database Name
QA_ORGN	character	06	ASND_ORGN
QA_ENGR_NAME	character	15	LAST_NAME
WPN_PGM	character	08	WPN_PGM_CODE + '-' + WPN_MOD_NUM
NOMENCLATURE	character	10	COM_NOM_TYPE + COM_NOM_NUM
PART_NUMBER	character	07	UNIT_STEM
PA	character	02	DOE_CODE
DESCRIPTION	character	30	UNIT_DESC

WPN_PGM_VIEW (Used to report QA information sorted by WPN_PGM, NOMENCLATURE, and PART_NUMBER)

View Name	data type	size	Database Name
WPN_PGM	character	08	WPN_PGM_CODE + '-' + WPN_MOD_NUM
NOMENCLATURE	character	10	COM_NOM_TYPE + COM_NOM_NUM
PART_NUMBER	character	07	UNIT_STEM
QA_ORGN	character	06	ASND_ORGN
QA_ENGINEER	character	15	LAST_NAME
PA	character	02	DOE_CODE
DESCRIPTION	character	30	UNIT_DESC

WPN_PGM_QA_V (Used to report QA information sorted by WPN_PGM, QA_ORGN, and QA_ENGINEER)

View Name	data type	size	Database Name
WPN_PGM	character	08	WPN_PGM_CODE + '-' + WPN_MOD_NUM
QA_ORGN	character	06	ASND_ORGN
QA_ENGINEER	character	15	LAST_NAME
NOMENCLATURE	character	10	COM_NOM_TYPE + COM_NOM_NUM
PART_NUMBER	character	07	UNIT_STEM
PA	character	02	DOE_CODE
DESCRIPTION	character	30	UNIT_DESC

4. Screen Definitions

Screens are identified by name in Figures 1 through 3 in Section 1 of this report. In particular, a screen is required for each frame identified in the application structure. INGRES calls a screen definition a form. Forms have been generated for the present implementation using the INGRES VIFRED (Visual Forms Editor). These forms are documented in this section. They are arranged in alphabetical order by name.

For an application to be built using INGRES ABF (Applications by forms), a "frame procedure" code must be written in DSL (Operation Specification Language) and SQL (Standard Query Language). An INGRES form plus a corresponding DSL frame procedure code completely defines a frame and an application is composed of a collection of frames. A frame will look like the form on the screen, but it will also have a menu line (generated by the DSL Code). The menu line shows the choices of user functions for that screen or frame.

Forms provide the input and output facilities of the application, and the menus of functions control the application flow. Frames establish the interface between the database and the computer terminal operator. Following are technical specifications of the forms generated using VIFRED for the PREMIS Version 1.1.2 application. The corresponding frame procedure code carries the same name as the form and is documented in a Section 5 of this document.

The highlighting characteristics of INGRES forms did not allow the menu numeric choices and first-character choices to be "brighter" than the other text. Our "work-around" was to define those choices and their default values as data with the "brightness change" attribute, instead of form "trim." These values, therefore, do not display in the form printed, but are listed in the technical description below the form:

##

DATA ENGINEERING REPORTS		ALLID
LIST	DOCUMENTS BY NOMENCLATURE/DATA ENGINEER	
Enter last name of data engineer (Blank = Report on all D.E.'s):		
KEYPAD CHOICE	REPORT DESTINATION	MENU CHOICE
-----	-----	-----
	DISPLAY report on terminal screen	ISPLAY
	put report in VAX FILE	ILE
	send report to PRINTER named in VAX symbol MYPRINTER	RINTER
	submit BATCH job to run report, print on MYPRINTER	ATCH
	eXIT to previous menu	IT
	STOP application	TOP
<p>ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR></p>		

00

Form name: allid
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 14
Number of trim strings: 27
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: data_eng
Field title:
Display format: -c15
Special display attributes: Reverse video, Brightness change, Force upper case, No auto tab
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rep_type
Field title:
Display format: -c13
Special display attributes: Display only
Type of field: regular
Default value: ALL

Validation check:
Validation error message:

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P

Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: B
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S

Validation check:
Validation error message:

##

QUALITY ASSURANCE - CHARACTERISTIC AND DEVELOPMENT REPORT INFO		CHARDEV
ENTER NOMENCLATURE	OR	PART NUM -
DESCRIPTION		
DESIGN ENGINEER	PHONE NUM	
ORGANIZATION		
FINAL DRAFT DATA	SANDIA PUBLICATION DATA	

PERCENT COMPLETE %	DOCUMENT TYPE	
PLANNED RELEASE DATE	DOCUMENT NUM	
ACTUAL RELEASE DATE	PLANNED PUB. DATE	
	ACTUAL PUB. DATE	

00

Form name: chardev
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 21
 Number of fields: 14
 Number of trim strings: 9
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: nomenclature
 Field title: NOMENCLATURE
 Display format: -C10
 Special display attributes: Reverse video, Brightness change, Force upper case
 Type of field: regular
 Default value:
 Validation check:
 Validation error message:

Field name: part_num
 Field title: PART NUM
 Display format: -C7
 Special display attributes: Reverse video, Brightness change
 Type of field: regular
 Default value:
 Validation check:
 Validation error message:

Field name: suffix
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: percent_comp
Field title: PERCENT COMPLETE
Display format: -i3
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: doc_type
Field title: DOCUMENT TYPE
Display format: -c4
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: exp_rel_date
Field title: PLANNED RELEASE DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: doc_num
Field title: DOCUMENT NUM
Display format: -c8
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: act_rel_date
Field title: ACTUAL RELEASE DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: est pub date
Field title: PLANNED PUB. DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: act pub date
Field title: ACTUAL PUB. DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: -C28
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn engr
Field title: DESIGN ENGINEER
Display format: -C25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: phone num
Field title: PHONE NUM
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn orgn
Field title: ORGANIZATION
Display format: -C6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

DATA ENGINEERING REPORTS		COMPDOC
LIST DOCUMENTS COMPLETED IN A GIVEN TIME FRAME		
Enter document completion dates (Blank = Report on all Docs): -		
KEYPAD CHOICE	REPORT DESTINATION	MENU CHOICE
-----	-----	-----
	DISPLAY report on terminal	ISPLAY
	put report in VAX FILE	ILE
	send report to PRINTER named in VAX symbol MYPRINTER	RINTER
	eXIT to previous menu	IT
	STOP application	TOP
ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR>		

00

Form name: compdoc
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 12
Number of trim strings: 24
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: data_eng
Field title:
Display format: -c8
Special display attributes: Reverse video, Brightness change, Force upper case, No auto tab
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: end_date
Field title: -
Display format: -c8
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P

Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

SANDIA RECORDS OF MANUFACTURING COMPANY CODES & ADDRESSES		COMPFORM3						
MFG CODE:	COMPANY NAME:							
STRT ADDR1:								
STRT ADDR2:								
CITY NAME:	STATE CODE:	ZIP CODE						
COUNTRY NAME:	ENTRY DATE:							
PHONE NUM: () -	SOURCE:							
	NOTES:							
RQST NAME:	ORGN:	ALT MFG CODE:						
<table border="1"> <thead> <tr> <th>OLD COMPANY NAME</th> <th>OLD CODE</th> <th>NEW CODE</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> </tr> </tbody> </table>			OLD COMPANY NAME	OLD CODE	NEW CODE			
OLD COMPANY NAME	OLD CODE	NEW CODE						

00

Form name: compform3
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 20
Number of trim strings: 4
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: mfg_code
Field title: MFG CODE:
Display format: -c3
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: company_name
Field title: COMPANY NAME:
Display format: c34
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: company_nam2
Field title:
Display format: - c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: strt_addr1
Field title: STRT ADDR1:
Display format: c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: strt_addr2
Field title: STRT ADDR2:
Display format: c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: city_name
Field title: CITY NAME:
Display format: c20
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: state code
Field title: STATE CODE:
Display format: c2
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: zip_main
Field title: ZIP CODE
Display format: -c5
Special display attributes: Reverse video
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: zip_suffix
Field title: -
Display format: -c4
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: country_name
Field title: COUNTRY NAME:
Display format: c20
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: area_code
Field title: (
Display format: -c3
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: phone_prefix
Field title:)
Display format: -c3
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: phone_suffix
Field title: -
Display format: -c4
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: source
Field title: SOURCE:
Display format: c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: notes
Field title: NOTES:
Display format: -c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rqst_name
Field title: RQST NAME:
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rqst_orgn
Field title: ORGN:
Display format: -c6
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: history
Type of field: table field

Field name: old_name
Field title: OLD COMPANY NAME
Display format: C34
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: old_mfg_code
Field title: OLD CODE
Display format: C3
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: new_mfg_code
Field title: NEW CODE

Display format: C3
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: alt mfg code
Field title: ALT MFG CODE:
Display format: c2
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: entry date
Field title: ENTRY DATE:
Display format: c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

COMPING

COMPANY NAME	CODE	CITY	STATE

```
Form name: comping
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 1
Number of trim strings: 2
Date last modified: 19-feb-90
```

Field name: companies
Type of field: table field

```
Field name: company_name
Field title: COMPANY NAME
Display format: C34
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:
```

4-17

Display format: C3
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: city_name
Field title: CITY
Display format: C20
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: state_code
Field title: STATE
Display format: C2
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

##

MANUFACTURING COMPANIES INFORMATION SYSTEM		COMPMAIN
KEYPAD CHOICE		MENU CHOICE
DATA (update or retrieve) one company		ATA
RETRIEVE many companies		ETRIEVE
GENerate REPorTS		EN RPTS
HELP		ELP
eXIT to main menu		IT
STOP Application		TOP

ENTER the KEYPAD CHOICE on the numeric KEYPAD or
ENTER the MENU CHOICE and <CR>

00

Form name: compmain
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 12
Number of trim strings: 19
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D

Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: R
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: G
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: H

Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

DATA ENGINEERING MAIN MENU		DATA
KEYPAD CHOICE	ACTIVITY	MENU CHOICE
-----	-----	-----
	show INFOrMation on part with given nomenclature/part no.	NFO
	ENTER new nomenclatures or revise part information	ENTER
	add NEW data DDCuments or modify data document information	EW DOC
	generate REPORTs	EPORT
	eXIT to previous menu	IT
	STOP application	TOP
ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR>		

00

Form name: data
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 21
Number of fields: 12
Number of trim strings: 24
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: E
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: N
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: R
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_7
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

PRODUCT EVALUATION MANAGEMENT INFORMATION SYSTEM
DATA ENGINEERING
DOCUMENT DELETION

DELETEDOC

Do you wish to delete the document shown?

DOCUMENT ID	ISS	REQUIRED	CER	EST COMP	R DRAFT	DTER	MTL LIST	ML RQST
=====	=====	=====	=====	=====	=====	=====	=====	=====

00

Form name: deletedoc

Form owner: lb

Number of columns on screen: 80

Number of lines on screen: 17

Number of fields: 1

Number of trim strings: 5

Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: doc_table

Type of field: table field

Field name: dcmt_num

Field title: DOCUMENT ID

Display format: c11

Special display attributes: Display only, Reverse video, Brightness change, Force upper cas

Type of field: column in table field

Default value:

Validation check:

Validation error message:

Field name: issue_num

Field title: ISS

Display format: c3

Special display attributes: Display only, Reverse video, Brightness change

Type of field: column in table field

Default value:

Validation check:

Validation error message:

Field name: rqr_d dt
Field title: REQUIRED
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: cer_date
Field title: CER
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: e_comp dt
Field title: EST COMP
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: ruff_dft dt
Field title: R DRAFT
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: dter_date
Field title: DTER
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: matl_list
Field title: MTL LIST
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: req_ml_date

Field title: ML RQST

Display format: c8

Special display attributes: Display only, Brightness change

Type of field: column in table field

Default value:

Validation check:

Validation error message:

ENTER NOMENCLATURE

OR PART NUMBER

•

DESCRIPTION

WPN SYS	TYPE	E R NUM	REV	REV DATE	STAT	SUPPLIER

00

Form name: dochist

Form owner: lb

Number of columns on screen: 80

Number of lines on screen: 22

Number of fields: 5

Number of trim strings: 4

Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: nomenclature

Field title: NOMENCLATURE

Display format: -C12

Special display attributes: Reverse video, Brightness change, Force upper case

Type of field: regular

Default value:

Validation check:

Validation error message:

Field name: part num

Field title: PART NUMBER

Display format: -C7

Special display attributes: Reverse video, Brightness change

Type of field: regular

Default value:

Validation check:

Validation error message:

Field name: suffix
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: doc_hist
Type of field: table field

Field name: wpn_pgm
Field title: WPN SYS
Display format: c7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: type
Field title: TYPE
Display format: c4
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: engr_rels
Field title: E R NUM
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: rev_num
Field title: REV
Display format: c2
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: comp_dt
Field title: REV DATE
Display format: c11

Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: status
Field title: STAT
Display format: c4
Special display attributes: Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: mfg_name
Field title: SUPPLIER
Display format: c34
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: -c32
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

QUALITY ASSURANCE - ENGINEERING RELEASE INFORMATION				ENGRELS
ENTER	ENGINEERING RELEASE NUMBER	REV NUM	DOCUMENT TYPE	
PART NUMBER	-	SUBJECT	WEAPON SYS	
RQRD	DATE	EST COMP DATE	ISSUE DATE	
Q A	ORGN	ENGINEER	PHONE	
DESIGN ORGN		ENGINEER	PHONE	
DESCRIPTION STATUS				
NEXT ASSEMBLY DESCRIPTION			REASON	

00

Form name: engrels
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 23
 Number of fields: 20
 Number of trim strings: 3
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: er_num
 Field title: ENGINEERING RELEASE NUMBER
 Display format: -C8
 Special display attributes: Reverse video, Brightness change, Force upper case
 Type of field: regular
 Default value:
 Validation check:
 Validation error message:

Field name: rev_num
 Field title: REV NUM
 Display format: -c2
 Special display attributes: Reverse video, Brightness change
 Type of field: regular
 Default value:

Validation check:
Validation error message:

Field name: next_as
Type of field: table field

Field name: nomen
Field title: NEXT ASSEMBLY
Display format: c12
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: u_desc
Field title: DESCRIPTION
Display format: c32
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: reason
Type of field: table field

Field name: problems
Field title: REASON
Display format: c30
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: doc_type
Field title: DOCUMENT TYPE
Display format: -c4
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: part_num
Field title: PART NUMBER
Display format: -c7
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: suffix
Field title: -
Display format: -c3
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nomenclature
Field title: SUBJECT
Display format: -c12
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: wpn sys
Field title: WEAPON SYS
Display format: -c8
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rqrd dt
Field title: RQRD DATE
Display format: -c11
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: est comp dt
Field title: EST COMP DATE
Display format: -c11
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: issue dt
Field title: ISSUE DATE
Display format: -c11
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_orgn
Field title: QTA ORGN
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: ENGINEER
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_phone
Field title: PHONE
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_orgn
Field title: DESIGN ORGN
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_engr
Field title: ENGINEER
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_phon
Field title: PHONE
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: -C32
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: status
Field title: STATUS
Display format: -c32
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

DATA ENGINEERING - ENTER / REVISE PART INFORMATION			ENTER
ENTER NOMENCLATURE	<div> <div>SUPPLIER</div> <div>WPN PGM</div> </div>		
OR PART NUMBER			
PROD AGENCY			
DESIGN	ENGR	DIV	PHONE
DATA	ENGR	DIV	PHONE
PRODUCT	ENGR	DIV	PHONE
QUAL ASSUR	ENGR	DIV	PHONE
RELIABILITY	ENGR	DIV	PHONE
QUAL CONTRL	ENGR	DIV	PHONE

00

Form name: enter
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 19
Number of fields: 23
Number of trim strings: 4
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: nomen
Field title: NOMENCLATURE
Display format: -c12
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: part_num
Field title: PART NUMBER
Display format: -c7
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: part_suf
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change, Force upper case, No auto tab
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: sup_wepn
Type of field: table field

Field name: supplier
Field title: SUPPLIER.
Display format: -c34
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: wepn_pgm
Field title: WPN_PGM
Display format: -c08
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: prod_agency
Field title: PROD AGENCY
Display format: -c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: designengr
Field title: DESIGN ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_div
Field title: DIV
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: data_engr
Field title: DATA ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: data_div
Field title: DIV
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_engr
Field title: PRODUCT ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_div
Field title: DIV
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: QUAL ASSUR ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_div
Field title: DIV
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: rel engr
Field title: RELIABILITY ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rel div
Field title: DIV
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc engr
Field title: QUAL CONTRL ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc div
Field title: DIV
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: data phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: prod_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rel_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

LIST ALL DOCUMENTS THAT HAVE NOT BEEN COMPLETED

KEYPAD CHOICE	REPORT DESTINATION	MENU CHOICE
-----	-----	-----
	put report on terminal SCREEN	SCREEN
	put report in VAX FILE	FILE
	send report to PRINTER named in VAX symbol MYPRINTER	PRINTER
	EXIT to previous menu	EXIT
	QUIT application	QUIT

ENTER KEYPAD CHOICE on the numeric KEYPAD or
ENTER MENU CHOICE and <CR>

00

Form name: incompdoc
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 24
Number of fields: 10
Number of trim strings: 22
Date last modified: 27-sep-88

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular

Default value: S
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular

Default value: E
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: Q
Validation check:
Validation error message:

##

PRODUCT EVALUATION MANAGEMENT INFORMATION SYSTEM										INFO
DATA ENGINEERING - INFORMATION										
ENTER NOMENCLATURE				SUPPLIER				WPN PGM		
or PART NUMBER -										
PROD AGENCY				TMS REQUIRED						
DESIGN	ENGR			DIV			PHONE			
DATA	ENGR			DIV			PHONE			
PRODUCT	ENGR			DIV			PHONE			
QUAL ASSUR	ENGR			DIV			PHONE			
RELIABILITY	ENGR			DIV			PHONE			
QUAL CONTRL	ENGR			DIV			PHONE			
DOCUMENT NO	ISS	REQUIRED	CER	EST COMP	R DRAFT	DTER	MTL LIST	ML RQST		

00

Form name: info
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 25
Number of trim strings: 5
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: nomen
Field title: NOMENCLATURE
Display format: -c12
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: part num
Field title: PART NUMBER
Display format: -c7
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: part_suf
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change, Force upper case, No auto tab
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: sup_wepn
Type of field: table field

Field name: supplier
Field title: SUPPLIER
Display format: -c34
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: wepn_pgm
Field title: WPN_PGM
Display format: -c08
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: doc_table
Type of field: table field

Field name: dcmt_num
Field title: DOCUMENT NO
Display format: c11
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: issue_num
Field title: ISS
Display format: c3
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: rqr_d dt
Field title: REQUIRED
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: cer_date
Field title: CER
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: e_comp dt
Field title: EST COMP
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: ruff_dft dt
Field title: R DRAFT
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: dter_date
Field title: DTER
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: matl_list
Field title: MTL_LIST
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: req_ml_date
Field title: ML RQST
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: prod_agency
Field title: PROD AGENCY
Display format: -c34
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: tmsrequired
Field title: TMS REQUIRED
Display format: -C3
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: designengr
Field title: DESIGN ENGR
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_div
Field title: DIV
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: data_engr
Field title: DATA ENGR
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: data_div
Field title: DIV
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: data_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_engr
Field title: PRODUCT ENGR
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_div
Field title: DIV
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: QUAL ASSUR ENGR
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_div
Field title: DIV
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rel_engr
Field title: RELIABILITY ENGR
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rel_div
Field title: DIV
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rel_phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc engr
Field title: QUAL CONTRL ENGR
Display format: -c25
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc div
Field title: DIV
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc phon
Field title: PHONE
Display format: -c15
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

[illegible]

```
Form name: newdoc
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 21
Number of fields: 1
Number of trim strings: 6
Date last modified: 19-feb-90
```

Field name: doc_table
Type of field: table field

Field name: issue_num
Field title: ISS
Display format: c3
Special display attributes: Reverse video, Brightness change

Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: rqr_d dt
Field title: REQUIRED
Display format: c8
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: cer_date
Field title: CER
Display format: c8
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: e_comp dt
Field title: EST COMP
Display format: c8
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: ruff_dft dt
Field title: R DRAFT
Display format: c8
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: dter_date
Field title: DTER
Display format: c8
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: matl_list
Field title: MTL_LIST
Display format: c8
Special display attributes: Reverse video

Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: req_ml_date
Field title: ML RQST
Display format: c8
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

[illegible]

```
Form name: newpart
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 20
Number of trim strings: 4
Date last modified: 19-feb-90
```

```
Field name: nomenclature
Field title: NOMENCLATURE
Display format: -c12
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:
```

4-54

Validation check:
Validation error message:

Field name: unit_suffix
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: unit_desc
Field title: DESC
Display format: -c30
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_orgn
Field title: DESIGN ORGANIZATION
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_engr
Field title: ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engn
Field title: QUAL ASSURANCE ORGN
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: nuc_sfty_org
Field title: NUCLEAR SAFETY ORGN
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_egr
Field title: ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prodagency
Field title: PROD AGENCY
Display format: -c34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: tms flag
Field title: NEEDS EVALUATION ? (Y/N)
Display format: -c1
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prodengr
Field title: PROD ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc_engr
Field title: QUAL CONT ENGR
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: next_as
Type of field: table field

Field name: cmpnt_nom
Field title: NEXT ASSEMBLY
Display format: c12
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: unit_stem
Field title: PART NO
Display format: c7
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: unit_sffx
Field title: SUFFIX
Display format: c3
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: u_desc
Field title: DESCRIPTION
Display format: c18
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: supplier
Type of field: table field

Field name: mfg
Field title: SUPPLIER
Display format: c24
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: wpn_sys
Field title: WPN
Display format: c4
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: mod_num
Field title: MOD
Display format: c3
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: desn_agency
Field title: DSN_AGENCY
Display format: -c34
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_phone
Field title: PHONE
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_phone
Field title: PHONE
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_phn
Field title: PHONE
Display format: -c14
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

DATA ENGINEERING REPORTS		NOTCOMPDC
LIST ALL DOCUMENTS THAT HAVE NOT BEEN COMPLETED		
KEYPAD CHOICE	REPORT DESTINATION	MENU CHOICE
-----	-----	-----
	DISPLAY report on terminal	ISPLAY
	put report in VAX FILE	ILE
	send report to PRINTER named in VAX symbol MYPRINTER	RINTER
	eXIT to previous menu	IT
	STOP application	TOP
ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR>		

00

Form name: notcompdc
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 22
Number of fields: 10
Number of trim strings: 23
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:

Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:

Validation error message:

Field name: key_choice_5

Field title:

Display format: -c2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 5

Validation check:

Validation error message:

Field name: men_choice_5

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: S

Validation check:

Validation error message:

PARTS

[illegible]

60

```
Form name: parts
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 4
Number of trim strings: 2
Date last modified: 19-feb-90
```

FIELD DESCRIPTIONS

```
Field name: type
Field title: TYPE
Display format: -C2
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:
```

```
Field name: wpn_pgm
Field title: WEAPON PROGRAM
Display format: -C4
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
```

Validation check:
Validation error message:

Field name: mod_num
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: cmpnts
Type of field: table field

Field name: part_num
Field title: PART NO
Display format: c7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: subject
Field title: SUBJECT
Display format: c10
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: c20
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: desn_orgn
Field title: DESIGN
Display format: c6
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: qa_orgn
Field title: QA-ORGN

Display format: c6
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: qa_engr_name
Field title: QA-ENGINEER
Display format: c10
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: prod_agency
Field title: PA
Display format: c3
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: program
Field title: SYSTEM
Display format: c7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

##

QUALITY ASSURANCE - PRODUCT QUALIFICATION TEAM INFORMATION				PQTEAM
ENTER NOMENCLATURE		OR PART NUMBER		
WEAPON PROGRAM		SUPPLIER		
DESCRIPTION		EQUIP NUMBER		
MEMBER TITLE	ENGINEER'S NAME	ORGANIZATION	PHONE NUMBER	
QUALITY ASSURANCE	ENGR			
DESIGN	ENGR			
NUCLEAR SAFETY	ENGR			
PRODUCTION AGENCY		ENGINEERS		
PRODUCT MFG	ENGR			
QUALITY CONTROL	ENGR			
MEMBER TITLE	ENGINEER NAME	ORGANIZATION	PHONE NUMBER	

00

Form name: pqteam
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 23
 Number of fields: 23
 Number of trim strings: 9
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: nomenclature
 Field title: ENTER NOMENCLATURE
 Display format: -C15
 Special display attributes: Reverse video, Brightness change, Force upper case
 Type of field: regular
 Default value:
 Validation check:
 Validation error message:

Field name: part_num
 Field title: PART NUMBER
 Display format: -C7
 Special display attributes: Reverse video, Brightness change
 Type of field: regular
 Default value:

Validation check:
Validation error message:

Field name: sffx
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: wpn_pgm
Field title: WEAPON PROGRAM
Display format: -C4
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: mod_num
Field title: -
Display format: -c2
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: supplier
Field title: SUPPLIER
Display format: -c34
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: eqp_num
Field title: EQUIP NUMBER
Display format: -c6
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: QUALITY ASSURANCE ENGR
Display format: -c24
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: qa_orgn
Field title:
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn engr
Field title: DESIGN ENGR
Display format: -c24
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_orgn
Field title:
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_egr
Field title: NUCLEAR SAFETY ENGR
Display format: -c24
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_org
Field title:
Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod agency
Field title: PRODUCTION AGENCY
Display format: -C34
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: prod engr
Field title: PRODUCT MFG ENGR
Display format: -c24
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc engr
Field title: QUALITY CONTROL ENGR
Display format: -c24
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: ad hoc
Type of field: table field

Field name: category
Field title: MEMBER TITLE
Display format: c16
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: engr name
Field title: ENGINEER NAME
Display format: c20
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: orgn
Field title: ORGANIZATION
Display format: C26
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: phone num
Field title: PHONE NUMBER

Display format: c13
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_phn
Field title:
Display format: -c14
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc_phone
Field title:

Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

QUALITY ASSURANCE - PRODUCT QUALIFICATION TEAM INFORMATION			PQTTEAM
ENTER NOMENCLATURE		OR PART NUMBER	
WEAPON PROGRAM		SUPPLIER	
DESCRIPTION			
MEMBER TITLE	ENGINEER'S NAME	PHONE NUMBER	ORGANIZATION

QUALITY ASSURANCE	ENGR		
DESIGN	ENGR		
NUCLEAR SAFETY	ENGR		
PRODUCTION AGENCY		ENGINEERS	
PRODUCT MFG	ENGR		
QUALITY CONTROL	ENGR		
+-----+-----+-----+			
MEMBER TITLE	ENGINEER NAME	PHONE NUMBER	ORGANIZATION
=====			
+-----+-----+-----+			

00

Form name: pqtteam
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 23
 Number of fields: 22
 Number of trim strings: 9
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: nomenclature
 Field title: ENTER NOMENCLATURE
 Display format: -C15
 Special display attributes: Reverse video, Brightness change, Force upper case
 Type of field: regular
 Default value:
 Validation check:
 Validation error message:

Field name: part_num
 Field title: PART NUMBER
 Display format: -C7
 Special display attributes: Reverse video, Brightness change
 Type of field: regular
 Default value:

Validation check:
Validation error message:

Field name: sffx
Field title: -
Display format: -c3
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: wpn_pgm
Field title: WEAPON PROGRAM
Display format: -C4
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: mod_num
Field title: -
Display format: -c2
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: supplier
Field title: SUPPLIER
Display format: -c34
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: ad_hoc
Type of field: table field

Field name: category
Field title: MEMBER TITLE
Display format: c16
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: engr_name
Field title: ENGINEER NAME

Display format: c20
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: phone num
Field title: PHONE NUMBER
Display format: c13
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: orgn
Field title: ORGANIZATION
Display format: C26
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: QUALITY ASSURANCE ENGR
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_orgn
Field title:

Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_engr
Field title: DESIGN ENGR
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: desn_orgn
Field title:
Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_egr
Field title: NUCLEAR SAFETY ENGR
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_phn
Field title:
Display format: -c14
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nuc_sfty_org
Field title:

Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_agency
Field title: PRODUCTION AGENCY
Display format: -C34
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_engr
Field title: PRODUCT MFG ENGR
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc_engr
Field title: QUALITY CONTROL ENGR
Display format: -c24
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qc_phone
Field title:
Display format: -c13
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

DATA ENGINEERING REPORTS		PROD
LIST PARTS BY PRODUCTION AGENCY		
Enter code for production agency (Blank = Report on all P.A.'s):		
KEYPAD CHOICE	REPORT DESTINATION	MENU CHOICE
-----	-----	-----
	DISPLAY report on terminal screen	ISPLAY
	put report in VAX FILE	ILE
	send report to PRINTER named in VAX symbol MYPRINTER	RINTER
	eXIT to previous menu	IT
	STOP application	TOP
ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR>		

00

Form name: prod
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 12
Number of trim strings: 25
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: prod_agy
Field title:
Display format: -c3
Special display attributes: Reverse video, Brightness change, Force upper case, No auto tab
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rep_type
Field title:
Display format: -c3
Special display attributes: Display only, Force upper case
Type of field: regular
Default value: ALL

Validation check:
Validation error message:

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P

Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

QUALITY ASSURANCE DATA ENTRY MENU		QAEMAIN
KEYPAD CHOICE		MENU CHOICE
	ENTER/REVISE	
	-----	-----
	NEW PART information	EW PART
	C & D RePorT status	D RPT
	TMS CENTER data	MS CENTER
	Product Qualification TEAM	Q TEAM
	HELP	ELP
	eXIT to previous menu	IT
	STOP Application	TOP
ENTER the KEYPAD CHOICE on the numeric KEYPAD or ENTER the MENU CHOICE and <CR>		

00

Form name: qaemain
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 20
Number of fields: 14
Number of trim strings: 26
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: N
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: C
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: T
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -C1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: H
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_7
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 7
Validation check:
Validation error message:

Field name: men_choice_7
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

[illegible]

```
Form name: qaengr
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 3
Number of trim strings: 3
Date last modified: 19-feb-90
```

```
Field name: engr_name
Field title: ENGINEER NAME
Display format: -C25
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:
```

4-82

Validation check:
Validation error message:

Field name: assignment
Type of field: table field

Field name: wpn_nomen
Field title: WPN PGM
Display format: c7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: part no
Field title: PART NO
Display format: C7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: sffx
Field title: SUF
Display format: c3
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: cmpnt_nomen
Field title: COMPONENT
Display format: c10
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: c12
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: pq_type
Field title: TYPE

Display format: c4
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: dcmt_num
Field title: DOC_NUM
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: rev_num
Field title: REV
Display format: c2
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: est_comp_dt
Field title: EST COMP
Display format: c11
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: status
Field title: STAT
Display format: c4
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

##

QUALITY ASSURANCE MAIN MENU		QAMAIN
KEYPAD CHOICE		MENU CHOICE
-----		-----
enter DATA		ATA
RETRIEVE information		ETRIEVE
GENerate REPorTS		EN RPTS
HELP		ELP
eXIT to main menu		IT
STOP Application		TOP
ENTER the KEYPAD CHOICE on the numeric KEYPAD or ENTER the MENU CHOICE and <CR>		

00

Form name: qamain
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 12
Number of trim strings: 22
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D

Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: R
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: G
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: H

Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

QUALITY ASSURANCE REPORTS		QAREPORT
KEYPAD		MENU
CHOICE		CHOICE
-----		-----
Create Report sorted by NOMENCLATURE		NOMENCLATURE
Create Report sorted by qa orgn and ENGINEER		ENGINEER
Create Report sorted by PART NUMBER		PART NUM
Create Report sorted by WPN PGM and nomenclature		PN PGM
Create Report sorted by Wpn Pgm and QA ORGanization		QA ORGN
Create Report sorted by production AGENCY		AGENCY
HELP		HELP
eXIT to main menu		EXIT
STOP Application		STOP
ENTER the KEYPAD CHOICE on the numeric KEYPAD or ENTER the MENU CHOICE and <CR>		

00

Form name: qareport
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 22
Number of fields: 18
Number of trim strings: 30
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -C2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: N
Validation check:

Validation error message:

Field name: key_choice_2

Field title:

Display format: -c2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 2

Validation check:

Validation error message:

Field name: men_choice_2

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: E

Validation check:

Validation error message:

Field name: key_choice_3

Field title:

Display format: -c2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 3

Validation check:

Validation error message:

Field name: men_choice_3

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: P

Validation check:

Validation error message:

Field name: key_choice_4

Field title:

Display format: -c2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 4

Validation check:

Validation error message:

Field name: men_choice_4

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: W

Validation check:

Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: WP
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: A
Validation check:
Validation error message:

Field name: key_choice_7
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 7
Validation check:
Validation error message:

Field name: men_choice_7
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: H
Validation check:

Validation error message:

Field name: key_choice_8

Field title:

Display format: -c2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 8

Validation check:

Validation error message:

Field name: men_choice_8

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: X

Validation check:

Validation error message:

Field name: key_choice_9

Field title:

Display format: -c2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 9

Validation check:

Validation error message:

Field name: men_choice_9

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: S

Validation check:

Validation error message:

##

QUALITY ASSURANCE RETRIEVAL MENU		QARMAIN
KEYPAD CHOICE	RETRIEVE INFORMATION	MENU CHOICE
-----	-----	-----
	Product Qualification TEAM	Q TEAM
	QA ENGINEER	ENGR
	DOCUMENT HISTORY by part	DC HIST
	system COMPONENTS	OMPONENTS
	ENGR RELEASE	NGR RELS
	LIST PARTS by part num or nomenclature	IST PARTS
	HELP	ELP
	EXIT to previous menu	IT
	STOP Application	TOP
ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR>		

00

Form name: qarmain
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 18
Number of trim strings: 30
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P

Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: Q
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: C

Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: E
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: L
Validation check:
Validation error message:

Field name: key_choice_7
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 7
Validation check:
Validation error message:

Field name: men_choice_7
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: H

Validation check:
Validation error message:

Field name: key_choice_8
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 8
Validation check:
Validation error message:

Field name: men_choice_8
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_9
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 9
Validation check:
Validation error message:

Field name: men_choice_9
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

Validation check:
Validation error message:

Field name: eer est c dt
Field title: EST EER ISSUE DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: issue date
Field title: ISSUE DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: nomenclature
Field title: NOMENCLATURE
Display format: -c15
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: part num
Field title: PART NUM
Display format: -c7
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: suffix
Field title: -
Display format: -c3
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: wpn pgm
Field title: WEAPON PGM
Display format: -c4
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: mod_num
Field title: -
Display format: -c2
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: supplier
Field title: SUPPLIER
Display format: -c26
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: equipment
Type of field: table field

Field name: equip_num
Field title: EQP NUM
Display format: c6
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: serial_num
Field title: SER NUM
Display format: c3
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: eqp_desc
Field title: DESC
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: doc_type
Field title: TYPE

Display format: C4
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: er_num
Field title: ER NUM
Display format: c8
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: iss_num
Field title: ISS
Display format: c2
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: qer_rqrd_dt
Field title: RQRD DATE
Display format: c11
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: est_comp_dt
Field title: EST COMP DT
Display format: c11
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: iss_date
Field title: ISSUE DATE
Display format: c11
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION

Display format: -c32
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: prod agency
Field title: PRODUCTION AGENCY
Display format: -c26
Special display attributes: Display only, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

##

DATA ENGINEERING REPORTS GENERATION		REPORT MENU CHOICE
KEYPAD CHOICE	REPORT	
	list all NEW parts by nomenclature/data engineer	EW
	list ALL parts and documents by nomenclature/data engr	LL_ID
	list all parts by WEAPON nomenclature	EAPON
	list all parts by PRODUCTION agency	ROD
	list all data documents that are INCOMPLETE	NCOMPLETE
	list all data documents COMPLETED in a given time frame	OMplete
	HELP	ELP
	eXIT to previous menu	IT
	STOP Application	TOP
ENTER KEYPAD CHOICE on the numeric KEYPAD or ENTER MENU CHOICE and <CR>		

00

Form name: report
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 23
 Number of fields: 18
 Number of trim strings: 30
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
 Field title:
 Display format: -c2
 Special display attributes: Display only, Underline
 Type of field: regular
 Default value: 1
 Validation check:
 Validation error message:

Field name: men_choice_1
 Field title:
 Display format: -c1
 Special display attributes: Display only, Underline, Brightness change
 Type of field: regular
 Default value: N

Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: A
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: W
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P

Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: I
Validation check:
Validation error message:

Field name: key_choice_6
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 6
Validation check:
Validation error message:

Field name: men_choice_6
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: C
Validation check:
Validation error message:

Field name: key_choice_7
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 7
Validation check:
Validation error message:

Field name: men_choice_7
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: H

Validation check:
Validation error message:

Field name: key_choice_8
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 8
Validation check:
Validation error message:

Field name: men_choice_8
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_9
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 9
Validation check:
Validation error message:

Field name: men_choice_9
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

QUALITY ASSURANCE REPORTS		RPT CHOICE
KEYPAD CHOICE	CHOOSE THE OUTPUT MEDIUM	MENU CHOICE
-----		-----
	DISPLAY report on the Screen	ISPLAY
	send report to a FILE	ILE
	PRINT report on a printer	RINT
	eXIT to previous menu	IT
	STOP application	TOP
<p>ENTER the KEYPAD CHOICE on the numeric KEYPAD or ENTER the MENU CHOICE and <CR></p>		

00

Form name: rptchoice
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 10
Number of trim strings: 21
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D

Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X

Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

##

QUALITY ASSURANCE - TMS CENTER INFORMATION											TMSCNTR
ENTER ENGR RELEASE NUM				REV NUM		DOCUMENT TYPE					
QA ENGINEER NAME				ORGN		PHONE NUM					
NOMENCLATUR	PART NO	SFX	WPN PGM	MOD	SUPPLIER						EQP NUM
COMM CTR CONTACT				ORGN		PHONE NUM					
TRANSMITTAL DATE				MESSAGE NUM		TWX REC'D					
FILE CAB LOC				STATUS							
OTHER RELATED DOCUMENTS				MASS-11 DIR NAME		RSLTS RPT					
WPN PGM	NOMENCLATURE	TYPE	DOC NUM	REV	QUAL ASSURANCE ENG	DESCRIPTION					

00

Form name: tmscntr
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 23
 Number of fields: 18
 Number of trim strings: 4
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: engr_rels_no
 Field title: ENGR RELEASE NUM
 Display format: -c8
 Special display attributes: Reverse video, Brightness change, Force upper case
 Type of field: regular
 Default value:
 Validation check:
 Validation error message:

Field name: rev_num
 Field title: REV NUM
 Display format: -c2
 Special display attributes: Reverse video, Brightness change
 Type of field: regular
 Default value:

Validation check:
Validation error message:

Field name: doc_type
Field title: DOCUMENT TYPE
Display format: -c4
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_engr_name
Field title: QA ENGINEER NAME
Display format: -c25
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: qa_orgn
Field title: ORGN
Display format: -c6
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: subject
Type of field: table field

Field name: nomen
Field title: NOMENCLATUR
Display format: c10
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: part_num
Field title: PART NO
Display format: c7
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: suffix
Field title: SFX

Display format: c3
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: wpn_pgm
Field title: WPN PGM
Display format: c4
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: mod_num
Field title: MOD
Display format: c2
Special display attributes: Reverse video
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: supplier
Field title: SUPPLIER
Display format: c34
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: eqp_num
Field title: EQP NUM
Display format: c7
Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: com_ctr_cont
Field title: COMM CTR CONTACT
Display format: -c25
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: com_ctr_orgh
Field title: ORGN

Display format: -c6
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: trans date
Field title: TRANSMITTAL DATE
Display format: -c11
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: message num
Field title: MESSAGE NUM
Display format: -c5
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: twx recd
Field title: TWX REC'D
Display format: -c1
Special display attributes: Reverse video
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: file cab loc
Field title: FILE CAB LOC
Display format: -c9
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: status
Field title: STATUS
Display format: -c28
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: m11_dir_name
Field title: MASS-11 DIR NAME

Display format: -c7
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rslts rpt fl
Field title: RSLTS RPT
Display format: -c1
Special display attributes: Reverse video, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: relt_docs
Type of field: table field

Field name: weapon_pgm
Field title: WPN PGM
Display format: c7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: subj
Field title: NOMENCLATURE
Display format: c12
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: type
Field title: TYPE
Display format: c4
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: er_num
Field title: DOC NUM
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: iss_num
Field title: REV
Display format: c2
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: qa_engr
Field title: QUAL ASSURANCE ENG
Display format: c18
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: c20
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: qa_phone
Field title: PHONE NUM
Display format: -c14
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: com_ctr_phon
Field title: PHONE NUM
Display format: -c14
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

PRODUCT EVALUATION MANAGEMENT INFORMATION SYSTEM		VERSION 1.1 TOPFRAME
KEYPAD CHOICE		MENU CHOICE
-----		-----
	DATA engineering activities	ATA
	Quality Assurance activities	A
	COMPANY activities	OMPANY
	PREMIS HELP	ELP
	STOP Application	TOP
ENTER the KEYPAD CHOICE using the numeric keypad or ENTER the MENU CHOICE and <CR>		

00

Form name: topframe
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 22
Number of fields: 11
Number of trim strings: 21
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: pr
Field title:
Display format: -C6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value: PREMIS
Validation check:
Validation error message:

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:

Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: Q
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: C
Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:

Validation error message:

Field name: men_choice_4

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: H

Validation check:

Validation error message:

Field name: key_choice_5

Field title:

Display format: -C2

Special display attributes: Display only, Underline

Type of field: regular

Default value: 5

Validation check:

Validation error message:

Field name: men_choice_5

Field title:

Display format: -c1

Special display attributes: Display only, Underline, Brightness change

Type of field: regular

Default value: S

Validation check:

Validation error message:

##

PRODUCT EVALUATION MANAGEMENT INFORMATION SYSTEM
 DATA ENGINEERING
 DOCUMENT ADDITION AND REVISION
 SPECIFICATION OF UNITS FOR A GIVEN DOCUMENT

UNITDOC

Enter part numbers for units having data requirements in document:

Part number	Part suffix

00

Form name: unitdoc
 Form owner: lb
 Number of columns on screen: 80
 Number of lines on screen: 22
 Number of fields: 4
 Number of trim strings: 6
 Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: unit_table
 Type of field: table field

Field name: unit_stem
 Field title: Part number
 Display format: c7
 Special display attributes: Reverse video, Force upper case
 Type of field: column in table field
 Default value:
 Validation check:
 Validation error message:

Field name: unit_da_suf
 Field title: Part suffix
 Display format: c3

Special display attributes: Reverse video, Force upper case
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: doc_type
Field title:
Display format: -c2
Special display attributes: Display only, Reverse video, Underline, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: doc_num
Field title:
Display format: -c6
Special display attributes: Display only, Reverse video, Underline, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: doc_iss_num
Field title:
Display format: -c2
Special display attributes: Display only, Reverse video, Underline, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

##

DATA ENGINEERING REPORTS

LIST PARTS BY WEAPON NOMENCLATURE

WEAPON

Enter weapon program (Blank = Report on all weapons) :

KEYPAD
CHOICE

REPORT DESTINATION

MENU
CHOICE

DISPLAY report on terminal screen

ISPLAY

put report in VAX FILE

ILE

send report to PRINTER named in VAX symbol MYPRINTER

RINTER

eXIT to previous menu

IT

STOP application

TOP

ENTER KEYPAD CHOICE on the numeric KEYPAD or
ENTER MENU CHOICE and <CR>

00

Form name: weapon
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 12
Number of trim strings: 25
Date last modified: 19-feb-90

FIELD DESCRIPTIONS

Field name: data_eng
Field title:
Display format: -c15
Special display attributes: Reverse video, Brightness change, Force upper case, No auto tal
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: rep_type
Field title:
Display format: -c3
Special display attributes: Display only, Force upper case
Type of field: regular
Default value: ALL

Validation check:
Validation error message:

Field name: key_choice_1
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 1
Validation check:
Validation error message:

Field name: men_choice_1
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: D
Validation check:
Validation error message:

Field name: key_choice_2
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 2
Validation check:
Validation error message:

Field name: men_choice_2
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: F
Validation check:
Validation error message:

Field name: key_choice_3
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 3
Validation check:
Validation error message:

Field name: men_choice_3
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: P

Validation check:
Validation error message:

Field name: key_choice_4
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 4
Validation check:
Validation error message:

Field name: men_choice_4
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: X
Validation check:
Validation error message:

Field name: key_choice_5
Field title:
Display format: -c2
Special display attributes: Display only, Underline
Type of field: regular
Default value: 5
Validation check:
Validation error message:

Field name: men_choice_5
Field title:
Display format: -c1
Special display attributes: Display only, Underline, Brightness change
Type of field: regular
Default value: S
Validation check:
Validation error message:

[illegible]

00

```
Form name: wpnsys
Form owner: lb
Number of columns on screen: 80
Number of lines on screen: 23
Number of fields: 7
Number of trim strings: 4
Date last modified: 19-feb-90
```

FIELD DESCRIPTIONS

```
Field name: part nom
Field title: PART NOMEN
Display format: -c12
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:
```

Field name: wpn_pgm
Field title: WEAPON NOMENCLATURE
Display format: -c4
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:

Validation check:
Validation error message:

Field name: mod
Field title: -
Display format: -c2
Special display attributes: Reverse video, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: cfg
Field title: TEST CFG
Display format: -c7
Special display attributes: Reverse video, Brightness change, Force upper case
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: subas
Type of field: table field

Field name: nomen
Field title: COMPONENT
Display format: c10
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: description
Field title: DESCRIPTION
Display format: c20
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: part_no
Field title: DRAW NO
Display format: c7
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: sffx
Field title: SUF

Display format: c3
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: type
Field title: TYPE
Display format: c4
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: doc_num
Field title: DOC NUM
Display format: c8
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: rev_num
Field title: REV
Display format: c2
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: rels dt
Field title: RELEASE DAT
Display format: c11
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: status
Field title: STAT
Display format: c4
Special display attributes: Display only, Brightness change
Type of field: column in table field
Default value:
Validation check:
Validation error message:

Field name: part_num
Field title: PART NUMBER

Display format: -c6
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

Field name: suffix
Field title: -
Display format: -c3
Special display attributes: Display only, Brightness change
Type of field: regular
Default value:
Validation check:
Validation error message:

5. Frames and Procedures

INGRES ABF frames are used to add menu selections to forms, and to invoke procedures which control execution of the application. Functions are specified by the menu selections. The application accomplishes the functions by making calls on procedures written in ABF Operations Specification Language (OSL), or in other programming languages such as C.

Following are the procedures written to implement the PREMIS Version 1.1.2 application. SQL statements in the source define precisely the operations which are being performed on the database.

The procedures are arranged in alphabetical order in the following section.

```

/*
                                frame: ALLID
DRIVER FOR REPORT TO LIST ALL OR NEW PARTS BY NOMENCLATURE/DATA ENGINEER
                                written by john orman 6/27/88
*/
initialize (built_tab = vchar(3),
            old_data_eng = vchar(15),
            param_eng = vchar(40),
            param_type = vchar(3)) = {
/*
                                empty all_id_rpt table
*/
                                delete from all_id_rpt ;
/*
set flag telling whether all_id_rpt table has been built */
    built_tab := 'no' ;
    if rep_type = 'NEW PARTS W/O' then
        param_type := 'new'
    else
        param_type := 'all'
    endif ;
/*
                                initialize data_eng from previous query */
    old_data_eng = 'XXXXXXXXXXXXXXXXX' ;
'DISPLAY' = {
/*
                                if there has been a change of data engineer, reset
                                built_tab
*/
                                if data_eng != :old_data_eng then
                                    if built_tab = 'yes' then
/*
                                        empty all_id_rpt table
*/
                                        delete from all_id_rpt ;
                                        endif ;
                                        old_data_eng := :data_eng ;
                                        built_tab := 'no' ;
                                    endif ;
/*
                                        if not already built, build table all_id_rpt
                                        - needed for report
*/
                                        if built_tab = 'no' then
                                            callproc build_all_id(
                                                data_eng = :data_eng,
                                                rep_type = :rep_type) ;
                                            built_tab = 'yes' ;
                                        endif ;
/*
                                        create a string for report parameter */
                                        if :data_eng = '' then
                                            param_eng := 'data_eng = ALL, rep_type = ' +
                                                :param_type ;
                                        else
                                            param_eng := 'data_eng = ' + :data_eng +
                                                ', rep_type = ' + :param_type ;
                                        endif ;
/*
                                        send report on unit table to screen
*/
                                        call report (report = 'all_id_rpt', mode = 'wrap',
                                            param = :param_eng) ;

```

```

    }

'FILE' = {
/*      if there has been a change of data engineer, reset
        built_tab

*/
    if data_eng != :old_data_eng then
        if built_tab = 'yes' then

/*          empty all_id_rpt table

*/          delete from all_id_rpt ;
            endif ;
            old_data_eng := :data_eng ;
            built_tab := 'no' ;
        endif ;

/*      if not already built, build table all_id_rpt
        - needed for report

*/
    if built_tab = 'no' then
        callproc build_all_id(
                                data_eng = :data_eng,
                                rep_type = :rep_type) ;
        built_tab = 'yes' ;
    endif ;

/*      create a string for report parameter */
    if :data_eng = '' then
        param_eng := 'data_eng = ALL, rep_type = ' +
                    :param_type ;
    else
        param_eng := 'data_eng = ' + :data_eng +
                    ', rep_type = ' + :param_type ;
    endif ;

/*      send report on unit table to file parts.rpt

*/
    call report (report = 'all_id_rpt', mode = 'wrap',
                file = 'parts_' + :param_type + '.rpt',
                param = :param_eng) ;
    message 'File containing report is parts_' +
            :param_type + '.rpt' ;
    sleep 3 ;
}

'PRINTER' =
{
/*      if there has been a change of data engineer, reset
        built_tab

*/
    if data_eng != :old_data_eng then
        if built_tab = 'yes' then

/*          empty all_id_rpt table

*/          delete from all_id_rpt ;
            endif ;
            old_data_eng := :data_eng ;
            built_tab := 'no' ;
        endif ;

/*      if not already built, build table all_id_rpt

```

```

- needed for report
*/
    if built_tab = 'no' then
        callproc build_all_id(
            data_eng = :data_eng,
            rep_type = :rep_type) ;
        built_tab = 'yes' ;
    endif ;
/*
    create a string for report parameter
    if :data_eng = '' then
        param_eng := 'data_eng = ALL, rep_type = ' +
                    :param_type ;
    else
        param_eng := 'data_eng = ' + :data_eng +
                    ', rep_type = ' + :param_type ;
    endif ;

    put report in file to be sent to printer
*/
call report (report = 'all_id_rpt', mode = 'wrap',
            file = 'parts.rpt',
            param = :param_eng) ;
/*
    send file containing report to appropriate printer
    use special procedure print_dispose
    first parameter to procedure print_dispose if name
    of file to be printed jlo 9/21/88
*/
call system 'QDB1:[DBAPRDB.SOURCE]print_dispose parts.rpt' ;
message 'PARTS.RPT is being printed on MYPRINTER' ;
sleep 3 ;
}
'BATCH' =
{
/*
    if there has been a change of data engineer, reset
    built_tab
*/
    if data_eng != :old_data_eng then
        if built_tab = 'yes' then
/*
            empty all_id_rpt table
*/
            delete from all_id_rpt ;
            endif ;
            old_data_eng := :data_eng ;
            built_tab := 'no' ;
        endif ;
/*
        submit batch procedure to build report table,
        call report, and print report
*/
call system 'submit/noprint/parameters=( "' + :built_tab +
            '" , "' + :param_type + '" ) ' +
            'DB1:[DBAPRDB.SOURCE]BATCH_RPT' ;
/*
    table all_id_rpt will be built by batch job if not already built
*/
    built_tab = 'yes' ;
    message 'Batch job to run report has been submitted' ;
    sleep 3 ;
}

```

```

        message 'ALL_ID.RPT will be printed on MYPRINTER' ;
        sleep 3 ;
    }
    /*
    'HELP', key frskey1 =
    {
        help_forms (subject = 'Reports, Nomenclature and Data Engineer',
            file = 'db1:[dbaprd.b.help]allid.help');
    }
    /*
    'XIT', key frskey3 = {
        RETURN ;
    }
    'STOP', key frskey2 = {
        EXIT ;
    }

```

```

PROCEDURE BUILD_ALL_ID(data_eng = vchar(15),
                      rep_type = vchar(3),
                      sq_data_eng = vchar(15),
                      row_count = integer) =
{
/*
        PROCEDURE: BUILD ALL ID
        CREATE ALL ID REPORT TABLE
        FOR A GIVEN DATA ENGINEER
        modifications made to select only new parts    jlo 9/16/88
*/
/*      eliminate any leading or trailing blanks */
build_all_id := select sq_data_eng = squeeze(:data_eng)
                  from weapon ;
/*
        FIRST INSERT ALL WEAPONS INTO TABLE -
        HAVE LONGER NOMENCLATURE
*/
if sq_data_eng != '' then
/*      user has selected a data engineer      */
  if rep_type = 'NEW' then
/*      select only parts with NO data documents */
    INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                                PART_NUMBER,
                                PROD_AGENCY,
                                DESN_AGENCY,
                                DATA_ENGNR )
SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PRQ_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ' ' +
    PERSON.MIDDL_INIT +
    ' ' +
    PERSON.LAST_NAME
FROM
    UNIT,
    WEAPON,
    DA_EMPL,
    PERSON
WHERE
    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EMPL.EMPL_NUM
    AND
    DA_EMPL.ID_NUM = PERSON.ID_NUM
    AND
    PERSON.LAST NAME = :sq_data_eng
    AND NOT EXISTS
      (SELECT *
       FROM UNIT_DOC,

```



```

        DATA_DOCUMNT
    WHERE
        UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
        AND
        UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
        AND
        UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
        AND
        UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
        AND
        UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

/*
    NOW INSERT ALL COMPONENTS INTO TABLE

INSERT INTO ALL_ID_RPT (NOMENCLATURE,
                        PART_NUMBER,
                        PROD_AGENCY,
                        DESN_AGENCY,
                        DATA_ENGR )

SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ' ' +
    PERSON.MIDDL_INIT +
    ' ' +
    PERSON.LAST_NAME
FROM
    UNIT,
    COMPONENT,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM
    AND
    PERSON.LAST_NAME = :sq_data_eng
    AND NOT EXISTS
        (SELECT *
         FROM UNIT_DOC,
              DATA_DOCUMNT
         WHERE
             UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
             AND
             UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
             AND
             UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
             AND
             UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
             AND
             UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

```

else
/*      SELECT ALL PARTS      */
      INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                                  PART_NUMBER,
                                  PROD_AGENCY,
                                  DESN_AGENCY,
                                  DATA_ENGNR )

      SELECT
        WEAPON.WPN_PGM_CODE +
        ',' +
        WEAPON.WPN_TEST_CFG +
        '-' +
        WEAPON.WPN_MOD_NUM,
        UNIT.UNIT_STEM +
        UNIT.UNIT_DA_SUF,
        UNIT.PRD_MFG_CODE,
        UNIT.DSN_MFG_CODE,
        PERSON.FIRST_NAME +
        ',' +
        PERSON.MIDDL_INIT +
        ',' +
        PERSON.LAST_NAME
      FROM
        UNIT,
        WEAPON,
        DA_EML,
        PERSON
      WHERE
        UNIT.UNIT_STEM = WEAPON.UNIT_STEM
        AND
        UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
        AND
        UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
        AND
        DA_EML.ID_NUM = PERSON.ID_NUM
        AND
        PERSON.LAST_NAME = :sq_data_eng ;

/*
      NOW INSERT ALL COMPONENTS INTO TABLE
*/
      INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                                  PART_NUMBER,
                                  PROD_AGENCY,
                                  DESN_AGENCY,
                                  DATA_ENGNR )

      SELECT
        COMPONENT.COM_NOM_TYPE +
        COMPONENT.COM_NOM_NUM,
        UNIT.UNIT_STEM +
        UNIT.UNIT_DA_SUF,
        UNIT.PRD_MFG_CODE,
        UNIT.DSN_MFG_CODE,
        PERSON.FIRST_NAME +
        ',' +
        PERSON.MIDDL_INIT +
        ',' +
        PERSON.LAST_NAME
      FROM
        UNIT,
        COMPONENT,

```

```

        DA_EEMPL,
        PERSON
WHERE
        UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
        AND
        UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
        AND
        UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
        AND
        DA_EEMPL.ID_NUM = PERSON.ID_NUM
        AND
        PERSON.LAST_NAME = :sq_data_eng ;

endif ;
else
/*          no data engineer entered          */
if rep_type = 'NEW' then
/*          only put parts with no data documents in report */
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGR )

SELECT
        WEAPON.WPN_PGM_CODE +
        '/' +
        WEAPON.WPN_TEST_CFG +
        '-' +
        WEAPON.WPN_MOD_NUM,
        UNIT.UNIT_STEM +
        UNIT.UNIT_DA_SUF,
        UNIT.PRD_MFG_CODE,
        UNIT.DSN_MFG_CODE,
        PERSON.FIRST_NAME +
        ' ' +
        PERSON.MIDDL_INIT +
        ' ' +
        PERSON.LAST_NAME

FROM
        UNIT,
        WEAPON,
        DA_EEMPL,
        PERSON

WHERE
        UNIT.UNIT_STEM = WEAPON.UNIT_STEM
        AND
        UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
        AND
        UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
        AND
        DA_EEMPL.ID_NUM = PERSON.ID_NUM
        AND NOT EXISTS
            (SELECT *
             FROM UNIT_DOC,
                  DATA_DOCUMNT
             WHERE
                 UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
                 AND
                 UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
                 AND
                 UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM

```

```

AND
UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
AND
UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

```

/*
SOME WEAPONS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
CHANGE MADE BY JOHN ORMAN 9/14/89
*/

```

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )

SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    WEAPON.UNIT_STEM +
    WEAPON.UNIT_DA_SUF

FROM
    WEAPON
WHERE
    NOT EXISTS
        (SELECT * FROM UNIT WHERE
            UNIT.UNIT_STEM = WEAPON.UNIT_STEM
            AND
            UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF)
    AND NOT EXISTS
        (SELECT *
            FROM UNIT_DOC,
            DATA_DOCUMNT
            WHERE
                UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
                AND
                UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
                AND
                UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
                AND
                UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
                AND
                UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

```

/*
NOW INSERT ALL COMPONENTS INTO TABLE
*/

```

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGNR )

SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF,
    UNIT.PR_D_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ' ' +
    PERSON.MIDDLE_INIT +
    ' ' +
    PERSON.LAST_NAME

FROM

```

UNIT,
COMPONENT,
DA_EML,
PERSON

WHERE

UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
AND
UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
AND
UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
AND
DA_EML.ID_NUM = PERSON.ID_NUM
AND NOT EXISTS
(SELECT *
FROM UNIT_DOC,
DATA_DOCUMNT
WHERE
UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
AND
UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
AND
UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
AND
UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
AND
UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM) ;

/*

SOME COMPONENTS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
CHANGE MADE BY JOHN ORMAN 9/14/89

*/

INSERT INTO ALL_ID_RPT (NOMENCLATURE,
PART_NUMBER)

SELECT

COMPONENT.COM_NOM_TYPE +
COMPONENT.COM_NOM_NUM,
COMPONENT.UNIT_STEM +
COMPONENT.UNIT_DA_SUF

FROM

COMPONENT

WHERE NOT EXISTS

(SELECT * FROM UNIT WHERE
UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
AND
UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF)

AND NOT EXISTS

(SELECT *
FROM UNIT_DOC,
DATA_DOCUMNT
WHERE
UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
AND
UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
AND
UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
AND
UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
AND
UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM) ;

else

/*

put all parts in report */

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGR )

SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    WEAPON.UNIT_STEM +
    WEAPON.UNIT_DA_SUF,
    UNIT.PROD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ',' +
    PERSON.MIDDLE_INIT +
    ',' +
    PERSON.LAST_NAME
FROM
    UNIT,
    WEAPON,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM ;

/*
    SOME WEAPONS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
    CHANGE MADE BY JOHN ORMAN  9/14/89
*/

```

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )

SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    WEAPON.UNIT_STEM +
    WEAPON.UNIT_DA_SUF
FROM
    WEAPON
WHERE
    NOT EXISTS
        (SELECT * FROM UNIT WHERE
            UNIT.UNIT_STEM = WEAPON.UNIT_STEM
            AND
            UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF) ;

```

/*
 NOW INSERT ALL COMPONENTS INTO TABLE
 */

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,

```

```

                                DESN_AGENCY,
                                DATA_ENGNR )
SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    , , +
    PERSON.MIDDL_INIT +
    , , +
    PERSON.LAST_NAME
FROM
    UNIT,
    COMPONENT,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM ;

/*
    SOME COMPONENTS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
    CHANGE MADE BY JOHN ORMAN    9/14/89
*/
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )
SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF
FROM
    COMPONENT
WHERE NOT EXISTS
    (SELECT * FROM UNIT WHERE
        UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
        AND
        UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF) ;

    endif ;
endif ;
/*
    see if have any entries in table */
build_all_id := select row_count = count(*)
                from all_id_rpt ;
if row_count = 0 then
/*
    put an informative entry into table */
    insert into all_id_rpt(
                                NOMENCLATURE,
                                DATA_ENGNR )
    values ('NO DATA' , :sq_data_eng) ;

endif ;
}

```

```

PROCEDURE build_prod(prod_agy = vchar(15),
                    rep_type = vchar(3),
                    sq_prod_agy = vchar(15),
                    row_count = integer) =

```

```

{
/*

```

```

        PROCEDURE: build_prod
        CREATE ALL_ID REPORT TABLE
        FOR A GIVEN production agency
        modifications made to select only new parts jlc 9/16/88

```

```

*/
/*      eliminate any leading or trailing blanks */
build_prod := select sq_prod_agy = squeeze(:prod_agy)
                from weapon ;

```

```

/*
        FIRST INSERT ALL WEAPONS INTO TABLE -
        HAVE LONGER NOMENCLATURE

```

```

*/
if sq_prod_agy != '' then
/*      user has selected a production agency */
  if rep_type = 'NEW' then
/*      select only parts with NO data documents */
    INSERT INTO ALL_ID_RPT (NOMENCLATURE,
                          PART_NUMBER,
                          PROD_AGENCY,
                          DESN_AGENCY,
                          DATA_ENGNR )

```

```

SELECT

```

```

    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ',' +
    PERSON.MIDDL_INIT +
    ',' +
    PERSON.LAST_NAME

```

```

FROM

```

```

    UNIT,
    WEAPON,
    DA_EEMPL,
    PERSON

```

```

WHERE

```

```

    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
    AND
    DA_EEMPL.ID_NUM = PERSON.ID_NUM
    AND
    UNIT.PRD_MFG_CODE = :sq_prod_agy
    AND NOT EXISTS
        (SELECT *
         FROM UNIT_DOC,

```



```

DATA_DOCUMNT
WHERE
UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
AND
UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
AND
UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
AND
UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
AND
UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

/*

NOW INSERT ALL COMPONENTS INTO TABLE

*/

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGNR )

```

SELECT

```

COMPONENT.COM_NOM_TYPE +
COMPONENT.COM_NOM_NUM,
UNIT.UNIT_STEM +
UNIT.UNIT_DA_SUF,
UNIT.PRD_MFG_CODE,
UNIT.DSN_MFG_CODE,
PERSON.FIRST_NAME +
', ' +
PERSON.MIDDL_INIT +
', ' +
PERSON.LAST_NAME

```

FROM

```

UNIT,
COMPONENT,
DA_EML,
PERSON

```

WHERE

```

UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
AND
UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
AND
UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
AND
DA_EML.ID_NUM = PERSON.ID_NUM
AND
UNIT.PRD_MFG_CODE = :sq_prod_agy
AND NOT EXISTS
  (SELECT *
   FROM UNIT_DOC,
        DATA_DOCUMNT
   WHERE
      UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
      AND
      UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
      AND
      UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
      AND
      UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
      AND
      UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

```

else
/*      SELECT ALL PARTS      */
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGNR )

SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ',' +
    PERSON.MIDDL_INIT +
    ',' +
    PERSON.LAST_NAME

FROM
    UNIT,
    WEAPON,
    DA_EEMPL,
    PERSON

WHERE
    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
    AND
    DA_EEMPL.ID_NUM = PERSON.ID_NUM
    AND
    UNIT.PRD_MFG_CODE = :sq_prod_agy ;

/*
      NOW INSERT ALL COMPONENTS INTO TABLE
*/
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGNR )

SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ',' +
    PERSON.MIDDL_INIT +
    ',' +
    PERSON.LAST_NAME

FROM
    UNIT,
    COMPONENT,

```

```

        DA_EEMPL,
        PERSON
    WHERE
        UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
        AND
        UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
        AND
        UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
        AND
        DA_EEMPL.ID_NUM = PERSON.ID_NUM
        AND
        UNIT.PR_D_MFG_CODE = :sq_prod_agy ;

    endit ;
else
/*      no production agency entered      */
/*      if rep_type = 'NEW' then          */
/*      only put parts with no data documents in report */
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGR )

SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PR_D_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ',' +
    PERSON.MIDDL_INIT +
    ',' +
    PERSON.LAST_NAME

FROM
    UNIT,
    WEAPON,
    DA_EEMPL,
    PERSON

WHERE
    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
    AND
    DA_EEMPL.ID_NUM = PERSON.ID_NUM
    AND NOT EXISTS
        (SELECT *
         FROM UNIT_DOC,
              DATA_DOCUMNT
         WHERE
             UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
             AND
             UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
             AND
             UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM

```

```

AND
UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
AND
UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

```

/*
*/

```

NOW INSERT ALL COMPONENTS INTO TABLE

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGR )

SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PR_D_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ' ' +
    PERSON.MIDDL_INIT +
    ' ' +
    PERSON.LAST_NAME
FROM
    UNIT,
    COMPONENT,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM
    AND NOT EXISTS
        (SELECT *
         FROM UNIT_DOC,
              DATA_DOCUMNT
         WHERE
             UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
             AND
             UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
             AND
             UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
             AND
             UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
             AND
             UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

else

```

/*      put all parts in report */
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGR )

```

```

SELECT
    WEAPON.WPN_PGM_CODE +

```

```

        ' ' +
        WEAPON.WPN_TEST_CFG +
        ' ' +
        WEAPON.WPN_MOD_NUM,
        UNIT.UNIT_STEM +
        UNIT.UNIT_DA_SUF,
        UNIT.PRD_MFG_CODE,
        UNIT.DSN_MFG_CODE,
        PERSON.FIRST_NAME +
        ' ' +
        PERSON.MIDDL_INIT +
        ' ' +
        PERSON.LAST_NAME
FROM
    UNIT,
    WEAPON,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM ;

/*
    NOW INSERT ALL COMPONENTS INTO TABLE
*/
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGR )

SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    UNIT.UNIT_STEM +
    UNIT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ' ' +
    PERSON.MIDDL_INIT +
    ' ' +
    PERSON.LAST_NAME
FROM
    UNIT,
    COMPONENT,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM ;

endif ;

```

```

endif ;
/*          see if have any entries in table */
build_prod := select row_count = count(*)
               from all_id_rpt ;
if row_count = 0 then
/*          put an informative entry into table */
insert into all_id_rpt(
               NOMENCLATURE,
               DATA_ENGNR )
values ('NO DATA', :sq_prod_agy) ;
endif ;
}

```

/*

Procedure for CHARDEV frame
Written by Norman Stevens

CHARDEV

*/

Initialize (rcount = integer, w_pcode = vchar(4), w_cfg = vchar(7),
w_modno = vchar(2), h_type = vchar(4), h_doc_num = vchar(8),
h_desn = vchar(3), answer = vchar(1)) =

{
resume field nomenclature;
}

'RETRIEVE' =

{

/*

DSL to retrieve part number and suffix from PREMIS

*/

```
if nomenclature = '' then
  if part_num = '' then
    answer := prompt 'You must enter a part number or a nomenclature' +
      '. Hit <CR> to continue.';
    resume field nomenclature;
  else
    chardev := select nomenclature = c.com_nom_type + c.com_nom_num
      from component c where c.unit_stem = :part_num and
        c.unit_da_suf = :suffix;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      chardev := select w_pcode = w.wpn_pgm_code,
        w_cfg = w.wpn_test_cfg, w_modno = w.wpn_mod_num from weapon w
        where w.unit_stem = :part_num and w.unit_da_suf = :suffix;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        answer := prompt 'No component exist with that part number.' +
          ' Hit <CR> to continue.';
      else
        if w_cfg = '' then
          nomenclature := w_pcode + '-' + w_modno;
        else
          nomenclature := w_pcode + '/' + w_cfg + '-' + w_modno;
        endif;
      endif;
    endif;
  endif;
else
  chardev := select part_num = c.unit_stem, suffix = c.unit_da_suf
    from component c where c.com_nom_type = left(:nomenclature, 2) and
      c.com_nom_num = shift(:nomenclature, -2);
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    answer := prompt 'No part number exists with that nomenclature.' +
      'Hit <CR> to continue';
  endif;
endif;
```

/*

DSL to retrieve the unit description from PREMIS

*/

```
chardev := select description = u.unit_desc from unit_desc u
  where u.unit_stem = :part_num and u.unit_da_suf = :suffix;
```

/*

DSL to retrieve the design engineer data from PREMIS

```

*/
chardev := select desn_engr = p.first_name + ' ' + p.middl_init + ' ' +
  p.last_name, desn_orgn = p.asnd_orgn,
  phone_num = left(shift(p.phone_num, -5), 1) + '-' +
    shift(p.phone_num, -6)
  from person p, unit u where (p.id_num = u.da_desn_engr) and
    (:part_num = u.unit_stem and :suffix = u.unit_da_suf);
if phone_num = '-' then
  phone_num := '';
endif;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
  desn_engr := 'UNKNOWN';
  desn_orgn := '';
  chardev := select desn_orgn = d.division from desn_orgn d, unit u
    where :part_num = u.unit_stem and :suffix = u.unit_da_suf
      and u.da_desn_engr = d.division;
endif;

/*
  DSL to retrieve the characteristic and development report data
  from PREMIS

*/
chardev := select doc_type = u.doc_type, doc_num = u.doc_num,
  percent_comp = c.percent_comp, exp_rel_date = c.est_draft_dt,
  act_rel_date = c.finl_drft_dt, est_pub_date = c.est_pub_dt,
  act_pub_date = c.published_dt from unit u, char_dev_rpt c
  where (u.unit_stem = :part_num and u.unit_da_suf = :suffix)
    and (c.doc_type = u.doc_type and c.doc_num = u.doc_num and
      c.doc_iss_num = u.doc_iss_num);
inquire_ingres(rcount = rowcount);
if rcount = 0 then
  answer := prompt 'C & D Report data for that unit is not in PREMIS.' +
    ' Hit <CR> to continue';
endif;
}

'ADD/REVISE' =
{
/*
  DSL to store the characteristic and development data in PREMIS

*/
if doc_type != '' and doc_num != '' then
  if :doc_type != 'SAND' and :doc_type != 'MEMO' then
    answer := prompt 'DOC TYPE must be "SAND" or "MEMO". ' +
      'Hit <CR> to continue';
  else
    if percent_comp < 0 then
      answer := prompt 'Percent complete cannot be less than 0. ' +
        'Hit <CR> to continue';
    elseif percent_comp > 100 then
      answer := prompt 'Percent complete cannot be greater than 100. ' +
        'Hit <CR> to continue';
    else
      chardev := select h_type = c.doc_type, h_doc_num = c.doc_num
        from char_dev_rpt c
        where c.doc_type = :doc_type and c.doc_num = :doc_num;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        insert into char_dev_rpt (doc_type, doc_num, percent_comp,
          est_draft_dt, finl_drft_dt, est_pub_dt, published_dt)

```



```

        values (:doc_type, :doc_num, :percent_comp, :exp_rel_date,
               :act_rel_date, :est_pub_date, :act_pub_date);
    else
        update char_dev_rpt c set percent_comp = :percent_comp,
            est_draft_dt = :exp_rel_date, finl_drft_dt = :act_rel_date,
            est_pub_dt = :est_pub_date, published_dt = :act_pub_date
            where c.doc_type = :doc_type and c.doc_num = :doc_num;
    endif;
    if :part_num = '' then
        answer := prompt 'We cannot update the unit record unless we know'+
            ' the Part Number. Hit <CR> to continue';
    else
        chardev := select h_desn = u.dsn_mfg_code from unit u
            where u.unit_stem = :part_num and u.unit_da_suf = :suffix;
        inquire_ingres(rcount=rowcount);
        if rcount = 0 then
            insert into unit (unit_stem, unit_da_suf, doc_type, doc_num)
                values (:part_num, :suffix, :doc_type, :doc_num);
        else
            update unit u set doc_type = :doc_type, doc_num = :doc_num
                where u.unit_stem = :part_num and u.unit_da_suf = :suffix;
        endif;
    endif;
endif;
endif;
endif;
}

'CLEAR' =
{
    clear field all;
    resume field nomenclature;
}

'HELP', key frskey1 =
{
    help_forms (subject = 'characteristic and development report',
        file = 'db1:[dbaprdb.help]chardev.help');
}

'XIT', key frskey3 =
{
    RETURN;
}

'STOP', key frskey2 =
{
    EXIT;
}

```

```

/*
    frame: COMPDOG
    DRIVER FOR REPORT TO LIST ALL DATA DOCUMENTS THAT WERE COMPLETED
    IN A GIVEN TIME FRAME
    ***** NOT IMPLEMENTED AS OF 9/27/88 *****
    written by john orman 9/27/88
*/
initialize (
) = {
/*
    ***** DISPLAY SCREEN ***** */
    redisplay ;
/*
    ***** SHOW THAT SCREEN NOT IMPLEMENTED ***** */
    message '***** This report has not been implemented *****' ;
    sleep 5 ;
    RETURN ;
}
'DISPLAY' = {
}
'FILE' = {
}
'PRINTER' =
{
}
'HELP', key frskey1 =
{
    help_forms (subject = 'List all data documents completed',
        file = 'db1:[dbaprdb.help]compdoc.help');
}
'XIT', key frskey3 = {
    RETURN ;
}
'STOP', key frskey2 = {
    EXIT ;
}
}

```

```
*/
Initialize (answer = vchar(1), rcount = integer1, err_count = integer1,
  rqst_id = vchar(9), comp_type = vchar(1), rqst_code = vchar(3),
  phone_num = vchar(10), zip_code = vchar(9), cur_code = vchar(3),
  cur_name = vchar(34)) =
```

```
{
}
```

```
'RETRIEVE' =
```

```
{
```

```
  if mfg_code = '' then
    if company_name = '' then
      answer := prompt 'You must enter either a company code or a company '
        + 'name to retrieve data. Hit <CR> to continue';
```

```
    else
```

```
      message 'Retrieving Information ...';
      compform3 := select mfg_code = c.mfg_code, strt_addr1 = a.strt_addr1,
        strt_addr2 = a.strt_addr2, city_name = a.city_name,
        state_code = a.state_code, zip_main = left(a.zip_code, 5),
        zip_suffix = shift(a.zip_code, -5),
        country_name = a.country_name, area_code = left(c.phone_num, 3),
        phone_prefix = left(shift(c.phone_num, -3), 3),
        phone_suffix = shift(c.phone_num, -6),
        source = c.source, entry_date = c.entry_date,
        company_nam2 = c.company_nam2, notes = c.notes,
        rqst_name = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
        rqst_orgn = p.asnd_orgn, alt_mfg_code = c.alt_mfg_code,
        rqst_code = p.mfg_code from company c, address a, person p
        where :company_name = c.company_name and c.mfg_code = a.address_id
          and c.comp_req = p.id_num;
      inquire_ingres (rcount = rowcount);
      if rcount = 0 then
        answer := prompt 'There is no company with that name in the ' +
          'database. Hit <CR> to continue';
```

```
      else
```

```
        history := select old_name = h.old_name, old_mfg_code = h.old_mfg_code,
          new_mfg_code = h.new_mfg_code from company_hist h
          where :mfg_code = h.current_code;
```

```
      endif;
```

```
    endif;
```

```
  else
```

```
    message 'Retrieving Information ...';
    compform3 := select company_name = c.company_name,
      strt_addr1 = a.strt_addr1, strt_addr2 = a.strt_addr2,
      city_name = a.city_name, state_code = a.state_code,
      zip_main = left(a.zip_code, 5), zip_suffix = shift(a.zip_code, -5),
      country_name = a.country_name, area_code = left(c.phone_num, 3),
      phone_prefix = left(shift(c.phone_num, -3), 3),
      phone_suffix = shift(c.phone_num, -6), source = c.source,
      company_nam2 = c.company_nam2, notes = c.notes,
      entry_date = c.entry_date, rqst_code = p.mfg_code,
      rqst_name = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
      rqst_orgn = p.asnd_orgn, alt_mfg_code = c.alt_mfg_code
      from company c, address a, person p
      where :mfg_code = c.mfg_code and c.mfg_code = a.address_id
        and c.comp_req = p.id_num;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
```

```

        answer := prompt 'There is no company with that code in the ' +
            'database. Hit <CR> to continue';
    else
        history := select old_name = h.old_name, old_mfg_code = h.old_mfg_code,
            new_mfg_code = h.new_mfg_code from company_hist h
            where :mfg_code = h.current_code;
    endif;
endif;
if rcount = 0 then
    cur_code := '';
    cur_name := '';
elseif rcount = 1 then
    cur_code := :mfg_code;
    cur_name := :company_name;
    if rqst_code != 'SBT' and rqst_code != 'SBJ' then
        compform3 := select rqst_orgh = p.mfg_code from company c, person p
            where :mfg_code = c.mfg_code and c.comp_req = p.id_num;
    endif;
endif;
}

'ADD/REVISE' =
{
    if cur_code = '' then
        cur_code := :mfg_code;
    endif;
    err_count := 0;
    if mfg_code = '' then
        err_count = err_count + 1;
        answer := prompt 'You cannot add a company unless you input a mfg_code '
            + '. Hit <CR> to continue';
        resume field mfg_code;
    endif;
    if company_name = '' then
        err_count = err_count + 1;
        answer := prompt 'You cannot add a company unless you input a company '
            + 'name. Hit <CR> to continue';
        resume field company_name;
    endif;
    if strt_addr1 = '' then
        err_count = err_count + 1;
        answer := prompt 'You cannot add a company unless you input a street '
            + 'address. Hit <CR> to continue';
        resume field strt_addr1;
    endif;
    if city_name = '' then
        err_count = err_count + 1;
        answer := prompt 'You cannot add a company unless you input a city '
            + 'name. Hit <CR> to continue';
        resume field city_name;
    endif;
    if state_code = '' then
        if country_name = '' then
            err_count = err_count + 1;
            answer := prompt 'You cannot add a company unless you input either '
                + 'a state code or a country name. Hit <CR> to continue';
            resume field state_code;
        endif;
    else
        compform3 := select rcount = count(*) from state

```

```

where :state_code = state.state_code;
if rcount = 0 then
  err_count = err_count + 1;
  answer := prompt 'The state code you have entered is invalid '
    + '. Hit <CR> to continue';
  resume field state_code;
endif;
endif;
if :rqst_name = '' then
  rqst_id := '';
else
  compform3 := select comp_type = left(:rqst_orgn, 1) from company;
  if comp_type = "0" then
    rcount := 0;
    compform3 := select rcount = count(*), rqst_id = p.id_num
      from person p where :rqst_name = p.first_name + ' ' + p.middl_init
        + ' ' + p.last_name and :rqst_orgn = p.asnd_orgn;
    if rcount = 0 then
      compform3 := select rqst_id = p.id_num, rcount = count(*)
        from person p where :rqst_name = left(p.first_name, 1) + ' ' +
          p.middl_init + ' ' + p.last_name and :rqst_orgn = p.asnd_orgn;
      if rcount = 0 then
        compform3 := select rcount = count(*), rqst_id = p.id_num
          from person p where :rqst_name = p.last_name
            and :rqst_orgn = p.asnd_orgn;
        if rcount = 0 then
          rqst_id := '';
          answer := prompt 'The requestor will not be stored in the ' +
            ' database. Contact your DBA. Hit <CR> to continue';
        elseif rcount = 1 then
          compform3 := select rqst_name = p.first_name + ' ' +
            p.middl_init + ' ' + p.last_name from person p
              where :rqst_id = p.id_num;
        else
          rqst_id := '';
          answer := prompt 'More than one person in the ' +
            ' database has that last name and organization. Please '
            + ' enter the initials. Hit <CR> to continue';
        endif;
      elseif rcount = 1 then
        compform3 := select rqst_name = p.first_name + ' ' +
          p.middl_init + ' ' + p.last_name from person p
            where :rqst_id = p.id_num;
      else
        rqst_id := '';
        answer := prompt 'There is more than one person in the ' +
          ' organization with the same initials. Contact your DBA';
      endif;
    endif;
  else
    rcount := 0;
    compform3 := select rcount = count(*), rqst_id = p.id_num
      from person p where :rqst_name = p.first_name + ' ' + p.middl_init
        + ' ' + p.last_name and :rqst_orgn = p.mfg_code;
    if rcount = 0 then
      compform3 := select rqst_id = p.id_num, rcount = count(*)
        from person p where :rqst_name = left(p.first_name, 1) + ' ' +
          p.middl_init + ' ' + p.last_name and :rqst_orgn = p.mfg_code;
      if rcount = 0 then
        compform3 := select rqst_id = p.id_num, rcount = count(*)

```

```

        from person p where :rqst_name = p.last_name
        and :rqst_orgn = p.mfg_code;
    if rcount = 0 then
        rqst_id := '';
        answer := prompt 'The requestor will not be stored in the ' +
            ' database. Contact your DBA. Hit <CR> to continue';
    elseif rcount = 1 then
        compform3 := select rqst_name = p.first_name + ' ' +
            p.middl_init + ' ' + p.last_name from person p
            where :rqst_id = p.id_num;
    else
        rqst_id := '';
        answer := prompt 'There is more than one person in the ' +
            ' database with that last name and organization. Please '
            + 'enter the initials. Hit <CR> to continue';
    endif;
elseif rcount = 1 then
    compform3 := select rqst_name = p.first_name + ' ' +
        p.middl_init + ' ' + p.last_name from person p
        where :rqst_id = p.id_num;
else
    rqst_id := '';
    answer := prompt 'There is more than one person in the ' +
        'organization with the same initials. Contact your DBA';
endif;
endif;
endif;
endif;
if err_count = 0 then
    phone_num := area_code + phone_prefix + phone_suffix;
    zip_code := zip_main + zip_suffix;
    if mfg_code = cur_code then
        rcount := 0;
        compform3 := select rcount = count(*) from company
            where company.mfg_code = :mfg_code;
        if rcount = 0 then
            compform3 := select entry_date = _date(_bintim(0)) from company;
            insert into company (mfg_code, company_name, phone_num, source,
                alt_mfg_code, entry_date, comp_req, company_nam2, notes)
                values (:mfg_code, :company_name, :phone_num, :source,
                    :alt_mfg_code, :entry_date, :rqst_id, :company_nam2, :notes);
            answer := prompt 'The Company record was added to the database. ' +
                'Hit <CR> to continue';
        else
            if cur_name = company_name then
                update company c set phone_num = :phone_num, source = :source,
                    alt_mfg_code = :alt_mfg_code, entry_date = :entry_date,
                    comp_req = :rqst_id, company_nam2 = :company_nam2,
                    notes = :notes where :mfg_code = c.mfg_code;
                answer := prompt 'The Company record was revised in the ' +
                    'database. Hit <CR> to continue';
            else
                answer := prompt 'Do you really want to change the company ' +
                    'name?';
                if answer = 'Y' or answer = 'y' then
                    update company c set company_name = :company_name,
                        phone_num = :phone_num, source = :source,
                        alt_mfg_code = :alt_mfg_code, entry_date = :entry_date,
                        comp_req = :rqst_id, company_nam2 = :company_nam2,
                        notes = :notes where :mfg_code = c.mfg_code;
                endif;
            endif;
        endif;
    endif;
endif;

```

```

        insert into company_hist(current_code, old_mfg_code,
                                new_mfg_code, old_name) values (:mfg_code, :mfg_code,
                                                                :mfg_code, :cur_name);
        answer := prompt 'The Company record including the name was '
            + 'revised in the database. Hit <CR> to continue';
    else
        answer := prompt 'The Company record was not changed. Hit ' +
            '<CR> to continue';
        resume field company_name;
    endif;
endif;
endif;
else
    answer := prompt 'Did you intend to change the three character code?';
    if answer = 'Y' or answer = 'y' then
        insert into company_hist(current_code, new_mfg_code, old_mfg_code,
                                old_name) values (:mfg_code, :mfg_code, :cur_code, :cur_name);
        update company_hist ch set current_code = :mfg_code
            where ch.current_code = :cur_code;
        compform3 := select entry_date = _date(_bintim(0)) from company;
        update company c set mfg_code = :mfg_code,
            company_name = :company_name, phone_num = :phone_num,
            source = :source, alt_mfg_code = :alt_mfg_code,
            entry_date = :entry_date, comp_req = :rqst_id,
            company_name2 = :company_name2, notes = :notes
            where :cur_code = c.mfg_code;
        answer := prompt 'The Company record including the code was ' +
            'revised in the database. Hit <CR> to continue';
    else
        answer := 'The company record remained unchanged. Hit <CR> to ' +
            'continue';
        resume field mfg_code;
    endif;
endif;
rcount := 0;
compform3 := select rcount = count(*) from address
    where address.address_id = :cur_code;
if rcount = 0 then
    insert into address (address_id, strt_addr1, strt_addr2, city_name,
                        state_code, country_name, zip_code) values (:mfg_code, :strt_addr1,
                                                                    :strt_addr2, :city_name, :state_code, :country_name, :zip_code);
else
    update address a set strt_addr1 = :strt_addr1, address_id = :mfg_code,
        strt_addr2 = :strt_addr2, city_name = :city_name,
        state_code = :state_code, country_name = :country_name,
        zip_code = :zip_code where :cur_code = a.address_id;
endif;
endif;
}

'CLEAR' =
{
    clear field all;
    resume field mfg_code;
}

'FIND OLD' =
{
    if history[1].old_mfg_code = '' then
        if history[1].old_name = '' then

```

```

if alt_mfg_code = '' then
    answer := prompt 'You must enter a company code, alternate code ' +
        'or a company name. Hit <CR> to continue';
    resume field history.old_name;
else
    message 'Retrieving Information ...';
    compform3 := select mfg_code = c.mfg_code,
        company_name = c.company_name, strt_addr1 = a.strt_addr1,
        strt_addr2 = a.strt_addr2, city_name = a.city_name,
        state_code = a.state_code, zip_main = left(a.zip_code, 5),
        zip_suffix = shift(a.zip_code, -5),
        country_name = a.country_name, company_nam2 = c.company_nam2,
        notes = c.notes,
        area_code = left(c.phone_num, 3),
        phone_prefix = left(shift(c.phone_num, -3), 3),
        phone_suffix = shift(c.phone_num, -6),
        source = c.source, entry_date = c.entry_date,
        rqst_name = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
        rqst_orgn = p.asnd_orgn, alt_mfg_code = c.alt_mfg_code,
        rqst_code = p.mfg_code from company c, address a, person p
        where :alt_mfg_code = c.alt_mfg_code
            and c.mfg_code = a.address_id and c.comp_req = p.id_num;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        answer := prompt 'There is no company with that two-digit code '
            + 'in the database. Hit <CR> to continue';
        resume field alt_mfg_code;
    else
        history := select old_name = h.old_name,
            old_mfg_code = h.old_mfg_code, new_mfg_code = h.new_mfg_code
            from company_hist h where :mfg_code = h.current_code;
        endif;
    endif;
else
    compform3 := select cur_code = h.current_code from company_hist h
        where :history[1].old_name = h.old_name;
    history := select old_name = h.old_name,
        old_mfg_code = h.old_mfg_code, new_mfg_code = h.new_mfg_code
        from company_hist h where :history[1].old_name = h.old_name;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        answer := prompt 'There is no company with that name '
            + 'in the database. Hit <CR> to continue';
        resume field history.old_name;
    endif;
endif;
else
    compform3 := select cur_code = h.current_code from company_hist h
        where :history[1].old_mfg_code = h.old_mfg_code;
    history := select old_name = h.old_name,
        old_mfg_code = h.old_mfg_code, new_mfg_code = h.new_mfg_code
        from company_hist h where :history[1].old_mfg_code = h.old_mfg_code;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        answer := prompt 'There is no company with that old code '
            + 'in the database. Hit <CR> to continue';
        resume field history.old_mfg_code;
    else
        compform3 := select mfg_code = c.mfg_code, notes = c.notes,
            company_name = c.company_name, company_nam2 = c.company_nam2,

```



```

    strt_addr1 = a.strt_addr1,      strt_addr2 = a.strt_addr2,
    city_name = a.city_name, state_code = a.state_code,
    zip_main = left(a.zip_code, 5), zip_suffix = shift(a.zip_code, -5),
    country_name = a.country_name, area_code = left(c.phone_num, 3),
    phone_prefix = left(shift(c.phone_num, -3), 3),
    phone_suffix = shift(c.phone_num, -6),
    source = c.source, entry_date = c.entry_date,
    rqst_name = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
    rqst_orgn = p.asnd_orgn, alt_mfg_code = c.alt_mfg_code,
    rqst_code = p.mfg_code from company c, address a, person p
    where :cur_code = c.mfg_code
        and c.mfg_code = a.address_id and c.comp_req = p.id_num;
    endif;
endif;
if rcount >= 1 then
    cur_code := :mfg_code;
    cur_name := :company_name;
    if rqst_code != 'SBT' and rqst_code != 'SBJ' then
        compform3 := select rqst_orgn = p.mfg_code from company c, person p
            where :mfg_code = c.mfg_code and c.comp_req = p.id_num;
    endif;
endif;
}

'GET DETAIL' =
{
    message 'Retrieving Information ...';
    compform3 := select mfg_code = c.mfg_code, company_name = c.company_name,
        strt_addr1 = a.strt_addr1, strt_addr2 = a.strt_addr2,
        city_name = a.city_name, state_code = a.state_code,
        zip_main = left(a.zip_code, 3), zip_suffix = shift(a.zip_code, -5),
        country_name = a.country_name, area_code = left(c.phone_num, 3),
        phone_prefix = left(shift(c.phone_num, -3), 3),
        phone_suffix = shift(c.phone_num, -6),
        source = c.source, entry_date = c.entry_date,
        company_name2 = c.company_name2, notes = c.notes,
        rqst_name = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
        rqst_orgn = p.asnd_orgn, alt_mfg_code = c.alt_mfg_code,
        rqst_code = p.mfg_code from company c, address a, person p
        where :history.new_mfg_code = c.mfg_code
            and c.mfg_code = a.address_id and c.comp_req = p.id_num;
    cur_code := :mfg_code;
    cur_name := :company_name;
    if rqst_code != 'SBT' and rqst_code != 'SBJ' then
        compform3 := select rqst_orgn = p.mfg_code from company c, person p
            where :mfg_code = c.mfg_code and c.comp_req = p.id_num;
    endif;
}

'HELP', key frskey1 =
{
    help_forms (subject = 'Retrieving Information',
        file = 'db1:[dbaprdh.help]compform3.help');
}

'XIT', key frskey3 =
{
    RETURN;
}

'STOP', key frskey2 =

```

```
{ EXIT;  
}
```

```
/*
Initialize (answer = vchar(1), rcount = integer1, chosen_code = vchar(3)) =
{
}

'RETRIEVE' =
{
  message 'Retrieving Information ...';
  if companies[1].company_name = '' then
    if companies[1].city_name = '' then
      if companies[1].state_code = '' then
        answer := prompt 'You must enter either a Company Name, City, or' +
          ' State. Hit <CR> to continue.';
      else
        rcount := 0;
        compinq := select rcount = count (*) from state s
          where :companies[1].state_code = s.state_code;
        if rcount = 0 then
          answer := prompt 'The state code you have entered is invalid.' +
            ' Hit <CR> to continue.';
          resume field companies.state_code;
        else
          companies := select company_name = c.company_name,
            mfg_code = c.mfg_code, city_name = a.city_name,
            state_code = a.state_code from company c, address a
            where :companies[1].state_code = a.state_code
              and left(a.address_id, 3) = c.mfg_code;
          inquire_ingres (rcount = rowcount);
          if rcount = 0 then
            answer := prompt 'There are no Companies with that state ' +
              'code in PREMIS. Hit <CR> to continue';
            resume field companies.company_name;
          endif;
        endif;
      endif;
    else
      if companies[1].state_code = '' then
        companies := select company_name = c.company_name,
          mfg_code = c.mfg_code, city_name = a.city_name,
          state_code = a.state_code from company c, address a
          where :companies[1].city_name = a.city_name
            and left(a.address_id, 3) = c.mfg_code;
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
          answer := prompt 'There are no companies in PREMIS located ' +
            'in ' + companies[1].city_name + ' Hit <CR> to continue';
          resume field companies.company_name;
        endif;
      else
        rcount := 0;
        compinq := select rcount = count (*) from state s
          where :companies[1].state_code = s.state_code;
        if rcount = 0 then
          answer := prompt 'The state code you have entered is invalid.' +
            ' Hit <CR> to continue.';
          resume field companies.state_code;
        else

```

```

companies := select company_name = c.company_name,
mfg_code = c.mfg_code, city_name = a.city_name,
state_code = a.state_code from company c, address a
where :companies[1].state_code = a.state_code
and :companies[1].city_name = a.city_name
and left(a.address_id, 3) = c.mfg_code;
inquire_ingres (rcount = rowcount);
if rcount = 0 then
    answer := prompt 'There are no Companies located in ' +
:companies[1].city_name + ', ' + companies[1].state_code +
'. Hit <CR> to continue';
    resume field companies.company_name;
endif;
endif;
endif;
endif;
else
if companies[1].city_name = '' then
if companies[1].state_code = '' then
companies := select company_name = c.company_name,
mfg_code = c.mfg_code, city_name = a.city_name,
state_code = a.state_code from company c, address a
where :companies[1].company_name = c.company_name
and c.mfg_code = left(a.address_id, 3);
inquire_ingres (rcount = rowcount);
if rcount = 0 then
    answer := prompt 'There are no Companies stored in PREMIS ' +
'named ' + :companies[1].company_name +
'. Hit <CR> to continue';
    resume field companies.company_name;
endif;
else
rcount := 0;
compinq := select rcount = count (*) from state s
where :companies[1].state_code = s.state_code;
if rcount = 0 then
    answer := prompt 'The state code you have entered is invalid.' +
' Hit <CR> to continue.';
    resume field companies.state_code;
else
companies := select company_name = c.company_name,
mfg_code = c.mfg_code, city_name = a.city_name,
state_code = a.state_code from company c, address a
where :companies[1].company_name = c.company_name
and :companies[1].state_code = a.state_code
and c.mfg_code = left(a.address_id, 3);
inquire_ingres (rcount = rowcount);
if rcount = 0 then
    answer := prompt 'There are no Companies located in PREMIS' +
' named ' + :companies[1].company_name +
'. Hit <CR> to continue';
    resume field companies.company_name;
endif;
endif;
endif;
endif;
else
if companies[1].state_code = '' then
companies := select company_name = c.company_name,
mfg_code = c.mfg_code, city_name = a.city_name,
state_code = a.state_code from company c, address a

```

```

        where :companies[1].company_name = c.company_name
              and :companies[1].city_name   = a.city_name
              and c.mfg_code = left(a.address_id, 3);
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        answer := prompt 'There are no Companies located in ' +
            :companies[1].city_name + ' named ' +
            :companies[1].company_name + '. Hit <CR> to continue';
        resume field companies.company_name;
    endif;
else
    rcount := 0;
    compinq := select rcount = count (*) from state s
        where :companies[1].state_code = s.state_code;
    if rcount = 0 then
        answer := prompt 'The state code you have entered is invalid.' +
            ' Hit <CR> to continue.';
        resume field companies.state_code;
    else
        companies := select company_name = c.company_name,
            mfg_code = c.mfg_code, city_name = a.city_name,
            state_code = a.state_code from company c, address a
            where :companies[1].company_name = c.company_name
                and :companies[1].city_name   = a.city_name
                and :companies[1].state_code  = a.state_code
                and c.mfg_code = left(a.address_id, 3);
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
            answer := prompt 'There are no Companies located in ' +
                :companies[1].city_name + ', ' + :companies[1].state_code
                + ' named ' + :companies[1].company_name +
                '. Hit <CR> to continue';
            resume field companies.company_name;
        endif;
    endif;
endif;
endif;
endif;
}

'CLEAR' =
{
    clear field all;
    resume field companies.company_name;
}

'GET DETAIL' =
{
    chosen_code := companies.mfg_code;
    callframe compform3 (mfg_code := chosen_code);
}

'HELP', key frskey1 =
{
    help_forms (subject = 'Information on Company',
        file = 'db1:[dbaprdb.help]compinq.help');
}

'XIT', key frskey3 =
{
    RETURN;
}

```

```
'STOP', key frskey2 =  
{  
    EXIT;  
}
```

/*

Procedure for COMPMAN frame
Written by Caleb Crump

COMPMAN

*/

Initialize (answer = vchar(1))=

{
}

'DATA' =

{
 CALLFRAME COMPF0RM3;
}

'RETRIEVE' =

{
 CALLFRAME COMPINQ;
}

'GEN REPTS' =

{
 Message 'TWO reports (full list of companies) will be run' +
 ' in batch and print at "MYPRINTER";
 sleep 5;
 Answer := Prompt 'Hit <C> to cancel, or <RETURN> to continue.';
 If answer != 'C' and answer != 'c' then
 CALL SYSTEM 'SUBMIT DB1:[DBAPRDB.SOURCE]FULLCODES';
 endif;
}

'HELP', key frskey1 =

{
 help_forms (subject = 'COMPANY MAIN MENU',
 file = 'db1:[dbaprdb.help]compmain.help');
}

'XIT', key frskey3 =

{
 RETURN;
}

'STOP', key frskey2 =

{
 EXIT;
}

Procedure for DATA frame
Written by John Orman

```
/*  
*/  
'INFO' =  
{  
    CALLFRAME INFO ;  
}  
'ENTER' =  
{  
    CALLFRAME ENTER ;  
}  
'NEWDOC' =  
{  
    CALLFRAME NEWDOC ;  
}  
'REPORT' =  
{  
    CALLFRAME REPORT ;  
}  
/*  
'HELP', key frskey1 =  
{  
    help_forms (subject = ' callframes',  
        file = 'db1:[dbaprdb.help]data.help');  
}  
/*  
'XIT',key frskey3 =  
{  
    RETURN ;  
}  
'STOP', key frskey2 =  
{  
    EXIT ;  
}  
}
```



```

PROCEDURE DATE_SLASH(date_in = vchar(12),
                    month = vchar(2),
                    day = vchar(2),
                    year = vchar(2),
                    month_len = integer2,
                    day_len = integer2) =

```

```

{
/*

```

```

PROCEDURE: DATE_SLASH
GIVEN A DATE IN INGRES DATE FORMAT,
CREATE AN EIGHT CHARACTER VCHAR STRING IN THE
FORMAT MO/DA/YR
WRITTEN BY JOHN ORMAN 8/2/88

```

```

*/
if :date_in != '' then
/*
    have a valid date -
    extract the month, day and year
    must convert date_in back to date for date_part function to work
    shift year to eliminate '19' in '1988'
*/
    date_slash := select
        month = vchar(date_part('month', date(:date_in))),
        day   = vchar(date_part('day',   date(:date_in))),
        year  = shift(vchar(date_part('year', date(:date_in))), -2)
        from weapon ;
/*
    insert leading 0 for month and day, if necessary */
    date_slash := select
        month_len = length(:month)
        from weapon ;
    if month_len != 2 then
        month := '0' + month ;
    endif ;
    date_slash := select
        day_len = length(:day)
        from weapon ;
    if day_len != 2 then
        day := '0' + day ;
    endif ;
/*
    return date with slashes inserted
*/
    return month + '/' + day + '/' + year ;
else
/*
    date_in is blank - return blank */
    return '' ;
endif ;
}

```

```

/*
                                FRAME: DELETEDOC
                                DELETEDOC
                                OSL to fill DELETEDOC form
                                Written by John Orman 7/14/88

```

```

*/
initialize (
    rcount = integer,
    h_cer_date = vchar(8),
    h_dter_date = vchar(8),
    h_matl_list = vchar(8),
    h_rqrd_dt = vchar(8),
    h_e_comp_dt = vchar(8),
    h_ruff_dft_dt = vchar(8),
    h_req_ml_date = vchar(8),
    h_unit_stem = vchar(7),
    tmlrdt = vchar(8),
    trddt = vchar(8),
    tdocid = vchar(10),
    tdoctype = vchar(2),
    tdocnum = vchar(8)
) = {
    resume field doc_table ;
}

'YES' = {
/* ***** DOCUMENTS ***** */
/*      delete a data document -
/*      change data in unit_doc, data_documnt and document tables */
/*      extract document type and number */
deletedoc := select
    tdocid = squeeze(:doc_table.docmt_num)
    from document ;
deletedoc := select
    tdoctype = left(:tdocid, 2)
    from document ;
deletedoc := select
    tdocnum =
        squeeze(shift(:tdocid, -2))
    from document ;
if tdoctype != '' then
    if tdocnum != '' then
/*      user has entered all mandatory fields */
/*      ***** start update transaction ***** */
        begin transaction ;
/*      update unit_document table */
/*      see if document is in unit_doc table */
deletedoc := select
    h_unit_stem = unit_doc.unit_stem
    from unit_doc
    where
        unit_doc.doc_iss_num =
            :doc_table.issue_num
        and
        unit_doc.doc_type = :tdoctype
        and
        unit_doc.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      the document does not exist in unit_doc table */
    message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +
        :doc_table.issue_num +

```

```

        ' does not exist in the database' ;
    sleep 3 ;
else
/*
    the document is in unit_doc table - delete */
    delete from unit_doc
        where
            unit_doc.doc_iss_num =
                :doc_table.issue_num
            and
            unit_doc.doc_type = :tdoctype
            and
            unit_doc.doc_num = :tdocnum ;
    message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +
        :doc_table.issue_num +
        ' was deleted from the database' ;
    sleep 3 ;
endif ;

/*
    update document table
    delete from document
        where
            document.doc_iss_num =
                :doc_table.issue_num
            and
            document.doc_type = :tdoctype
            and
            document.doc_num = :tdocnum ;

/*
    update data_documnt table
    delete from data_documnt
        where
            data_documnt.doc_iss_num =
                :doc_table.issue_num
            and
            data_documnt.doc_type = :tdoctype
            and
            data_documnt.doc_num = :tdocnum ;

/*
    update doc_rough_dt table
    delete from doc_rough_dt
        where
            doc_rough_dt.doc_iss_num =
                :doc_table.issue_num
            and
            doc_rough_dt.doc_type = :tdoctype
            and
            doc_rough_dt.doc_num = :tdocnum ;

/*
    update doc_rq_ml_dt table
    delete from doc_rq_ml_dt
        where
            doc_rq_ml_dt.doc_iss_num =
                :doc_table.issue_num
            and
            doc_rq_ml_dt.doc_type = :tdoctype
            and
            doc_rq_ml_dt.doc_num = :tdocnum ;

/*
    *****
    end update transaction ***** */
end transaction ;
else
/*
    a document type was entered but no document number */
    message 'Document ' + tdoctype +
        ' must have document number' ;
    sleep 3 ;

```

```

endif ;
/*      a document with no document id was entered
      no error message since this could result from
      continuation of input line to end of line
*/
endif ;
RETURN ;
}
'HELP', key frskey1 =
{
    help_forms (subject = ' delete data document',
        file = 'db1:[dbaprdb.help]deletedoc.help');
}
'XIT', key frskey3 =
{
    RETURN;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

/*

Procedure for DOCHIST frame
Written by Norman Stevens

DOCHIST

*/

```
Initialize (w_pcode = vchar(4), w_cfg = vchar(7), w_modno = vchar(2),
           h_stat = vchar(1), rcount = integer1) = {
  resume field nomenclature;
}
```

'RETRIEVE' = {

/*

DSL to retrieve part number or nomenclature from PREMIS

*/

```
if nomenclature = '' then
  if part_num = '' then
    message 'You must enter either a part number or a nomenclature';
    sleep 5;
  else
    dochist := select nomenclature = c.com_nom_type + c.com_nom_num
      from component c where c.unit_stem = :part_num and
        c.unit_da_suf = :suffix;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      dochist := select w_pcode = w.wpn_pgm_code,
        w_cfg = w.wpn_test_cfg, w_modno = w.wpn_mod_num from weapon w
        where w.unit_stem = :part_num and w.unit_da_suf = :suffix;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        message 'No part with that part number can be found in PREMIS';
        sleep 5;
      else
        if w_cfg = '' then
          nomenclature := w_pcode + '-' + w_modno;
        else
          nomenclature := w_pcode + '/' + w_cfg + '-' + w_modno;
        endif;
      endif;
      w_pcode := ' ';
      w_cfg := '';
      w_modno := ' ';
    endif;
  endif;
else
  dochist := select part_num = c.unit_stem, suffix = c.unit_da_suf
    from component c where c.com_nom_type = left(:nomenclature, 2)
      and c.com_nom_num = shift(:nomenclature, -2);
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    message 'No PART with that nomenclature can be found in PREMIS';
    sleep 5;
  endif;
endif;
```

/*

DSL to retrieve unit description from PREMIS

*/

```
dochist := select description = u.unit_desc from unit_desc u
  where u.unit_stem = :part_num and u.unit_da_suf = :suffix;
```

/*

DSL to retrieve document data from PREMIS

*/

```

doc_hist := select wpn_pgm = m.wpn_pgm_code + '-' + m.wpn_mod_num,
  type = q.doc_type, engr_rels = q.doc_num, rev_num = q.doc_iss_num,
  comp_dt = q.issue_dt, mfg_name = c.company_name, status = q.status_code
  from mfg_inst m, qa_engr_rels q, company c, tms_inst t
  where (m.unit_stem = :part_num and m.unit_da_suf = :suffix) and
        t.mfg_inst_num = m.mfg_inst_num and m.mfg_code = c.mfg_code
        and q.doc_type = t.doc_type and q.doc_num = t.doc_num and
        q.doc_iss_num = t.doc_iss_num;
}

'CLEAR' = {
  clear field all;
  resume field nomenclature;
}
'HELP', key firskey1 =
{
  help_forms (subject = 'Retrieve document data',
    file = 'db1:[dbaprdb.help]dochist.help');
}
'XIT', key firskey3 =
{
  RETURN;
}
'STOP', key firskey2 =
{
  EXIT;
}

```

```
/*
```

Procedure for ENGRELs frame
Written by Norman Stevens

ENGR RELS

```
*/
Initialize (h_mfg_inst_num = vchar(4), h_type = vchar(1), rcount=integer,
            h_stat = vchar(4), w_pcode = vchar(4), w_cfg = vchar(7),
            w_modno = vchar(2), h_doc_num = vchar(8), h_iss_num = vchar(2)) =
{
    h_mfg_inst_num := '    ';
    resume field er_num;
}

'RETRIEVE' = {
/*
    OSL TO retrieve issue date and doc type
*/
    h_doc_num := er_num;
    h_iss_num := rev_num;
    clear field all;
    er_num := h_doc_num;
    rev_num := h_iss_num;
    if er_num = '' then
        message 'You must enter an engineering release number';
        sleep 5;
        resume field er_num;
    endif;
    if rev_num = '' then
        engrels := select rev_num = max(qer.doc_iss_num),
            doc_type = qer.doc_type from qa_engr_rels qer
            where qer.doc_num = :er_num;
    endif;
    engrels := select doc_type = qer.doc_type, issue_dt = qer.issue_dt,
        h_mfg_inst_num = t.mfg_inst_num, h_stat = qer.status_code,
        part_num = m.unit_stem, suffix = m.unit_da_suf,
        wpn_sys = m.wpn_pgm_code + '-' + m.wpn_mod_num
        from qa_engr_rels qer, mfg_inst m, tms_inst t
        where (qer.doc_num = :er_num and qer.doc_iss_num = :rev_num) and
            t.mfg_inst_num = m.mfg_inst_num and qer.doc_type = t.doc_type and
            qer.doc_num = t.doc_num and qer.doc_iss_num = t.doc_iss_num;
    inquire_ingres(rcount=rowcount);
    if rcount = 0 then
        message 'No Information on an Engineering Release with that number ' +
            'is stored in PREMIS';
        sleep 5;
    else
        if h_stat = 'ACC' then
            status = 'ACCEPTABLE';
        elseif h_stat = 'CA' then
            status = 'CONDITIONALLY ACCEPTABLE';
        elseif h_stat = 'AP' then
            status = 'ACTION PENDING';
        elseif h_stat = 'FC' then
            status = 'FULL CERTIFICATION';
        elseif h_stat = 'INST' then
            status = 'INSTRUCTIONS';
        elseif h_stat = 'P' then
            status = 'PLAN';
        elseif h_stat = 'NA' then
            status = 'NOT ACCEPTABLE';
        elseif h_stat = 'ACA' then
```

```

        status = 'ACCEPTABLE WITH CORREC. ACTION';
    elseif h_stat = 'PC' then
        status = 'PARTIAL CERTIFICATION';
    elseif h_stat = 'NR' then
        status = 'NOT REQUIRED';
    elseif h_stat = 'REQ' then
        status = 'REEVALUATION REQUIRED';
    else
        status = h_stat;
    endif;
endif;

/*
    DSL TO retrieve the nomenclature
*/
engrels := select nomenclature = cmpnt.com_nom_type + cmpnt.com_nom_num
    from component cmpnt where cmpnt.unit_stem = :part_num
    and cmpnt.unit_da_suf = :suffix;
inquire_ingres(rcount=rowcount);
if rcount=0 then
    engrels := select w_pcode = w.wpn_pgm_code, w_cfg = w.wpn_test_cfg,
        w_modno = w.wpn_mod_num from weapon w
        where w.unit_stem = :part_num and w.unit_da_suf = :suffix;
    if w_cfg = '' then
        nomenclature := w_pcode + '-' + w_modno;
    else
        nomenclature := w_pcode + '/' + w_cfg + '-' + w_modno;
    endif;
endif;
if desn_phon = '-' then
    desn_phon := '';
endif;

/*
    DSL TO retrieve the description
*/
engrels := select description = u_d.unit_desc from unit_desc u_d
    where u_d.unit_stem = :part_num and u_d.unit_da_suf = :suffix;

/*
    DSL TO retrieve the design and qa engineer information
*/
engrels := select qa_orgn = p.asnd_orgn,
    qa_engr = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
    qa_phone = left(shift(p.phone_num, -5), 1) + '-' +
    shift(p.phone_num, -6) from unit u, person p
    where p.id_num = u.da_qa_engr and
    (u.unit_stem = :part_num and u.unit_da_suf = :suffix);
if qa_phone = '-' then
    qa_phone := '';
endif;
engrels := select desn_orgn = p.asnd_orgn,
    desn_engr = p.first_name + ' ' + p.middl_init + ' ' + p.last_name,
    desn_phon = left(shift(p.phone_num, -5), 1) + '-' +
    shift(p.phone_num, -6) from unit u, person p
    where p.id_num = u.da_desn_engr and
    (u.unit_stem = :part_num and u.unit_da_suf = :suffix);
if desn_phon = '-' then
    desn_phon := '';
endif;

/*
    DSL TO retrieve the document dates
*/

```



```
engrels := select est_comp_dt = d.est_comp_dt, rqr_dt = d.required_dt
from document d where d.doc_type = :doc_type and d.doc_num = :er_num
and d.doc_iss_num = :rev_num;
```

```
/*
```

```
DSL TO retrieve next assembly information
```

```
*/
```

```
next_as := select nomen = cmpnt.com_nom_type + cmpnt.com_nom_num,
u_desc = u_d.unit_desc from component cmpnt, unit_desc u_d, gozinto g
where (g.part_stem = :part_num and g.part_da_suf = :suffix) and
(g.unit_stem = cmpnt.unit_stem and g.unit_da_suf = cmpnt.unit_da_suf)
and (g.unit_stem = u_d.unit_stem and g.unit_da_suf = u_d.unit_da_suf);
```

```
/*
```

```
DSL TO retrieve the reasons for a QER not being acceptable
```

```
*/
```

```
if h_stat != 'A' then
reason := select problems = q_p.problem_desc from qer_problems q_p
where q_p.doc_type = :doc_type and q_p.doc_num = :er_num and
q_p.doc_iss_num = :rev_num;
endif;
```

```
}
```

```
'CLEAR' = {
clear field all;
resume field er_num;
```

```
}
```

```
'HELP', key frskey1 =
```

```
{
```

```
help_forms (subject = 'To Retrieve Engineer Release Info',
file = 'db1:[dbaprdh.help]engrels.help');
```

```
}
```

```
'XIT', key frskey3 =
```

```
{
```

```
RETURN;
```

```
}
```

```
'STOP', key frskey2 =
```

```
{
```

```
EXIT;
```

```
}
```



```

mode 'query';
h_data_num := '';
h_qa_num := '';
h_rel_num := '';
h_desn_num := '';
h_desn_code := '';
h_pragy_code := '';
h_pragy_name := '';
h_prod_num := '';
h_qc_num := '';
h_data_name := '';
h_qa_name := '';
h_rel_name := '';
h_desn_name := '';
h_data_div := '';
h_qa_div := '';
h_rel_div := '';
h_desn_div := '';
h_prod_name := '';
h_qc_name := '';
resume field nomen ;
}
'RETRIEVE' = {
/*
    user inputs either nomenclature or part number -
    see which field has cursor
*/
    inquire_forms field " " (FNAME = name);
    errflg := 0 ;
    if FNAME = 'nomen' then
/*
        cursor is in nomenclature field */
/*
        clear all old fields, restore entered nomenclature */
        tnomen := :nomen ;
        clear field all ;
        nomen := :tnomen ;
        if nomen != '' then
/*
            nomenclature has been input */
/*
            ***** Assumes that nomenclature is in form of component nomenclature */
/*
            split the nomenclature into type and suffix */
/*
            get first two characters of nomenclature */
            enter := select tnomt = left(:nomen, 2)
                        from component ;
            nomflg = 'NO' ;
            enter := select nomflg = 'YES'
                        from component
                        where :tnomt = '[A-Z]*' ;
            if nomflg = 'YES' then
/*
                have a nomenclature with a non-blank type */
/*
                get nomenclature number */
                enter := select tnemn = shift(:nomen, -2)
                            from component ;
            else
/*
                nomenclature is in part number format */
                tnomt := '' ;
                tnemn := :nomen ;
            endif ;
/*
            get the part number */
            enter := select part_num = component.unit_stem,
                        part_suf = component.unit_da_suf
                        from component

```

```

                                where
                                component.com_nom_type = :tnomt
                                and
                                component.com_nom_num = :tnomn ;
/*                                see if found part number */
if part_num = '' then
    message 'No part number corresponding to the ' +
        'entered nomenclature was found' ;
    sleep 3;
    errflg := 1 ;
endif;
endif;
else
/* if field not in nomenclature field, assume part number is to be used */
/* clear all old fields, restore entered part number */
tpartn := :part_num ;
tparts := :part_suf ;
clear field all ;
part_num := :tpartn ;
part_suf := :tparts ;
if part_num != '' then
/* user has input part number */
/* now find nomenclature from component table */
tnomt := '' ;
tnomn := '' ;
enter := select tnomt = component.com_nom_type,
                tnomn = component.com_nom_num
                from component
                where
                component.unit_stem = :part_num
                and
                component.unit_da_suf = :part_suf ;
/* see if found part in component table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/* not find part in component table - try weapon table */
tnomt := '' ;
tnomn := '' ;
enter := select twpnpgm = weapon.wpn_pgm_code,
                twpncfg = weapon.wpn_test_cfg,
                twpnmod = weapon.wpn_mod_num
                from weapon
                where
                weapon.unit_stem = :part_num
                and
                weapon.unit_da_suf = :part_suf ;
/* see if found part in weapon table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
    errflg := 1 ;
    message 'No nomenclature corresponding to the ' +
        'entered part number was found' ;
    sleep 3;
else
/* found part in weapon table */
/* now concatenate pgm_code, test_cfg, and mod_num
   to form nomenclature and put on form */
nomen := twpnpgm + '/' + twpncfg + '-' + twpnmod ;
endif;
else

```

```

/*          found part in component table          */
/*          now concatenate nomenclature type and number and
           put on form          */
           nomen := :tnomt + :tnomn ;
       endif;
   endif;
endif;
if nomen != '' or part_num != '' then
if errilg = 0 then
/*          have a valid part to work with - load other form fields          */
/*          ***** now find suppliers and weapon programs *****          */
/*          - not every unit has a mfg_inst          */
enter.sup_wepn := select supplier = company.company_name,
                        wepn_pgm = mfg_inst.wpn_pgm_code +
                        '-' +
                        mfg_inst.wpn_mod_num
                        from company, mfg_inst
                        where
                        mfg_inst.unit_stem = :part_num
                        and
                        mfg_inst.unit_da_suf = :part_suf
                        and
                        company.mfg_code = mfg_inst.mfg_code ;
/*          ***** now find production agency *****          */
h_pragy_code := '' ;
enter := select
                        prod_agency = company.company_name,
                        h_pragy_code = company.mfg_code
                        from company, unit
                        where
                        unit.unit_stem = :part_num
                        and
                        unit.unit_da_suf = :part_suf
                        and
                        company.mfg_code = unit.prn_mfg_code ;
h_pragy_name := :prod_agency ;
if prod_agency = '' then
message 'No production agency for the ' +
        'entered part was found' ;
sleep 3;
endif;
/*          get engineer's info - retain last displayed name
           and employee or person id numbers.
           those will tell whether name has been changed by
           data entry and whether to update database if
           the 'ADD/REVISE' option is taken after a RETRIEVE
*/
/*          ***** da design engineer *****          */
/*          get design engineer from unit instead of component jlo 9/9/88          */
h_desn_num := '' ;
enter := select h_desn_num = da_empl.empl_num,
                save design agency code jlo 1/9/89          */
                h_desn_code = person.mfg_code,
                designngr = person.first_name +
                ',' +
                person.middl_init +
                ',' +
                person.last_name,
                desn_div = person.asnd_orgn,
                desn_phon = '(' +

```

```

        left(person.phone_num,3) +
        ')' +
        left(shift(person.phone_num, -3), 3) +
        '-' +
        shift(person.phone_num, -6)
    from unit, person, da_empl
    where
        unit.unit_stem = :part_num
    and
        unit.unit_da_suf = :part_suf
    and
        unit.da_desn_engr = da_empl.empl_num
    and
        da_empl.id_num = person.id_num ;
    h_desn_name := :designengr ;
    h_desn_div := :desn_div ;
/*      blank out phone if not available      */
    if desn_phon = '()-' then
        desn_phon := '' ;
    endif ;
/* *****      da data engineer      *****      */
    h_data_num = '' ;
    enter := select h_data_num = da_empl.empl_num,
        data_engr = person.first_name +
            , , +
            person.middl_init +
            , , +
            person.last_name,
        data_div = person.asnd_orgn,
        data_phon = '(' +
            left(person.phone_num,3) +
            ')' +
            left(shift(person.phone_num, -3), 3) +
            '-' +
            shift(person.phone_num, -6)
    from unit, person, da_empl
    where
        unit.unit_stem = :part_num
    and
        unit.unit_da_suf = :part_suf
    and
        unit.da_data_engr = da_empl.empl_num
    and
        da_empl.id_num = person.id_num ;
    h_data_name := :data_engr ;
    h_data_div := :data_div ;
/*      blank out phone if not available      */
    if data_phon = '()-' then
        data_phon := '' ;
    endif ;
/* *****      da quality assurance engineer      *****      */
    h_qa_num = '' ;
    enter := select h_qa_num = da_empl.empl_num,
        qa_engr = person.first_name +
            , , +
            person.middl_init +
            , , +
            person.last_name,
        qa_div = person.asnd_orgn,
        qa_phon = '(' +

```

```

                                left(person.phone_num,3) +
                                ')', +
                                left(shift(person.phone_num, -3), 3) +
                                ',_' +
                                shift(person.phone_num, -6)
from unit, person, da_empl
where
unit.unit_stem = :part_num
and
unit.unit_da_suf = :part_suf
and
unit.da_qa_engr = da_empl.empl_num
and
da_empl.id_num = person.id_num ;
h_qa_name := :qa_engr ;
h_qa_div := :qa_div ;
/*          blank out phone if not available          */
if qa_phon = '()-' then
    qa_phon := '' ;
endif ;
/* *****          da reliability engineer          ***** */
h_rel_num = '' ;
enter := select h_rel_num = da_empl.empl_num,
                rel_engr = person.first_name +
                ',,' +
                person.middl_init +
                ',,' +
                person.last_name,
rel_div = person.asnd_orgn,
rel_phon = '(' +
            left(person.phone_num,3) +
            ')', +
            left(shift(person.phone_num, -3), 3) +
            ',_' +
            shift(person.phone_num, -6)
from unit, person, da_empl
where
unit.unit_stem = :part_num
and
unit.unit_da_suf = :part_suf
and
unit.da_re_engr = da_empl.empl_num
and
da_empl.id_num = person.id_num ;
h_rel_name := :rel_engr ;
h_rel_div := :rel_div ;
/*          blank out phone if not available          */
if rel_phon = '()-' then
    rel_phon := '' ;
endif ;
/* *****          pa product engineer          ***** */
h_prod_num := '' ;
enter := select h_prod_num = person.id_num,
                prod_engr = person.first_name +
                ',,' +
                person.middl_init +
                ',,' +
                person.last_name,
prod_div = person.asnd_orgn,
prod_phon = '(' +

```

```

                                left(person.phone_num,3) +
                                ')', +
                                left(shift(person.phone_num, -3), 3) +
                                ',_' +
                                shift(person.phone_num, -6)
                                from unit, person
                                where
                                unit.unit_stem = :part_num
                                and
                                unit.unit_da_suf = :part_suf
                                and
                                unit.pa_prod_engr = person.id_num ;
/*                                blank out phone if not available */
                                if prod_phon = '()-' then
                                prod_phon := '';
                                endif ;
/*
***** pa quality control engineer ***** */
                                h_qc_num := '';
                                enter := select h_qc_name = person.id_num,
                                qc_engr = person.first_name +
                                ',,' +
                                person.middl_init +
                                ',,' +
                                person.last_name,
                                qc_div = person.asnd_orgn,
                                qc_phon = '(' +
                                left(person.phone_num,3) +
                                ')', +
                                left(shift(person.phone_num, -3), 3) +
                                ',_' +
                                shift(person.phone_num, -6)
                                from unit, person
                                where
                                unit.unit_stem = :part_num
                                and
                                unit.unit_da_suf = :part_suf
                                and
                                unit.pa_qc_engr = person.id_num ;
/*                                blank out phone if not available */
                                if qc_phon = '()-' then
                                qc_phon := '';
                                endif ;
                                endif
                                else
/*                                neither nomenclature nor part number entered */
                                message 'Nomenclature or part number is a required field';
                                sleep 3;
                                endif;
                                resume field nomen ;
                                }
                                'UPDATE' = {
/*
                                Revise information for part already in database
                                user inputs either nomenclature or part number or both -
                                do not check cursor - may be in data fields
*/
                                newflg := 1 ;
                                chk_part_num := 1 ;

```



```

tnomt := '';
tnomn := '';
if nomen != '' then
/*          nomenclature has been input */
/* ***** Assumes that nomenclature is in form of component nomenclature */
/*          split the nomenclature into type and suffix */
/*          get first two characters of nomenclature */
enter := select tnomt = left(:nomen, 2)
          from component ;
/*          check validity of nomenclature type */
enter := select h_dummy = com_nom_type.type
          from   com_nom_type
          where com_nom_type.type = :tnomt ;
inquire_ingres(rcount = rowcount) ;
if rcount = 0 then
/*          nomenclature does not have a valid type */
message 'Nomenclature type ' + :tnomt + ' is invalid - ' +
        'blank nomenclature used' ;
sleep 3;
tnomt := '';
nomen := '';
else
/*          get nomenclature number */
enter := select tnomn = shift(:nomen, -2)
          from component ;
/*          get the part number */
enter := select part_num = component.unit_stem,
          part_suf = component.unit_da_suf
          from component
          where
            component.com_nom_type = :tnomt
            and
            component.com_nom_num = :tnomn ;
/*          see if found part number */
inquire_ingres(rcount = rowcount) ;
if rcount = 0 then
message 'No part number corresponding to the ' +
        'entered nomenclature was found' ;
sleep 3;
else
/*          found part under nomenclature -
          do not use entered part number, if any */
newflg := 0 ;
chk_part_num := 0 ;
endif;
endif;
endif;
if chk_part_num = 1 then
/*          did not find part under nomenclature - look at part number */
/*          now find nomenclature from component table */
h_nomt := '';
h_nomn := '';
enter := select h_nomt = component.com_nom_type,
          h_nomn = component.com_nom_num
          from component
          where
            component.unit_stem = :part_num
            and
            component.unit_da_suf = :part_suf ;
/*          see if found part in component table */

```

```

        inquire_ingres (rcount = rowcount) ;
        if rcount = 0 then
/*          not find part in component table - try weapon table */
            enter := select twpnpgm = weapon.wpn_pgm_code,
                           twpncfg = weapon.wpn_test_cfg,
                           twpnmod = weapon.wpn_mod_num
                           from weapon
                           where
                           weapon.unit_stem = :part_num
                           and
                           weapon.unit_da_suf = :part_suf ;
/*          see if found part in weapon table */
            inquire_ingres (rcount = rowcount) ;
            if rcount = 0 then
                message 'No nomenclature corresponding to the ' +
                        'entered part number was found' ;
                sleep 3;
            else
/*          found part in weapon table */
                newflg := 0 ;
/*          now concatenate pgm_code, test_cfg, and mod_num
                to form nomenclature and put on form */
                nomen := twpnpgm + '/' + twpncfg + '-' + twpnmod ;
            endif;
        else
/*          found part in component table */
            newflg := 0 ;
/*          do not put database nomenclature on form -
            use what was entered, even if blank */
            if :nomen != :h_nomt + :h_nomn then
/*          have a difference in database and
            entered form nomenclature */
                message 'Nomenclature for part to be changed from ' +
                        :h_nomt + :h_nomn + ' to ' + :nomen ;
            endif ;
        endif ;
    endif ;
/*    must have found part in database to update part info */
    if newflg = 0 then
/*        check database integrity for this part */
/*        check existence in unit table */
        enter := select h_dummy = unit.unit_stem
                   from unit
                   where unit.unit_stem = :part_num
                   and
                   unit.unit_da_suf = :part_suf ;
        inquire_ingres (rcount = rowcount) ;
        err_unit := 0 ;
        if rcount = 0 then
/*          part not in unit table -
            but was in component or weapon table - error */
            message 'Part in component table but not unit table' +
                    ' - database error' ;
            sleep 3 ;
            err_unit := 1 ;
        endif ;
/*          if had database integrity error - do not process update */
        if err_unit = 0 then
/*          set production agency code */
            chg_pragy := 0 ;

```

```

if prod_agency != :h_pragy_name then
  chg_pragy := 1 ;
  h_pragy_code := '' ;
  h_pragy_name := '' ;
  /*
  /*      production agency may be input as code or synonym */
  /*      check for code */
  enter := select h_pragy_name = company.company_name
            from   company
            where  :prod_agency = company.mfg_code
            or
                  :prod_agency = company.company_name ;
  inquire_ingres (rcount = rowcount) ;
  if rcount != 0 then
    /*      valid code was entered      */
    h_pragy_code := :prod_agency ;
    prod_agency := :h_pragy_name ;
  else
    /*      not a valid code - try synonym table */
    enter := select h_pragy_code = company_syn.mfg_code
                  from   company_syn
                  where  :prod_agency = company_syn.company_name ;
    inquire_ingres (rcount = rowcount) ;
    if rcount = 0 then
      /*      not a valid code or synonym - error */
      message prod_agency +
        ' is invalid production agency code or name' ;
      sleep 3 ;
    else
      /*      replace the synonym company name with the complete company name */
      enter := select prod_agency = company.company_name
                    from   company
                    where  :h_pragy_code = company.mfg_code ;
      h_pragy_name := :prod_agency ;
    endif ;
  endif ;
endif ;
/*      for da employees, convert entered last names to da employee numbers */
/*      design engineer */
if :designngr != :h_desn_name or
:h_desn_div != :desn_div then
  h_desn_num := '' ;
  h_desn_code := '' ;
  h_desn_name := '' ;
  h_desn_div := '' ;
  /*      see how many design engineers with same last name */
  enter := select rcount = count (*) from person p
            where p.last_name = :designngr and p.asnd_orgn = :desn_div ;
  if rcount = 0 then
    /*      not under last name - try initials + last name */
    enter := select rcount = count(*) from person p
              where p.asnd_orgn = :desn_div
              and :designngr = left(p.first_name, 1) + " " + p.middl_init
              + " " + p.last_name ;
    if rcount = 0 then
      /*      name for design engineer is not a valid person */
      message 'There is no person named ' + :designngr +
        ' who works for ' + :desn_div ;
      sleep 3 ;
      designngr := '' ;
      desn_div := '' ;
    endif ;
  endif ;
endif ;

```

```

        desn_phon := '' ;
    elseif rcount = 1 then
/*      found design engineer by initials and last name */
        enter := select h_desn_num = p.id_num from person p
            where p.asnd_orgn = :desn_div
            and :designngr = left(p.first_name, 1) + " " +
                p.middl_init + " " + p.last_name;
    else
/*      found more than one design engineer by initials and last name */
        message 'More than one engr named ' +
            :designngr + ' works for ' + :desn_div + '. Please call '
            + 'your DBA.';
        sleep 3 ;
        designngr := '' ;
        desn_div := '' ;
        desn_phon := '' ;
    endif;
elseif rcount = 1 then
/*      found design engineer by last name */
        enter := select h_desn_num = p.id_num
            from person p
            where p.asnd_orgn = :desn_div and
                :designngr = p.last_name;
else
/*      duplicate last names in person table */
        message 'More than one engineer named ' +
            :designngr + ' works for ' + desn_div + '. Please enter '
            + 'the initials.';
        sleep 3 ;
        designngr := '' ;
        desn_div := '' ;
        desn_phon := '' ;
endif;
if rcount = 1 then
/*      have found design engineer in person table - h_desn_num has
        person's id number
        see if in design engineer table */
        enter := select h_desn_num = da_empl.id_num
            from da_empl, da_desn_engr
            where
                da_desn_engr.empl_num = da_empl.empl_num and
                da_empl.id_num = :h_desn_num ;
        inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*      have input invalid name for design engineer */
        message :designngr + ' is not a valid design engineer' ;
        sleep 3 ;
        h_desn_num := '' ;
        h_desn_code := '' ;
        h_desn_name := '' ;
        designngr := '' ;
        desn_div := '' ;
        desn_phon := '' ;
else
/*      have valid design engineer */
/*      retrieve information from person table by id_num */
        enter := select h_desn_code = person.mfg_code,
            h_desn_name = person.first_name +
                ' ' +
                person.middl_init +

```

```

        ' ' +
        person.last_name,
    desn_phon = '(' +
        left(person.phone_num,3) +
        ') ' +
        left(shift(person.phone_num, -3), 3) +
        ' ' +
        shift(person.phone_num, -6)
    from person
    where
        person.id_num = :h_desn_num ;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
/*
        could not retrieve design engineer -
        database error */
        message 'Error retrieving design engineer information - ' +
            'contact DBA' ;
        sleep 3 ;
        h_desn_num := '' ;
        h_desn_code := '' ;
        h_desn_name := '' ;
        designengr := '' ;
        desn_div := '' ;
        desn_phon := '' ;
    else
        designengr := :h_desn_name ;
        h_desn_div := :desn_div ;
/*
        blank out phone if not available */
        if desn_phon = '()-' then
            desn_phon := '' ;
        endif
    endif
endif ;
endif ;
endif ;
endif ;

/*
        data engineer */
    if data_engr != :h_data_name or
        :h_data_div != :data_div then
        h_data_num := '' ;
        h_data_name := '' ;
        h_data_div := '' ;
        enter := select h_data_num = da_empl.empl_num,
            h_data_name = person.first_name +
                ' ' +
                person.middl_init +
                ' ' +
                person.last_name,
            data_phon = '(' +
                left(person.phone_num,3) +
                ') ' +
                left(shift(person.phone_num, -3), 3) +
                ' ' +
                shift(person.phone_num, -6)
            from da_empl, person
        where
            person.last_name = :data_engr
            and
            person.asnd_orgn = :data_div
            and

```

```

        da_empl.id_num = person.id_num ;
/*      see if found data engineer in person and da_empl table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      name for data engineer is not a valid person */
message 'There is no person named ' + :data_engr +
        ' who works for ' + :data_div ;
sleep 3 ;
data_engr := '' ;
data_div := '' ;
data_phon := '' ;
else
/*      have found data engineer in person table -
        see if in data engineer table */
enter := select h_data_num = da_data_engr.empl_num
        from da_data_engr
        where
            da_data_engr.empl_num = :h_data_num ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      have input invalid name for data engineer */
message :data_engr + ' is not a valid data engineer' ;
sleep 3 ;
h_data_num := '' ;
h_data_name := '' ;
data_engr := '' ;
data_div := '' ;
data_phon := '' ;
else
/*      have valid data engineer */
data_engr := :h_data_name ;
h_data_div := :data_div ;
/*      blank out phone if not available */
if data_phon = '()-' then
    data_phon := '' ;
endif ;
endif ;
endif ;
endif ;
/*      qa engineer */
if qa_engr != :h_qa_name or
:h_qa_div != :qa_div then
h_qa_num := '' ;
h_qa_name := '' ;
h_qa_div := '' ;
enter := select h_qa_num = da_empl.empl_num,
        h_qa_name = person.first_name +
            , , +
            person.middl_init +
            , , +
            person.last_name,
        qa_phon = '(' +
            left(person.phone_num,3) +
            ')' +
            left(shift(person.phone_num, -3), 3) +
            '-' +
            shift(person.phone_num, -6)
        from da_empl, person
        where
            person.last_name = :qa_engr

```

```

        and
        person.asnd_orgn = :qa_div
        and
        da_empl.id_num = person.id_num ;
/*
    see if found qa engineer in person and da_empl table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*
    name for qa engineer is not a valid person */
    message 'There is no person named ' + :qa_engr +
        ' who works for ' + :qa_div ;
    sleep 3 ;
    qa_engr := '' ;
    qa_div := '' ;
    qa_phon := '' ;
else
/*
    have found qa engineer in person table -
    see if in qa engineer table */
    enter := select h_qa_num = da_qa_engr.empl_num
        from da_qa_engr
        where
            da_qa_engr.empl_num = :h_qa_num ;
    inquire_ingres (rcount = rowcount) ;
    if rcount = 0 then
/*
        have input invalid name for qa engineer */
        message :qa_engr + ' is not a valid qa engineer' ;
        sleep 3 ;
        h_qa_num := '' ;
        h_qa_name := '' ;
        qa_engr := '' ;
        qa_div := '' ;
        qa_phon := '' ;
    else
/*
        have valid qa engineer */
        qa_engr := :h_qa_name ;
        h_qa_div := :qa_div ;
/*
        blank out phone if not available */
        if qa_phon = '()-' then
            qa_phon := '' ;
        endif ;
    endif ;
endif ;
endif ;
/*
    reliability engineer */
if rel_engr != :h_rel_name or
:h_rel_div != :rel_div then
    h_rel_num := '' ;
    h_rel_name := '' ;
    h_rel_div := '' ;
    enter := select h_rel_num = da_empl.empl_num,
        h_rel_name = person.first_name +
            ' ' +
            person.middl_init +
            ' ' +
            person.last_name,
        rel_phon = '(' +
            left(person.phone_num,3) +
            ')' +
            left(shift(person.phone_num, -3), 3) +
            '-' +
            shift(person.phone_num, -6)

```

```

        from da_empl, person
      where
        person.last_name = :rel_engr
        and
        person.asnd_orgn = :rel_div
        and
        da_empl.id_num = person.id_num ;
/*      see if found reliability engineer in person and da_empl table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      name for reliability engineer is not a valid person */
message 'There is no person named ' + :rel_engr +
        ' who works for ' + :rel_div ;
sleep 3 ;
rel_engr := '' ;
rel_div := '' ;
rel_phon := '' ;
else
/*      have found reliability engineer in person table -
      see if in reliability engineer table */
enter := select h_rel_num = da_rel_engr.empl_num
        from da_rel_engr
        where
          da_rel_engr.empl_num = :h_rel_num ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      have input invalid name for reliability engineer */
message :rel_engr + ' is not a valid reliability engineer' ;
sleep 3 ;
h_rel_num := '' ;
h_rel_name := '' ;
rel_engr := '' ;
rel_div := '' ;
rel_phon := '' ;
else
/*      have valid reliability engineer */
rel_engr := :h_rel_name ;
h_rel_div := rel_div ;
/*      blank out phone if not available */
if rel_phon = '()-' then
rel_phon := '' ;
endif ;
endif ;
endif ;
endif ;
/*      PA production engineer */
if prod_engr != :h_prod_name or
(chg_pragy = 1 and prod_engr != '') then
h_prod_num := '' ;
h_prod_name := '' ;
/*      see how many production engineers with same last name */
enter := select rcount = count (*) from person p
        where p.last_name = :prod_engr and p.mfg_code = :h_pragy_code ;
if rcount = 0 then
/*      not under last name - try initials + last name */
enter := select rcount = count(*) from person p
        where p.mfg_code = :h_pragy_code
        and :prod_engr = left(p.first_name, 1) + " " + p.middl_init
        + " " + p.last_name ;
if rcount = 0 then

```



```

/*          name for production engineer is not a valid person */
message 'There is no person named ' + :prod_engr +
        ' who works for ' + :h_pragy_code ;
sleep 3 ;
prod_engr := '' ;
prod_div  := '' ;
prod_phon := '' ;
elseif rcount = 1 then
/*          found production engineer by initials and last name */
enter := select h_prod_num = p.id_num from person p
        where p.mfg_code = :h_pragy_code
        and :prod_engr = left(p.first_name, 1) + " " +
          p.middl_init + " " + p.last_name;
else
/*          found more than one production engineer by initials and last name */
message 'More than one engr named ' +
        :prod_engr + ' works for ' + :h_pragy_code +
        '. Please call your DBA.';
sleep 3 ;
prod_engr := '' ;
prod_div  := '' ;
prod_phon := '' ;
endif;
elseif rcount = 1 then
/*          found production engineer by last name */
enter := select h_prod_num = p.id_num
        from person p
        where p.mfg_code = :h_pragy_code and
          :prod_engr = p.last_name;
else
/*          duplicate last names in person table */
message 'More than one engineer named ' +
        :prod_engr + ' works for ' + :h_pragy_code + '. Please enter '
        + 'the initials.';
sleep 3 ;
prod_engr := '' ;
prod_div  := '' ;
prod_phon := '' ;
endif;
if rcount = 1 then
/*          have found production engineer in person table - h_prod_num has
           person's id number
           see if in production engineer table */
enter := select h_prod_num = pa_prod_engr.id_num
        from pa_prod_engr
        where
          pa_prod_engr.id_num = :h_prod_num ;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*          have input invalid name for production engineer */
message :prod_engr + ' is not a valid production engineer' ;
sleep 3 ;
h_prod_num := '' ;
h_prod_name := '' ;
prod_engr := '' ;
prod_div  := '' ;
prod_phon := '' ;
else
/*          have valid production engineer */
/*          retrieve information from person table by id_num */

```

```

enter := select h_prod_name = person.first_name +
               , , +
               person.middl_init +
               , , +
               person.last_name,
               prod_div = person.asnd_orgn,
               prod_phon = '(' +
               left(person.phone_num,3) +
               ')', +
               left(shift(person.phone_num, -3), 3) +
               ',_', +
               shift(person.phone_num, -6)
               from person
               where
               person.id_num = :h_prod_num ;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*
               could not retrieve production engineer -
               database error */
               message 'Error retrieving production engineer ' +
               'information - contact DBA' ;
               sleep 3 ;
               h_prod_num := '' ;
               h_prod_name := '' ;
               prod_engr := '' ;
               prod_div := '' ;
               prod_phon := '' ;
else
               prod_engr := :h_prod_name ;
/*
               blank out phone if not available */
               if prod_phon = '()-' then
               prod_phon := '' ;
               endif
               endif ;
               endif ;
               endif ;
               endif ;
/*
               PA qc engineer */
if qc_engr != :h_qc_name or
(chg_pragy = 1 and qc_engr != '') then
               h_qc_num := '' ;
               h_qc_name := '' ;
/*
               see how many qc engineers with same last name */
enter := select rcount = count (*) from person p
               where p.last_name = :qc_engr and p.mfg_code = :h_pragy_code ;
if rcount = 0 then
/*
               not under last name - try initials + last name */
enter := select rcount = count(*) from person p
               where p.mfg_code = :h_pragy_code
               and :qc_engr = left(p.first_name, 1) + " " + p.middl_init
               + " " + p.last_name;
if rcount = 0 then
/*
               name for qc engineer is not a valid person */
               message 'There is no person named ' + :qc_engr +
               ' who works for ' + :h_pragy_code ;
               sleep 3 ;
               qc_engr := '' ;
               qc_div := '' ;
               qc_phon := '' ;
elseif rcount = 1 then

```

```

/*          found qc engineer by initials and last name */
enter := select h_qc_num = p.id_num from person p
        where p.mfg_code = :h_pragy_code
        and :qc_engr = left(p.first_name, 1) + " " +
        p.middl_init + " " + p.last_name;

else
/* found more than one qc engineer by initials and last name */
message 'More than one engr named ' +
        :qc_engr + ' works for ' + :h_pragy_code +
        '. Please call your DBA.';
sleep 3 ;
qc_engr := '';
qc_div := '';
qc_phon := '';
endif;
elseif rcount = 1 then
/*          found qc engineer by last name */
enter := select h_qc_num = p.id_num
        from person p
        where p.mfg_code = :h_pragy_code and
        :qc_engr = p.last_name;

else
/*          duplicate last names in person table */
message 'More than one engineer named ' +
        :qc_engr + ' works for ' + :h_pragy_code + '. Please enter '
        + 'the initials.';
sleep 3 ;
qc_engr := '';
qc_div := '';
qc_phon := '';
endif;
if rcount = 1 then
/*          have found qc engineer in person table - h_qc_num has
        person's id number
        see if in qc engineer table */
enter := select h_qc_num = pa_qc_engr.id_num
        from pa_qc_engr
        where
        pa_qc_engr.id_num = :h_qc_num ;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*          have input invalid name for qc engineer */
message :qc_engr + ' is not a valid quality control engineer' ;
sleep 3 ;
h_qc_num := '';
h_qc_name := '';
qc_engr := '';
qc_div := '';
qc_phon := '';
else
/*          have valid qc engineer */
/*          retrieve information from person table by id_num */
enter := select h_qc_name = person.first_name +
        , , +
        person.middl_init +
        , , +
        person.last_name,
        qc_div = person.asnd_orgn,
        qc_phon = '(' +
        left(person.phone_num,3) +

```

```

        ),' +
        left(shift(person.phone_num, -3), 3) +
        ',_' +
        shift(person.phone_num, -6)
    from person
    where
        person.id_num = :h_qc_num ;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
/*      could not retrieve qc engineer -
        database error      */
        message 'Error retrieving quality control engineer ' +
            'information - contact DBA' ;
        sleep 3 ;
        h_qc_num := '' ;
        h_qc_name := '' ;
        qc_engr := '' ;
        qc_div := '' ;
        qc_phon := '' ;
    else
/*      qc_engr := :h_qc_name ;
        blank out phone if not available      */
        if qc_phon = '()-' then
            qc_phon := '' ;
        endif
    endif ;
endif ;
endif ;
endif ;
/*      now modify database      */
/*      ***** begin update transaction *****      */
begin transaction ;
err_unit := 0 ;
chg_unit := 0 ;
chg_comp := 0 ;
chg_mfg_inst := 0 ;
/*      part in unit table - update unit table      */
update unit
    set
        da_data_engr = :h_data_num,
        da_qa_engr = :h_qa_num,
        da_rel_engr = :h_rel_num,
        pa_prod_engr = :h_prod_num,
        pa_qc_engr = :h_qc_num,
        dsn_mfg_code = :h_desn_code,
        prd_mfg_code = :h_pragy_code,
        da_desn_engr = :h_desn_num
    where unit_da_suf = :part_suf
        and
            unit_stem = :part_num ;
/*      check success of update      */
inquire_ingres(rcount = rowcount) ;
if rcount = 0 then
/*      unsuccessful update - failure or no change in data      */
/*      do not set err_unit flag?      */
else
/*      unit table has been modified      */
        chg_unit := 1 ;
endif ;
/*      no error in updating part in unit table -

```

```

        now do component table */
/*      assume new part is a component, not a weapon */
/*      update component table */
update component
set
        com_nom_num = :tnomn,
        com_nom_type = :tnomt
    where
        unit_da_suf = :part_suf
        and
        unit_stem   = :part_num ;
/*      check success of update */
inquire_ingres (rcount = rowcount) ;
if rcount != 0 then
/*      successful update to unit component table */
    chg_comp := 1 ;
else
/*      unsuccessful update - error or no data change */
endif ;
/*      now insert suppliers and weapon programs into mfg_inst table */
/*      for new or old parts */
/*      unloadtable sup_wepn
{
/*      check for blank lines -
    unloadtable is always entered, even if no data
    was entered in the table - counts at least one row
*/
    if :sup_wepn.supplier != '' or :sup_wepn.wepn_pgm != '' then
/*      set supplier code */
/*      supplier may be input as code or synonym */
/*      check for code or actual company name */
        h_supp_code := '' ;
        h_supp_name := :sup_wepn.supplier ;
        enter        := select h_supp_name = company.company_name,
                                h_supp_code = company.mfg_code
                                from    company
                                where   company.mfg_code = :h_supp_name
                                or
                                company.company_name =
                                    :h_supp_name ;
        sup_wepn.supplier := :h_supp_name ;
        inquire_ingres (rcount = rowcount) ;
        if rcount = 0 then
/*      not a valid code - try synonym table */
            enter := select h_supp_code = company_syn.mfg_code
                            from    company_syn
                            where   :h_supp_name = company_syn.company_name ;
            inquire_ingres (rcount = rowcount) ;
            if rcount = 0 then
/*      not a valid code or synonym - error */
                message :sup_wepn.supplier +
                    ' is invalid supplier code or name' ;
                sleep 3 ;
            else
/*      replace the synonym company name with the complete company name */
                enter.sup_wepn := select
                                    supplier = company.company_name,
                                    wepn_pgm = :sup_wepn.wepn_pgm
                                    from    company
                                    where :h_supp_code =

```

company.mfg_code ;

```
endif ;
endif ;
if h_supp_code != '' then
/*      have valid supplier      */
/*      get weapon program      */
/*      find location of '-'    */
enter := select
        loc_dash =
            locate(:sup_wepn.wepn_pgm, '-')
        from component ;
/*      get pgm_code and mod_num - should work whether
        or not there is a '-' */
enter := select
        h_pgm_code =
            left(:sup_wepn.wepn_pgm, :loc_dash - 1),
        h_mod_num =
            shift(:sup_wepn.wepn_pgm, -:loc_dash)
        from component ;
/* if valid format for weapon, insert into mfg_inst table */
/* if h_pgm_code != '' and h_mod_num != '' then
        see if mfg_inst already exists */
enter := select h_dummy = mfg_inst.mfg_code
        from mfg_inst
        where
            mfg_inst.unit_stem = :part_num
            and
            mfg_inst.unit_da_suf = :part_suf
            and
            mfg_inst.mfg_code = :h_supp_code
            and
            mfg_inst.wpn_pgm_code = :h_pgm_code
            and
            mfg_inst.wpn_mod_num = :h_mod_num ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      no corresponding entry in mfg_inst table */
/*      construct next mfg instance number -
        one higher than previous maximum */
enter := select
        h_inst_num =
            vchar(max(
                int4(mfg_inst.mfg_inst_num)) + 1)
        from mfg_inst ;
insert into mfg_inst
        (unit_da_suf,
         unit_stem,
         mfg_code,
         wpn_mod_num,
         wpn_pgm_code,
         mfg_inst_num)
        values
        (:part_suf,
         :part_num,
         :h_supp_code,
         :h_mod_num,
         :h_pgm_code,
         :h_inst_num) ;
inquire_ingres (rcount = rowcount) ;
if rcount != 0 then
/*      successful insertion into mfg_inst */
```

```

        chg_mfg_inst := 1 ;
        message 'Supplier ' +
            :sup_wepn.supplier +
            ' and weapon program ' +
            :sup_wepn.wepn_pgm +
            ' inserted into database' ;
        sleep 3 ;
    else
        unsuccessful insertion into mfg_inst */
        message 'Error inserting supplier ' +
            :sup_wepn.supplier +
            ' and weapon program ' +
            :sup_wepn.wepn_pgm +
            ' into database' ;
        sleep 3 ;
        abort transaction */
        abort ;
    endif ;
endif ;
else
    message :sup_wepn.wepn_pgm + ' is an' +
        ' invalid weapon program identifier' ;
    sleep 3 ;
endif ;
endif ;
endif ;
} ; /* end of unloadtable sup_wepn */
/*
    have an old part - see if database was modified */
/*
    if chg_unit = 1 or chg_comp = 1 or chg_mfg_inst = 1 then
        successful update to unit and/or component
        and/or mfg_inst table */
        message 'Information for part ' +
            :part_num + '-' + :part_suf +
            ' has been revised in database' ;
        sleep 3 ;
    else
        unsuccessful update */
        message 'Data unchanged for component ' + :part_num +
            '-' + :part_suf ;
        sleep 3 ;
    endif ;
/*
    ***** end update transaction ***** */
    end transaction ;
endif ;
else
/*
    couldn't find part in database */
    message 'Part not found in database - no changes made' ;
    sleep 3 ;
endif ;
resume field nomen ;
}
'ADD' = {
/*
    Add a new part to database
    user inputs either nomenclature or part number or both -
    do not check cursor - may be in data fields
*/
    newflg := 1 ;
    chk_part_num := 1 ;
    tnomt := '' ;

```

```

tnomn := '' ;
if nomen != '' then
/*          nomenclature has been input */
/* ***** Assumes that nomenclature is in form of component nomenclature */
/*          split the nomenclature into type and suffix */
/*          get first two characters of nomenclature */
enter := select tnomt = left(:nomen, 2)
          from component ;
/*          check validity of nomenclature type */
enter := select h_dummy = com_nom_type.type
          from   com_nom_type
          where com_nom_type.type = :tnomt ;
inquire_ingres(rcount = rowcount) ;
if rcount = 0 then
    message :tnomt + 'is not a valid nomenclature type' +
            ' - blank nomenclature used' ;
    sleep 3 ;
    tnomt := '' ;
    nomen := '' ;
else
/*          have a nomenclature with a valid type */
/*          get nomenclature number */
enter := select tnomn = shift(:nomen, -2)
          from component ;
/*          see if part exists in component */
enter := select h_dummy = component.unit_stem
          from component
          where
            component.com_nom_type = :tnomt
            and
            component.com_nom_num = :tnomn ;
/*          see if found part number */
inquire_ingres(rcount = rowcount) ;
if rcount != 0 then
/*          found part under nomenclature -
            do not use entered part number, if any */
    message 'Part with nomenclature ' + :nomen +
            ' already exists in database - no changes made';
    sleep 3;
    newflg := 0 ;
    chk_part_num := 0 ;
    endif;
endif ;
endif;
if chk_part_num = 1 then
/*          did not find part under nomenclature - look at part number */
/*          if part_num != '' then
/*          user has input part number */
/*          now find nomenclature from component table */
h_nomt := '' ;
h_nomn := '' ;
enter := select h_nomt = component.com_nom_type,
                h_nomn = component.com_nom_num
                from component
                where
                component.unit_stem = :part_num
                and
                component.unit_da_suf = :part_suf ;
/*          see if found part in component table */
inquire_ingres (rcount = rowcount) ;

```



```

        if rcount = 0 then
/*          not find part in component table - try weapon table */
            enter := select twpnpgm = weapon.wpn_pgm_code,
                           twpncfg = weapon.wpn_test_cfg,
                           twpnmod = weapon.wpn_mod_num
                           from weapon
                           where
                           weapon.unit_stem = :part_num
                           and
                           weapon.unit_da_suf = :part_suf ;
/*          see if found part in weapon table */
            inquire_ingres (rcount = rowcount) ;
            if rcount != 0 then
/*          found part in weapon table */
                newflg := 0 ;
                message 'Part with part number ' + :part_num +
                    '-' + :part_suf +
                    ' already exists in database - no changes made';
                sleep 3;
            endif;
        else
/*          found part in component table */
            newflg := 0 ;
            message 'Part with part number ' + :part_num +
                '-' + :part_suf +
                ' already exists in database - no changes made';
            sleep 3;
        endif ;
    endif ;
endif ;
if newflg = 1 then
/*          part not in database */
/*          must have non-blank part number to enter new part */
    if part_num != '' then
/*          check database integrity for this part */
/*          check existence in unit table */
        enter := select h_dummy = unit.unit_stem
                   from unit
                   where unit.unit_stem = :part_num
                   and
                   unit.unit_da_suf = :part_suf ;
        inquire_ingres (rcount = rowcount) ;
        err_unit := 0 ;
        if rcount != 0 then
/*          part was in unit table -
            but was not in component or weapon table - error */
            message 'Part in unit table but not component table' +
                ' - database error' ;
            sleep 3 ;
            err_unit := 1 ;
        endif ;
/*          if had database integrity error - do not process update */
        if err_unit = 0 then
/*          set production agency code */
            chg_pragy := 0 ;
            if prod_agency != :h_pragy_name then
                chg_pragy := 1 ;
                h_pragy_code := '' ;
                h_pragy_name := '' ;
/*          production agency may be input as code or synonym */
            endif ;
        endif ;
    endif ;
endif ;

```

```

/*      check for code      */
enter := select h_pragy_name = company.company_name
           from   company
           where  :prod_agency = company.mfg_code
                  or
                  :prod_agency = company.company_name ;
inquire_ingres (rcount = rowcount) ;
if rcount != 0 then
/*      valid code was entered      */
    h_pragy_code := :prod_agency ;
    prod_agency := :h_pragy_name ;
else
/*      not a valid code - try synonym table */
    enter := select h_pragy_code = company_syn.mfg_code
                   from   company_syn
                   where  :prod_agency = company_syn.company_name ;
    inquire_ingres (rcount = rowcount) ;
    if rcount = 0 then
/*      not a valid code or synonym - error */
        message prod_agency +
            ' is invalid production agency code or name' ;
        sleep 3 ;
    else
/*      replace the synonym company name with the complete company name */
        enter := select prod_agency = company.company_name
                     from   company
                     where  :h_pragy_code = company.mfg_code ;
        h_pragy_name := :prod_agency ;
    endif ;
endif ;
endif ;
/* for da employees, convert entered last names to da employee numbers */
/*      design engineer      */
if :designngr != :h_desn_name or
:h_desn_div != :desn_div then
    h_desn_num := '' ;
    h_desn_code := '' ;
    h_desn_name := '' ;
    h_desn_div := '' ;
/*      see how many design engineers with same last name */
enter := select rcount = count (*) from person p
           where p.last_name = :designngr and p.asnd_orgn = :desn_div ;
if rcount = 0 then
/*      not under last name - try initials + last name */
    enter := select rcount = count(*) from person p
               where p.asnd_orgn = :desn_div
               and :designngr = left(p.first_name, 1) + " " + p.middl_init
                   + " " + p.last_name ;
    if rcount = 0 then
/*      name for design engineer is not a valid person */
        message 'There is no person named ' + :designngr +
            ' who works for ' + :desn_div ;
        sleep 3 ;
        designngr := '' ;
        desn_div := '' ;
        desn_phon := '' ;
    elseif rcount = 1 then
/*      found design engineer by initials and last name */
        enter := select h_desn_num = p.id_num from person p
                     where p.asnd_orgn = :desn_div

```

```

        and :designngr = left(p.first_name, 1) + " " +
            p.middl_init + " " + p.last_name;
    else
/*      found more than one design engineer by initials and last name */
        message 'More than one engr named ' +
            :designngr + ' works for ' + :desn_div + '. Please call '
            + 'your DBA.';
        sleep 3 ;
        designngr := '' ;
        desn_div   := '' ;
        desn_phon := '' ;
    endif;
elseif rcount = 1 then
/*      found design engineer by last name */
        enter := select h_desn_num = p.id_num
            from person p
            where p.asnd_orgn = :desn_div and
                :designngr = p.last_name;
else
/*      duplicate last names in person table */
        message 'More than one engineer named ' +
            :designngr + ' works for ' + desn_div + '. Please enter '
            + 'the initials.';
        sleep 3 ;
        designngr := '' ;
        desn_div   := '' ;
        desn_phon := '' ;
endif;
if rcount = 1 then
/*      have found design engineer in person table - h_desn_num has
        person's id number
        see if in design engineer table */
        enter := select h_desn_num = da_empl.id_num
            from da_empl, da_desn_engr
            where
                da_desn_engr.empl_num = da_empl.empl_num and
                da_empl.id_num = :h_desn_num ;
        inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*      have input invalid name for design engineer */
        message :designngr + ' is not a valid design engineer' ;
        sleep 3 ;
        h_desn_num := '' ;
        h_desn_code := '' ;
        h_desn_name := '' ;
        designngr := '' ;
        desn_div   := '' ;
        desn_phon := '' ;
else
/*      have valid design engineer */
/*      retrieve information from person table by id_num */
        enter := select h_desn_code = person.mfg_code,
            h_desn_name = person.first_name +
                ' ' +
                person.middl_init +
                ' ' +
                person.last_name,
            desn_phon = '(' +
                left(person.phone_num,3) +
                ')' +

```

```

                                left(shift(person.phone_num, -3), 3) +
                                '-' +
                                shift(person.phone_num, -6)
                                from person
                                where
                                    person.id_num = :h_desn_num ;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*
    could not retrieve design engineer -
    database error */
    message 'Error retrieving design engineer information - ' +
        'contact DBA' ;
    sleep 3 ;
    h_desn_num := '' ;
    h_desn_code := '' ;
    h_desn_name := '' ;
    designngr := '' ;
    desn_div := '' ;
    desn_phon := '' ;
else
    designngr := :h_desn_name ;
    h_desn_div := :desn_div ;
/*
    blank out phone if not available */
    if desn_phon = '()-' then
        desn_phon := '' ;
    endif
endif ;
endif ;
endif ;
endif ;
/*
    data engineer */
if data_engr != :h_data_name or
:h_data_div != :data_div then
    h_data_num := '' ;
    h_data_name := '' ;
    h_data_div := '' ;
enter := select h_data_num = da_empl.empl_num,
                h_data_name = person.first_name +
                ,
                person.middl_init +
                ,
                person.last_name,
                data_phon = '(' +
                left(person.phone_num,3) +
                ') ' +
                left(shift(person.phone_num, -3), 3) +
                '-' +
                shift(person.phone_num, -6)
                from da_empl, person
                where
                    person.last_name = :data_engr
                    and
                    person.asnd_orgn = :data_div
                    and
                    da_empl.id_num = person.id_num ;
/*
    see if found data engineer in person and da_empl table */
inquire_ingres(rcount = rowcount) ;
if rcount = 0 then
/*
    name for data engineer is not a valid person */
    message 'There is no person named ' + :data_engr +

```

```

        ' who works for ' + :data_div ;
sleep 3 ;
data_engr := '' ;
data_div := '' ;
data_phon := '' ;
else
/*
    have found data engineer in person table -
    see if in data engineer table */
enter := select h_data_num = da_data_engr.empl_num
            from da_data_engr
            where
                da_data_engr.empl_num = :h_data_num ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*
    have input invalid name for data engineer */
    message :data_engr + ' is not a valid data engineer' ;
    sleep 3 ;
    h_data_num := '' ;
    h_data_name := '' ;
    data_engr := '' ;
    data_div := '' ;
    data_phon := '' ;
else
/*
    have valid data engineer */
    data_engr := :h_data_name ;
    h_data_div := :data_div ;
/*
    blank out phone if not available */
    if data_phon = '()-' then
        data_phon := '' ;
    endif ;
endif ;
endif ;
endif ;
/*
    qa engineer */
if qa_engr != :h_qa_name or
:h_qa_div != :qa_div then
    h_qa_num := '' ;
    h_qa_name := '' ;
    h_qa_div := '' ;
enter := select h_qa_num = da_empl.empl_num,
                h_qa_name = person.first_name +
                    , , +
                    person.middl_init +
                    , , +
                    person.last_name,
                qa_phon = '(' +
                    left(person.phone_num,3) +
                    ')' +
                    left(shift(person.phone_num, -3), 3) +
                    '-' +
                    shift(person.phone_num, -6)
                from da_empl, person
            where
                person.last_name = :qa_engr
                and
                person.asnd_orgn = :qa_div
                and
                da_empl.id_num = person.id_num ;
/*
    see if found qa engineer in person and da_empl table */
inquire_ingres (rcount = rowcount) ;

```

```

if rcount = 0 then
/*      name for qa engineer is not a valid person */
message 'There is no person named ' + :qa_engr +
      ' who works for ' + :qa_div ;
sleep 3 ;
qa_engr := '' ;
qa_div := '' ;
qa_phon := '' ;
else
/*      have found qa engineer in person table -
      see if in qa engineer table */
enter := select h_qa_num = da_qa_engr.empl_num
      from da_qa_engr
      where
        da_qa_engr.empl_num = :h_qa_num ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      have input invalid name for qa engineer */
message :qa_engr + ' is not a valid qa engineer' ;
sleep 3 ;
h_qa_num := '' ;
h_qa_name := '' ;
qa_engr := '' ;
qa_div := '' ;
qa_phon := '' ;
else
/*      have valid qa engineer */
qa_engr := :h_qa_name ;
h_qa_div := :qa_div ;
/*      blank out phone if not available */
if qa_phon = '()-' then
  qa_phon := '' ;
endif ;
endif ;
endif ;
endif ;
/*      reliability engineer */
if rel_engr != :h_rel_name or
:h_rel_div != :rel_div then
h_rel_num := '' ;
h_rel_name := '' ;
h_rel_div := '' ;
enter := select h_rel_num = da_empl.empl_num,
      h_rel_name = person.first_name +
        , , +
        person.middl_init +
        , , +
        person.last_name,
      rel_phon = '(' +
        left(person.phone_num,3) +
        , , +
        left(shift(person.phone_num, -3), 3) +
        , , +
        shift(person.phone_num, -6)
      from da_empl, person
      where
        person.last_name = :rel_engr
        and
        person.asnd_orgn = :rel_div
        and

```

```

        da_empl.id_num = person.id_num ;
/*      see if found reliability engineer in person and da_empl table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      name for reliability engineer is not a valid person */
message 'There is no person named ' + :rel_engr +
        ' who works for ' + :rel_div ;
sleep 3 ;
rel_engr := '' ;
rel_div := '' ;
rel_phon := '' ;
else
/*      have found reliability engineer in person table -
        see if in reliability engineer table */
enter := select h_rel_num = da_rel_engr.empl_num
        from da_rel_engr
        where
            da_rel_engr.empl_num = :h_rel_num ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      have input invalid name for reliability engineer */
message :rel_engr + ' is not a valid reliability engineer' ;
sleep 3 ;
h_rel_num := '' ;
h_rel_name := '' ;
rel_engr := '' ;
rel_div := '' ;
rel_phon := '' ;
else
/*      have valid reliability engineer */
rel_engr := :h_rel_name ;
h_rel_div := rel_div ;
/*      blank out phone if not available */
if rel_phon = '()-' then
    rel_phon := '' ;
endif ;
endif ;
endif ;
endif ;
/*      PA production engineer */
if prod_engr != :h_prod_name or
(chg_pragy = 1 and prod_engr != '') then
    h_prod_num := '' ;
    h_prod_name := '' ;
/*      see how many production engineers with same last name */
enter := select rcount = count (*) from person p
        where p.last_name = :prod_engr and p.mfg_code = :h_pragy_code ;
if rcount = 0 then
/*      not under last name - try initials + last name */
enter := select rcount = count(*) from person p
        where p.mfg_code = :h_pragy_code
            and :prod_engr = left(p.first_name, 1) + " " + p.middl_init
                + " " + p.last_name ;
if rcount = 0 then
/*      name for production engineer is not a valid person */
message 'There is no person named ' + :prod_engr +
        ' who works for ' + :h_pragy_code ;
sleep 3 ;
prod_engr := '' ;
prod_div := '' ;

```

```

        prod_phon := '';
    elseif rcount = 1 then
        found production engineer by initials and last name */
        enter := select h_prod_num = p.id_num from person p
            where p.mfg_code = :h_pragy_code
            and :prod_engr = left(p.first_name, 1) + ' ' +
                p.middl_init + ' ' + p.last_name;
    else
        found more than one production engineer by initials and last name */
        message 'More than one engr named ' +
            :prod_engr + ' works for ' + :h_pragy_code +
            '. Please call your DBA.';
        sleep 3;
        prod_engr := '';
        prod_div := '';
        prod_phon := '';
    endif;
elseif rcount = 1 then
    found production engineer by last name */
    enter := select h_prod_num = p.id_num
        from person p
        where p.mfg_code = :h_pragy_code and
            :prod_engr = p.last_name;
else
    duplicate last names in person table */
    message 'More than one engineer named ' +
        :prod_engr + ' works for ' + :h_pragy_code + '. Please enter '
        + 'the initials.';
    sleep 3;
    prod_engr := '';
    prod_div := '';
    prod_phon := '';
endif;
if rcount = 1 then
    have found production engineer in person table - h_prod_num has
    person's id number
    see if in production engineer table */
    enter := select h_prod_num = pa_prod_engr.id_num
        from pa_prod_engr
        where
            pa_prod_engr.id_num = :h_prod_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        have input invalid name for production engineer */
        message :prod_engr + ' is not a valid production engineer';
        sleep 3;
        h_prod_num := '';
        h_prod_name := '';
        prod_engr := '';
        prod_div := '';
        prod_phon := '';
    else
        have valid production engineer */
        retrieve information from person table by id_num */
        enter := select h_prod_name = person.first_name +
            , , +
            person.middl_init +
            , , +
            person.last_name,
            prod_div = person.asnd_orgn,

```



```

        prod_phon = '(' +
                    left(person.phone_num,3) +
                    ')', +
                    left(shift(person.phone_num, -3), 3) +
                    ',' +
                    shift(person.phone_num, -6)
        from person
        where
            person.id_num = :h_prod_num ;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
/*
            could not retrieve production engineer -
            database error */
            message 'Error retrieving production engineer ' +
                'information - contact DBA' ;
            sleep 3 ;
            h_prod_num := '' ;
            h_prod_name := '' ;
            prod_engr := '' ;
            prod_div := '' ;
            prod_phon := '' ;
        else
            prod_engr := :h_prod_name ;
/*
            blank out phone if not available */
            if prod_phon = '()-' then
                prod_phon := '' ;
            endif
        endif ;
    endif ;
endif ;
endif ;
endif ;
/*
        PA qc engineer */
        if qc_engr != :h_qc_name or
        (chg_pragy = 1 and qc_engr != '') then
            h_qc_num := '' ;
            h_qc_name := '' ;
/*
            see how many qc engineers with same last name */
            enter := select rcount = count (*) from person p
                where p.last_name = :qc_engr and p.mfg_code = :h_pragy_code ;
            if rcount = 0 then
/*
                not under last name - try initials + last name */
                enter := select rcount = count(*) from person p
                    where p.mfg_code = :h_pragy_code
                    and :qc_engr = left(p.first_name, 1) + " " + p.middl_init
                        + " " + p.last_name;
                if rcount = 0 then
/*
                    name for qc engineer is not a valid person */
                    message 'There is no person named ' + :qc_engr +
                        ' who works for ' + :h_pragy_code ;
                    sleep 3 ;
                    qc_engr := '' ;
                    qc_div := '' ;
                    qc_phon := '' ;
                elseif rcount = 1 then
/*
                    found qc engineer by initials and last name */
                    enter := select h_qc_num = p.id_num from person p
                        where p.mfg_code = :h_pragy_code
                        and :qc_engr = left(p.first_name, 1) + " " +
                            p.middl_init + " " + p.last_name;
                else

```

```

/* found more than one qc engineer by initials and last name */
    message 'More than one engr named ' +
        :qc_engr + ' works for ' + :h_pragy_code +
        '. Please call your DBA.';
    sleep 3 ;
    qc_engr := '';
    qc_div := '';
    qc_phon := '';
endif;
elseif rcount = 1 then
/* found qc engineer by last name */
    enter := select h_qc_num = p.id_num
        from person p
        where p.mfg_code = :h_pragy_code and
            :qc_engr = p.last_name;

else
/* duplicate last names in person table */
    message 'More than one engineer named ' +
        :qc_engr + ' works for ' + :h_pragy_code + '. Please enter '
        + 'the initials.';
    sleep 3 ;
    qc_engr := '';
    qc_div := '';
    qc_phon := '';
endif;
if rcount = 1 then
/* have found qc engineer in person table - h_qc_num has
    person's id number
    see if in qc engineer table */
    enter := select h_qc_num = pa_qc_engr.id_num
        from pa_qc_engr
        where
            pa_qc_engr.id_num = :h_qc_num ;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
/* have input invalid name for qc engineer */
        message :qc_engr + ' is not a valid quality control engineer' ;
        sleep 3 ;
        h_qc_num := '';
        h_qc_name := '';
        qc_engr := '';
        qc_div := '';
        qc_phon := '';
    else
/* have valid qc engineer */
/* retrieve information from person table by id_num */
        enter := select h_qc_name = person.first_name +
            ', ' +
            person.middl_init +
            ', ' +
            person.last_name,
            qc_div = person.asnd_orgn,
            qc_phon = '(' +
                left(person.phone_num,3) +
                ')', +
                left(shift(person.phone_num, -3), 3) +
                ', ' +
                shift(person.phone_num, -6)
            from person
        where

```

```

                person.id_num = :h_qc_num ;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
/*          could not retrieve qc engineer -
              database error          */
    message 'Error retrieving quality control engineer ' +
            'information - contact DBA' ;
    sleep 3 ;
    h_qc_num := '' ;
    h_qc_name := '' ;
    qc_engr := '' ;
    qc_div := '' ;
    qc_phon := '' ;
else
    qc_engr := :h_qc_name ;
/*          blank out phone if not available      */
    if qc_phon = '()-' then
        qc_phon := '' ;
    endif
endif ;
endif ;
endif ;
endif ;
/*          now insert new part into database      */
/*          ***** begin update transaction ***** */
begin transaction ;
err_unit := 0 ;
/*          part not in unit table - add to unit table      */
insert into unit (unit_da_suf,
                  unit_stem,
                  da_data_engr,
                  da_qa_engr,
                  da_rel_engr,
                  pa_prod_engr,
                  pa_qc_engr,
                  dsn_mfg_code,
                  prd_mfg_code,
                  da_desn_engr)
    values (:part_suf,
            :part_num,
            :h_data_num,
            :h_qa_num,
            :h_rel_num,
            :h_prod_num,
            :h_qc_num,
            :h_desn_code,
            :h_pragy_code,
            :h_desn_num) ;
/*          check success of insertion      */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*          unsuccessful insertion      */
    message 'Error in adding new component ' +
            :part_num +
            ',' +
            :part_suf +
            ' - addition failed' ;
    sleep 3 ;
    err_unit := 1 ;
endif ;

```

```

if err_unit = 0 then
/*      no error in putting part into unit table -
      now do component table      */
/*      assume new part is a component, not a weapon      */
/*      add to component table      */
insert into component (unit_da_suf,
                        unit_stem,
                        com_nom_num,
                        com_nom_type)
values                  (:part_suf,
                        :part_num,
                        :tnomn,
                        :tnomt) ;
/*      check success of insertion      */
/*      inquire_ingres (rcount = rowcount) ;
if rcount != 0 then
/*      successful insertion      */
message 'New component ' + :part_num + '-' + :part_suf +
        ' has been entered into database' ;
sleep 3;
else
/*      unsuccessful insertion      */
message 'Error in adding new component ' +
        :part_num +
        '-' +
        :part_suf +
        ' - addition failed' ;
sleep 3;
/*      abort transaction      */
/*      abort ;
endif ;
/*      now insert suppliers and weapon programs into mfg_inst table      */
/*      for new or old parts      */
/*      unloadtable sup_wepn
{
/*      check for blank lines -
      unloadtable is always entered, even if no data
      was entered in the table - counts at least one row
*/
      if :sup_wepn.supplier != '' or :sup_wepn.wepn_pgm != '' then
/*      set supplier code      */
/*      supplier may be input as code or synonym      */
/*      check for code or actual company name      */
h_supp_code := '' ;
h_supp_name := :sup_wepn.supplier ;
enter       := select h_supp_name = company.company_name,
                        h_supp_code = company.mfg_code
                        from   company
                        where  company.mfg_code = :h_supp_name
                        or
                        company.company_name =
                                :h_supp_name ;
sup_wepn.supplier := :h_supp_name ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      not a valid code - try synonym table      */
enter := select h_supp_code = company_syn.mfg_code
            from   company_syn
            where  :h_supp_name = company_syn.company_name ;
inquire_ingres (rcount = rowcount) ;

```

```

        if rcount = 0 then
            not a valid code or synonym - error */
            message :sup_wepn.supplier +
                ' is invalid supplier code or name' ;
            sleep 3 ;
        else
            /* replace the synonym company name with the complete company name */
            enter.sup_wepn := select
                supplier = company.company_name,
                wepn_pgm = :sup_wepn.wepn_pgm
            from company
            where :h_supp_code =
                company.mfg_code ;

        endif ;
    endif ;
    if h_supp_code != '' then
        /* have valid supplier */
        /* get weapon program */
        /* find location of '-' */
        enter := select
            loc_dash =
                locate(:sup_wepn.wepn_pgm, '-')
            from component ;
        /* get pgm_code and mod_num - should work whether
            or not there is a '-' */
        enter := select
            h_pgm_code =
                left(:sup_wepn.wepn_pgm, :loc_dash - 1),
            h_mod_num =
                shift(:sup_wepn.wepn_pgm, -:loc_dash)
            from component ;
        /* if valid format for weapon, insert into mfg_inst table */
        /* if h_pgm_code != '' and h_mod_num != '' then
            see if mfg_inst already exists */
        enter := select h_dummy = mfg_inst.mfg_code
            from mfg_inst
            where
                mfg_inst.unit_stem = :part_num
            and
                mfg_inst.unit_da_suf = :part_suf
            and
                mfg_inst.mfg_code = :h_supp_code
            and
                mfg_inst.wpn_pgm_code = :h_pgm_code
            and
                mfg_inst.wpn_mod_num = :h_mod_num ;
        inquire_ingres (rcount = rowcount) ;
        if rcount = 0 then
            /* no corresponding entry in mfg_inst table */
            /* construct next mfg instance number -
                one higher than previous maximum */
            enter := select
                h_inst_num =
                    vchar(max(
                        int4(mfg_inst.mfg_inst_num)) + 1)
                from mfg_inst ;
            insert into mfg_inst
                (unit_da_suf,
                unit_stem,
                mfg_code,

```

```

                                wpn_mod_num,
                                wpn_pgm_code,
                                mfg_inst_num )
                                values
                                (:part_suf,
                                 :part_num,
                                 :h_supp_code,
                                 :h_mod_num,
                                 :h_pgm_code,
                                 :h_inst_num) ;
                                inquire_ingres (rcount = rowcount) ;
                                if rcount != 0 then
/*                                successful insertion into mfg_inst */
                                chg_mfg_inst := 1 ;
                                message 'Supplier ' +
                                    :sup_wepn.supplier +
                                    ' and weapon program ' +
                                    :sup_wepn.wepn_pgm +
                                    ' inserted into database' ;
                                sleep 3 ;
                                else
/*                                unsuccessful insertion into mfg_inst */
                                message 'Error inserting supplier ' +
                                    :sup_wepn.supplier +
                                    ' and weapon program ' +
                                    :sup_wepn.wepn_pgm +
                                    ' into database' ;
                                sleep 3 ;
/*                                abort transaction */
                                abort ;
                                endif ;
                                endif ;
                                else
                                message :sup_wepn.wepn_pgm + ' is an' +
                                    ' invalid weapon program identifier' ;
                                sleep 3 ;
                                endif ;
                                endif ;
                                } ; /* end of unloadtable sup_wepn */
                                endif ;
/*                                ***** end update transaction ***** */
                                end transaction ;
                                endif ;
                                else
/*                                No valid part number exists */
                                message 'Valid part number is required to add part to database';
                                sleep 3;
                                endif;
                                endif;
                                resume field nomen ;
                                }
                                'CLEAR' = {
/*                                clear screen of all data, put cursor at nomenclature,
                                    awaiting input */
                                clear field all ;
/*                                clear all hidden fields so in synch with screen */
                                h_data_num := '';
                                h_qa_num := '';
                                h_rei_num := '';
                                h_desn_num := '';

```

```

        h_desn_code := '';
        h_pragy_code := '';
        h_pragy_name := '';
        h_prod_num := '';
        h_qc_num := '';
        h_data_name := '';
        h_qa_name := '';
        h_rel_name := '';
        h_desn_name := '';
        h_data_div := '';
        h_qa_div := '';
        h_rel_div := '';
        h_desn_div := '';
        h_prod_name := '';
        h_qc_name := '';
        resume field nomen ;
    }

```

```

'HELP', key frskey1 =
{
    help_forms (subject = 'Enter a new part or modify part Info',
        file = 'db1:[dbaprdb.help]enter.help');
}
'XIT', key frskey3 =
{
    RETURN;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

/*

FRAME: INFO INFO
 DSL to fill INFO form
 Written by John Orman 6/15/88

*/

```
initialize (levl = integer2,
  num_part = integer2,
  nomflg = vchar(3),
  rcount = integer4,
  doc_count = integer4,
  doc_index = integer4,
  errflg = integer1,
  tmlrdt = vchar(12),
  trddt = vchar(12),
  tmfgnum = vchar(4),
  tnomen = vchar(12),
  tpartn = vchar(10),
  tparts = vchar(3),
  tnomt = vchar(3),
  tnomn = vchar(9),
  twpnpgm = vchar(4),
  twpncfg = vchar(7),
  twpnmod = vchar(2),
  FNAME = vchar(8),
  doc_table.h_cer_date = vchar(12),
  doc_table.h_dter_date = vchar(12),
  doc_table.h_matl_list = vchar(12),
  doc_table.h_rqrd_dt = vchar(12),
  doc_table.h_e_comp_dt = vchar(12)
) = {
  mode 'query';
  resume field nomen ;
}
```

'RETRIEVE' = {

/*

user inputs either nomenclature or part number -
 see which field has cursor

*/

```
inquire_forms field " " (FNAME = name);
errflg := 0 ;
if FNAME = 'nomen' then
```

/*

cursor is in nomenclature field */

/*

clear all old fields, restore entered nomenclature */

tnomen := :nomen ;

clear field all ;

nomen := :tnomen ;

if nomen != '' then

/*

nomenclature has been input */

/*

***** Assumes that nomenclature is in form of component nomenclature */

/*

split the nomenclature into type and suffix */

/*

get first two characters of nomenclature */

info := select tnomt = left(:nomen, 2)

from component ;

nomflg = 'NO' ;

info := select nomflg = 'YES'

from component

where :tnomt = '[A-Z]*' ;

if nomflg = 'YES' then

/*

have a nomenclature with a non-blank type */

/*

get nomenclature number */


```

        info := select tnomn = shift(:nomen, -2)
                from component ;
    else
        nomenclature is in part number format */
        tnomt := '' ;
        tnomn := :nomen ;
    endif ;
/*
        get the part number */
message 'Retrieving Information...';
info := select part_num = component.unit_stem,
                part_suf = component.unit_da_suf
                from component
                where
                    component.com_nom_type = :tnomt
                    and
                    component.com_nom_num = :tnomn ;
/*
        see if found part number */
if part_num = '' then
    message 'No part number corresponding to the ' +
        'entered nomenclature was found' ;
    errflg := 1 ;
    sleep 3 ;
endif ;
endif ;
elseif FNAME = 'part_num' or FNAME = 'part_suf' then
/*
        cursor is in part number or part suffix field */
/*
        clear all old fields, restore entered part number */
tpartn := :part_num ;
tparts := :part_suf ;
clear field all ;
part_num := :tpartn ;
part_suf := :tparts ;
if part_num != '' then
/*
        user has input part number */
/*
        now find nomenclature from component table */
tnomt := '' ;
tnomn := '' ;
message 'Retrieving Information...';
INFO := select tnomt = component.com_nom_type,
                tnomn = component.com_nom_num
                from component
                where
                    component.unit_stem = :part_num
                    and
                    component.unit_da_suf = :part_suf ;
/*
        see if found part in component table */
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*
        not find part in component table - try weapon table */
tnomt := '' ;
tnomn := '' ;
INFO := select twpnpgm = weapon.wpn_pgm_code,
                twpncfg = weapon.wpn_test_cfg,
                twpnmod = weapon.wpn_mod_num
                from weapon
                where
                    weapon.unit_stem = :part_num
                    and
                    weapon.unit_da_suf = :part_suf ;
/*
        see if found part in weapon table */

```

```

        inquire_ingres (rcount = rowcount) ;
        if rcount = 0 then
            errflg := 1 ;
            message 'No nomenclature corresponding to the ' +
                'entered part number was found' ;
            sleep 3;
        else
            /* found part in weapon table */
            /* now concatenate pgm_code, test_cfg, and mod_num
               to form nomenclature and put on form */
            nomen := twpnpgm + '/' + twpncfg + '-' + twpnmod ;
        endif;
    else
        /* found part in component table */
        /* now concatenate nomenclature type and number and
           put on form */
        nomen := tnomt + tnomn ;
    endif;
endif;
endif;
if nomen != '' or part_num != '' then
    if errflg = 0 then
        /* have a valid part to work with - load other form fields */
        /* ***** now find suppliers and weapon programs ***** */
        /* - not every unit has a mfg_inst */
        info.sup_wepn := select supplier = company.company_name,
                               wepn_pgm = mfg_inst.wpn_pgm_code +
                               '-' +
                               mfg_inst.wpn_mod_num
                           from company, mfg_inst
                           where
                               mfg_inst.unit_stem = :part_num
                               and
                               mfg_inst.unit_da_suf = :part_suf
                               and
                               company.mfg_code = mfg_inst.mfg_code ;
        /* ***** now find production agency ***** */
        info := select prod_agency = company.company_name
                       from company, unit
                       where
                           unit.unit_stem = :part_num
                           and
                           unit.unit_da_suf = :part_suf
                           and
                           company.mfg_code = unit.pr_d_mfg_code ;
        if prod_agency = '' then
            message 'No production agency for the ' +
                'entered part was found' ;
            sleep 3;
        endif;
        /* ***** now compute TMS required field ***** */
        /* if part is in mfg_inst and tms_activity, TMS is required */
        tmfgnum = '' ;
        info := select tmfgnum = mfg_inst.mfg_inst_num
                   from tms_activity, mfg_inst
                   where
                       mfg_inst.unit_stem = :part_num
                       and
                       mfg_inst.unit_da_suf = :part_suf
                       and

```

```

        mfg_inst.mfg_inst_num = tms_activity.mfg_inst_num ;
    if tmfnum != '' then
        tmsrequired := 'YES' ;
    else
        tmsrequired := 'NO' ;
    endif ;

/* ***** da design engineer ***** */
/* get design engineer from unit instead of component jlo 9/9/88 */
info := select designngr = person.first_name +
    , , +
    person.middl_init +
    , , +
    person.last_name,
    desn_div = person.asnd_orgn,
    desn_phon = '(' +
        left(person.phone_num,3) +
        ')', +
        left(shift(person.phone_num, -3), 3) +
        '-' +
        shift(person.phone_num, -6)
    from unit, person, da_empl
    where
        unit.unit_stem = :part_num
    and
        unit.unit_da_suf = :part_suf
    and
        unit.da_desn_engr = da_empl.empl_num
    and
        da_empl.id_num = person.id_num ;
/* blank out phone if not available */
if desn_phon = '()-' then
    desn_phon := '' ;
endif ;

/* ***** da data engineer ***** */
info := select data_engr = person.first_name +
    , , +
    person.middl_init +
    , , +
    person.last_name,
    data_div = person.asnd_orgn,
    data_phon = '(' +
        left(person.phone_num,3) +
        ')', +
        left(shift(person.phone_num, -3), 3) +
        '-' +
        shift(person.phone_num, -6)
    from unit, person, da_empl
    where
        unit.unit_stem = :part_num
    and
        unit.unit_da_suf = :part_suf
    and
        unit.da_data_engr = da_empl.empl_num
    and
        da_empl.id_num = person.id_num ;
/* blank out phone if not available */
if data_phon = '()-' then
    data_phon := '' ;
endif ;

/* ***** da quality assurance engineer ***** */

```

```

info := select qa_engr = person.first_name +
                , , +
                person.middl_init +
                , , +
                person.last_name,
qa_div = person.asnd_orgn,
qa_phon = '(' +
        left(person.phone_num,3) +
        ') ' +
        left(shift(person.phone_num, -3), 3) +
        '_ ' +
        shift(person.phone_num, -6)
from unit, person, da_empl
where
unit.unit_stem = :part_num
and
unit.unit_da_suf = :part_suf
and
unit.da_qa_engr = da_empl.empl_num
and
da_empl.id_num = person.id_num ;
/*      blank out phone if not available      */
if qa_phon = '()-' then
    qa_phon := '' ;
endif ;
/* ***** da reliability engineer ***** */
info := select rel_engr = person.first_name +
                , , +
                person.middl_init +
                , , +
                person.last_name,
rel_div = person.asnd_orgn,
rel_phon = '(' +
        left(person.phone_num,3) +
        ') ' +
        left(shift(person.phone_num, -3), 3) +
        '_ ' +
        shift(person.phone_num, -6)
from unit, person, da_empl
where
unit.unit_stem = :part_num
and
unit.unit_da_suf = :part_suf
and
unit.da_rel_engr = da_empl.empl_num
and
da_empl.id_num = person.id_num ;
/*      blank out phone if not available      */
if rel_phon = '()-' then
    rel_phon := '' ;
endif ;
/* ***** pa product engineer ***** */
info := select prod_engr = person.first_name +
                , , +
                person.middl_init +
                , , +
                person.last_name,
prod_div = person.asnd_orgn,
prod_phon = '(' +
        left(person.phone_num,3) +

```

```

        '),' +
        left(shift(person.phone_num, -3), 3) +
        '),' +
        shift(person.phone_num, -6)
from unit, person
where
unit.unit_stem = :part_num
and
unit.unit_da_suf = :part_suf
and
unit.pa_prod_engr = person.id_num ;
/*      blank out phone if not available      */
if prod_phon = '()-' then
    prod_phon := '';
endif ;
/* ***** pa quality control engineer ***** */
info := select qc_engr = person.first_name +
              ',' +
              person.middl_init +
              ',' +
              person.last_name,
qc_div = person.asnd_orgn,
qc_phon = '(' +
        left(person.phone_num,3) +
        '),' +
        left(shift(person.phone_num, -3), 3) +
        '),' +
        shift(person.phone_num, -6)
from unit, person
where
unit.unit_stem = :part_num
and
unit.unit_da_suf = :part_suf
and
unit.pa_qc_engr = person.id_num ;
/*      blank out phone if not available      */
if qc_phon = '()-' then
    qc_phon := '';
endif ;
/* ***** DOCUMENTS ***** */
/*      select all data documents -
      extract data from data_documnt and document tables */
/*      document type, document number, issue number,
      dates : cer, dter, ml, required, estimated completion */
info.doc_table := select
        dcmt_num = unit_doc.doc_type + ':' +
        unit_doc.doc_num,
        issue_num = unit_doc.doc_iss_num,
        h_cer_date = data_documnt.cer_dt,
        h_dter_date = data_documnt.dter_dt,
        h_matl_list = data_documnt.ml_dt,
        h_rqrd_dt = document.required_dt,
        h_e_comp_dt = document.est_comp_dt
from unit_doc, data_documnt, document
where
unit_doc.unit_stem = :part_num
and
unit_doc.unit_da_suf = :part_suf
and
unit_doc.doc_type = data_documnt.doc_type

```

```

        and
        unit_doc.doc_num = data_documnt.doc_num
        and
        unit_doc.doc_iss_num = data_documnt.doc_iss_num
        and
        unit_doc.doc_type = document.doc_type
        and
        unit_doc.doc_num = document.doc_num
        and
        unit_doc.doc_iss_num = document.doc_iss_num ;
/*      unload rows in form document table      */
/*      loop through rows of doc_table, filling in optional dates */
unloadtable doc_table
{
/*      convert dates to appropriate format - mo/da/yr */
doc_table.cer_date :=
        date_slash(date_in = :doc_table.h_cer_date) ;
doc_table.dter_date :=
        date_slash(date_in = :doc_table.h_dter_date) ;
doc_table.matl_list :=
        date_slash(date_in = :doc_table.h_matl_list) ;
doc_table.rqrd_dt :=
        date_slash(date_in = :doc_table.h_rqrd_dt) ;
doc_table.e_comp_dt :=
        date_slash(date_in = :doc_table.h_e_comp_dt) ;
/*      rough draft date - optional      */
trddt := '' ;
info := select trddt =
        doc_rough_dt.r_draft_dt
        from doc_rough_dt
        where
        doc_rough_dt.doc_type =
            left(:doc_table.dcmt_num,2)
        and
        doc_rough_dt.doc_num =
            shift(:doc_table.dcmt_num, -4)
        and
        doc_rough_dt.doc_iss_num =
            :doc_table.issue_num ;
doc_table.ruff_dft_dt = date_slash(date_in = :trddt) ;
/*      ml request date - optional      */
tmlrdt := '' ;
info := select tmlrdt = doc_rq_ml_dt.req_ml_dt
        from doc_rq_ml_dt
        where
        doc_rq_ml_dt.doc_type =
            left(:doc_table.dcmt_num,2)
        and
        doc_rq_ml_dt.doc_num =
            shift(:doc_table.dcmt_num, -4)
        and
        doc_rq_ml_dt.doc_iss_num =
            :doc_table.issue_num ;
doc_table.req_ml_date := date_slash(date_in = :tmlrdt) ;
} ; /*      end of unloadtable doc_table      */
endif ;
else
/*      neither nomenclature nor part number entered */
message 'Nomenclature or part number is a required field';
sleep 3;

```

```

endif ;
resume field nomen ;
}
'CLEAR' = {
/*      clear screen of all data, put cursor at nomenclature,
        awaiting input
        clear field all ;
        resume field nomen ;
*/
}
'HELP', key frskey1 =
{
    help_forms (subject = 'QA Information Screen',
        file = 'db1:[dbaprdb.help]info.help');
}
'XIT', key frskey3 =
{
    RETURN;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

/*

FRAME: NEWDOC
OSL to fill NEWDOC form
Written by John Orman 6/23/88

NEWDOC

*/

```

initialize (
    rcount = integer,
    h_cer_date = vchar(12),
    h_dter_date = vchar(12),
    h_matl_list = vchar(12),
    h_rqrd_dt = vchar(12),
    h_e_comp_dt = vchar(12),
    h_ruff_dft_dt = vchar(12),
    h_req_ml_date = vchar(12),
    t_cer_date = vchar(12),
    t_dter_date = vchar(12),
    t_matl_list = vchar(12),
    t_rqrd_dt = vchar(12),
    t_e_comp_dt = vchar(12),
    t_ruff_dft_dt = vchar(12),
    t_req_ml_date = vchar(12),
    h_unit_stem = vchar(7),
    tmlrtdt = vchar(12),
    trddt = vchar(12),
    tdocid = vchar(10),
    tdoctype = vchar(2),
    tdocnum = vchar(8),
    add_it = vchar(3),
    doc_present = vchar(3)
) = {
    resume field doc_table ;
}

'RETRIEVE' = {
/*      ***** RETRIEVE DOCUMENTS ***** */
/*      select all data documents -
/*      extract data from data_docmnt and document tables */
/*      document type, document number, issue number,
/*      dates : cer, dter, ml, required, estimated completion */
/*      unload doc_table and process rows */
    unloadtable doc_table
    {
/*      loop through rows of doc_table, filling in dates */
/*      extract document type and number */
        newdoc := select
            tdocid = squeeze(:doc_table.dcmnt_num)
            from document ;
        newdoc := select
            tdoctype = left(:tdocid, 2)
            from document ;
        newdoc := select
            tdocnum =
                squeeze(shift(:tdocid, -2))
            from document ;
        if tdoctype != '' then
            if tdocnum != '' then
/*      user has entered all mandatory fields */
                h_cer_date := '' ;
                h_dter_date := '' ;
                h_matl_list := '' ;
                h_rqrd_dt := '' ;
    
```



```

h_e_comp_dt := '' ;
newdoc := select
                                h_cer_date = data_documnt.cer_dt,
                                h_dter_date = data_documnt.dter_dt,
                                h_matl_list = data_documnt.ml_dt,
                                h_rqrd_dt = document.required_dt,
                                h_e_comp_dt = document.est_comp_dt
                                from data_documnt,document
                                where
                                data_documnt.doc_iss_num =
                                    :doc_table.issue_num
                                and
                                data_documnt.doc_type = :tdoctype
                                and
                                data_documnt.doc_num = :tdocnum
                                and
                                document.doc_iss_num =
                                    :doc_table.issue_num
                                and
                                document.doc_type = :tdoctype
                                and
                                document.doc_num = :tdocnum ;
/*      see if found document      */
inquire_ingres (rcount = rowcount) ;
if rcount != 0 then
/*      the document exists in document table      */
/*      convert dates to mo/da/yr format      */
doc_table.cer_date := date_slash(date_in = :h_cer_date) ;
doc_table.dter_date := date_slash(date_in = :h_dter_date) ;
doc_table.matl_list := date_slash(date_in = :h_matl_list) ;
doc_table.rqrd_dt := date_slash(date_in = :h_rqrd_dt) ;
doc_table.e_comp_dt := date_slash(date_in = :h_e_comp_dt) ;
/*      rough draft date - optional      */
trddt := '' ;
newdoc := select trddt =
                                doc_rough_dt.r_draft_dt
                                from doc_rough_dt
                                where
                                doc_rough_dt.doc_type = :tdoctype
                                and
                                doc_rough_dt.doc_num = :tdocnum
                                and
                                doc_rough_dt.doc_iss_num =
                                    :doc_table.issue_num ;
doc_table.ruff_dft_dt = date_slash(date_in = :trddt) ;
/*      ml request date - optional      */
tmlrdt := '' ;
newdoc := select tmlrdt = doc_rq_ml_dt.req_ml_dt
                                from doc_rq_ml_dt
                                where
                                doc_rq_ml_dt.doc_type = :tdoctype
                                and
                                doc_rq_ml_dt.doc_num = :tdocnum
                                and
                                doc_rq_ml_dt.doc_iss_num =
                                    :doc_table.issue_num ;
doc_table.req_ml_date := date_slash(date_in = :tmlrdt) ;
else
/*      the document does not exist in document table      */
message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +

```

```

                :doc_table.issue_num +
                ' was not found in the database' ;
        sleep 3 ;
/*      clear date fields if document not found */
        doc_table.cer_date := '' ;
        doc_table.dter_date := '' ;
        doc_table.matl_list := '' ;
        doc_table.rqrd_dt := '' ;
        doc_table.e_comp_dt := '' ;
        doc_table.ruff_dft_dt := '' ;
        doc_table.req_ml_date := '' ;
    endif ;
else
/*      a document type was entered but no document number */
    message 'Document ' + :tdoctype +
        ' must have document number' ;
    sleep 3 ;
endif ;
/*      a document with no document id was entered
    no error message since this could result from
    continuation of input line to end of line
*/
endif ;
} ; /*      end of unloadtable doc_table */
resume field doc_table.dcmnt_num ;
}
'ADD/REVISE' = {
/*      ***** DOCUMENTS ***** */
/*      add or revise data documents -
    change data in data_dcmnt and document tables */
/*      document type, document number, issue number,
    dates : cer, dter, ml, required, estimated completion */
/*      unload doc_table and process rows */
unloadtable doc_table
{
/*      loop through rows of doc_table */
/*      extract document type and number */
newdoc := select
    tdocid = squeeze(:doc_table.dcmnt_num)
    from document ;
newdoc := select
    tdoctype = left(:tdocid, 2)
    from document ;
newdoc := select
    tdocnum =
        squeeze(shift(:tdocid, -2))
    from document ;
if tdoctype != '' then
    if tdocnum != '' then
/*      user has entered all mandatory fields */
/*      ***** begin update transaction ***** */
        begin transaction ;
/*      see if document is in unit_doc table */
        add_it = 'NO' ;
        doc_present = 'NO' ;
        newdoc := select
            h_unit_stem = unit_doc.unit_stem
            from unit_doc
            where
                unit_doc.doc_iss_num =

```

```

                                :doc_table.issue_num
                                and
                                unit_doc.doc_type = :tdoctype
                                and
                                unit_doc.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*                                the document does not exist in unit_doc table */
/*                                get part numbers associated with this document,
                                insert in unit_doc table */
add_it := callframe UNITDOC (
                                UNITDOC.doc_num := :tdocnum;
                                UNITDOC.doc_type := :tdoctype;
                                UNITDOC.doc_iss_num := :doc_table.issue_num
                                ) ;
if add_it = 'YES' then
    message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +
            :doc_table.issue_num +
            ' was inserted into the database' ;
    sleep 3 ;
    doc_present = 'YES' ;
endif ;
else
    message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +
            :doc_table.issue_num +
            ' will be updated in the database' ;
    sleep 3 ;
    doc_present = 'YES' ;
endif ;
/*                                update document table */
/*                                see if document is in document table */
newdoc := select
                                h_rqrd_dt = document.required_dt,
                                h_e_comp_dt = document.est_comp_dt
                                from document
                                where
                                document.doc_iss_num =
                                    :doc_table.issue_num
                                and
                                document.doc_type = :tdoctype
                                and
                                document.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*                                the document does not exist in document table */
/*                                add document if it is now in unit_doc table */
if doc_present = 'YES' then
    insert into document (doc_type, doc_num, doc_iss_num,
                        est_comp_dt, required_dt)
        values (
            :tdoctype,
            :tdocnum,
            :doc_table.issue_num,
            :doc_table.e_comp_dt,
            :doc_table.rqrd_dt ) ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*                                unsuccessful insertion - abort transaction */
    message 'Error in inserting document ' +
            :tdoctype + ' ' + :tdocnum + ' ' +

```

```

                                :doc_table.issue_num +
                                ' into the document table' ;
                                sleep 3 ;
                                abort ;
                                endif ;
                                endif ;
else
/* the document is already in document table - update */
/* update only if there are changes to be made jlo 1/9/89 */
/* convert form fields to 12 character date for compare */
newdoc := select t_rqrd_dt = date(:doc_table.rqrd_dt),
                  t_e_comp_dt = date(:doc_table.e_comp_dt)
                  from component ;
if :h_rqrd_dt != :t_rqrd_dt or
:h_e_comp_dt != :t_e_comp_dt then
update document
set
  required_dt = :doc_table.rqrd_dt,
  est_comp_dt = :doc_table.e_comp_dt
where
  document.doc_iss_num =
    :doc_table.issue_num
  and
  document.doc_type = :tdoctype
  and
  document.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount != 0 then
/* successful update */
message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +
        :doc_table.issue_num +
        ' was updated in the document table' ;
sleep 3 ;
else
/* unsuccessful update - abort transaction */
message 'Error in updating document ' +
        :tdoctype + ' ' + :tdocnum + ' ' +
        :doc_table.issue_num +
        ' in the document table' ;
sleep 3 ;
abort ;
endif ;
endif ;
endif ;
/* update data_documnt table */
/* see if document is in data_documnt table */
newdoc := select
  h_cer_date = data_documnt.cer_dt,
  h_dter_date = data_documnt.dter_dt,
  h_matl_list = data_documnt.ml_dt
  from data_documnt
  where
    data_documnt.doc_iss_num =
      :doc_table.issue_num
    and
    data_documnt.doc_type = :tdoctype
    and
    data_documnt.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then

```

```

/*          the document does not exist in data_documnt table */
/*          add document if it is now in unit_doc table */
if doc_present = 'YES' then
    insert into data_documnt (doc_type, doc_num, doc_iss_num,
                             cer_dt, dter_dt, ml_dt)
        values (
            :tdoctype,
            :tdocnum,
            :doc_table.issue_num,
            :doc_table.cer_date,
            :doc_table.dter_date,
            :doc_table.matl_list) ;
    inquire_ingres (rcount = rowcount) ;
    if rcount = 0 then
/*          unsuccessful insertion - abort transaction */
        message 'Error in inserting document ' +
            :tdoctype + ' ' + :tdocnum + ' ' +
            :doc_table.issue_num +
            ' into the data document table' ;
        sleep 3 ;
        abort ;
    endif ;
endif ;
else
/*          the document is already in data_documnt table - update */
/*          update only if there are changes to be made jlo 1/9/89 */
/*          convert form fields to 12 character date for compare */
newdoc := select t_cer_date = date(:doc_table.cer_date),
                t_dter_date = date(:doc_table.dter_date),
                t_matl_list = date(:doc_table.matl_list)
            from component ;
if :h_cer_date != :t_cer_date or
:h_dter_date != :t_dter_date or
:h_matl_list != :t_matl_list then
    update data_documnt
        set
            cer_dt = :doc_table.cer_date,
            dter_dt = :doc_table.dter_date ,
            ml_dt = :doc_table.matl_list
        where
            data_documnt.doc_iss_num =
                :doc_table.issue_num
            and
            data_documnt.doc_type = :tdoctype
            and
            data_documnt.doc_num = :tdocnum ;
    inquire_ingres (rcount = rowcount) ;
    if rcount != 0 then
/*          successful update */
        message 'Document ' + :tdoctype + ' ' + :tdocnum + ' ' +
            :doc_table.issue_num +
            ' was updated in the data document table' ;
        sleep 3 ;
    else
/*          unsuccessful update - abort transaction */
        message 'Error in updating document ' +
            :tdoctype + ' ' + :tdocnum + ' ' +
            :doc_table.issue_num +
            ' in the data document table' ;
        sleep 3 ;
    endif
endif

```

```

        abort ;
    endif ;
endif ;
endif ;
/*
/*      update doc_rough_dt table      */
/*      see if document is in doc_rough_dt table      */
newdoc := select
        h_ruff_dft_dt = doc_rough_dt.r_draft_dt
        from doc_rough_dt
        where
            doc_rough_dt.doc_iss_num =
                :doc_table.issue_num
        and
            doc_rough_dt.doc_type = :tdoctype
        and
            doc_rough_dt.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      the document does not exist in doc_rough_dt table      */
/*      add document if it is now in unit_doc table      */
if doc_present = 'YES' then
/*      insert row in doc_rough_dt table if
        rough draft date input      */
if doc_table.ruff_dft_dt != '' then
    insert into doc_rough_dt (doc_type, doc_num,
        doc_iss_num,
        r_draft_dt)
        values (
            :tdoctype,
            :tdocnum,
            :doc_table.issue_num,
            :doc_table.ruff_dft_dt ) ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
/*      unsuccessful insertion - abort transaction      */
message 'Error in inserting document ' +
        :tdoctype + ' ' + :tdocnum + ' ' +
        :doc_table.issue_num +
        ' into the rough draft date table' ;
        sleep 3 ;
        abort ;
    endif ;
endif ;
endif ;
else
/*      the document is already in doc_rough_dt table - update */
/*      update only if there are changes to be made jlo 1/9/89 */
/*      convert form fields to 12 character date for compare */
newdoc := select t_ruff_dft_dt =
        date(:doc_table.ruff_dft_dt)
        from component ;
if :h_ruff_dft_dt != :t_ruff_dft_dt then
    update doc_rough_dt
    set
        r_draft_dt =
            :doc_table.ruff_dft_dt
    where
        doc_rough_dt.doc_iss_num =
            :doc_table.issue_num
        and

```

```

doc_rough_dt.doc_type = :tdoctype
and
doc_rough_dt.doc_num = :tdocnum ;
if rcount != 0 then
    successful update      */
    message 'Document ' + :tdoctype + ' ' + :tdocnum +
        ' , ' +
        :doc_table.issue_num +
        ' was updated in the rough draft' +
        ' date table' ;
    sleep 3 ;
else
    unsuccessful update - abort transaction      */
    message 'Error in updating document ' +
        :tdoctype + ' ' + :tdocnum + ' ' +
        :doc_table.issue_num +
        ' in the rough draft date table' ;
    sleep 3 ;
    abort ;
endif ;
endif ;
endif ;
update doc_rq_ml_dt table      */
/* see if document is in doc_rq_ml_dt table      */
newdoc := select
    h_req_ml_date = doc_rq_ml_dt.req_ml_dt
from
    doc_rq_ml_dt
where
    doc_rq_ml_dt.doc_iss_num =
        :doc_table.issue_num
    and
    doc_rq_ml_dt.doc_type = :tdoctype
    and
    doc_rq_ml_dt.doc_num = :tdocnum ;
inquire_ingres (rcount = rowcount) ;
if rcount = 0 then
    the document does not exist in doc_rq_ml_dt table */
/* add document if it is now in unit_doc table */
if doc_present = 'YES' then
    update doc_rq_ml_dt table if ml request date input */
    if doc_table.req_ml_date != '' then
        insert into doc_rq_ml_dt (doc_type, doc_num,
            doc_iss_num,
            req_ml_dt)
            values (
                :tdoctype,
                :tdocnum,
                :doc_table.issue_num,
                :doc_table.req_ml_date ) ;
        inquire_ingres (rcount = rowcount) ;
        if rcount = 0 then
            unsuccessful insertion - abort transaction      */
            message 'Error in inserting document ' +
                :tdoctype + ' ' + :tdocnum + ' ' +
                :doc_table.issue_num +
                ' into the ml requested date table' ;
            sleep 3 ;
            abort ;
        endif ;
    endif ;
endif ;

```

```

        endif ;
    endif ;
else
/*
the document is already in doc_rough_dt table - update */
/*
update only if there are changes to be made jlo 1/9/89 */
/*
convert form fields to 12 character date for compare */
newdoc := select t_req_ml_date =
                date(:doc_table.req_ml_date)
            from component ;
if :h_req_ml_date != :t_req_ml_date then
update doc_rq_ml_dt
set
req_ml_dt =
    :doc_table.req_ml_date
where
doc_rq_ml_dt.doc_iss_num =
    :doc_table.issue_num
and
doc_rq_ml_dt.doc_type = :tdoctype
and
doc_rq_ml_dt.doc_num = :tdocnum ;
if rcount != 0 then
/*
successful update */
message 'Document ' + :tdoctype + ' ' + :tdocnum +
    ' ' +
    :doc_table.issue_num +
    ' was updated in the ml requested' +
    ' date table' ;
sleep 3 ;
else
/*
unsuccessful update - abort transaction */
message 'Error in updating document ' +
    :tdoctype + ' ' + :tdocnum + ' ' +
    :doc_table.issue_num +
    ' in the ml requested date table' ;
sleep 3 ;
abort ;
endif ;
endif ;
endif ;
/*
***** end update transaction **** */
end transaction ;
else
/*
a document type was entered but no document number */
message 'Document ' + :tdoctype +
    ' must have document number' ;
sleep 3 ;
endif ;
/*
a document with no document id was entered
no error message since this could result from
continuation of input line to end of line
*/
endif ;
} ; /* end of unloadtable doc_table */
resume field doc_table.dcmnt_num ;
}
'DELETE_ONE' = {
/*
***** DOCUMENTS ***** */
/*
delete data document SET BY CURSOR in doc_table on form - */
if :doc_table.dcmnt_num != '' then

```



```

/*      pass a document to frame deletedoc      */
/*      must use select from database table, not form table */
callframe DELETEDOC (
    deletedoc.doc_table :=
        select
            dcmt_num = :doc_table.dcmt_num,
            issue_num = :doc_table.issue_num,
            rqrd_dt = :doc_table.rqrd_dt,
            cer_date = :doc_table.cer_date,
            e_comp_dt = :doc_table.e_comp_dt,
            ruff_dft_dt = :doc_table.ruff_dft_dt,
            dter_date = :doc_table.dter_date,
            matl_list = :doc_table.matl_list,
            req_ml_date = :doc_table.req_ml_date
        from component ) ;
else
    message 'ID for document to be deleted cannot be blank' ;
    sleep 3 ;
endif ;
resume field doc_table.dcmt_num ;
}
'ALL/DELETE' = {
/*      ***** DOCUMENTS *****      */
/*      delete all data documents in doc_table on form -
/*      unload doc_table and process rows      */
unloadtable doc_table
{
/*      do not call deletedoc if cursor on blank line -
/*      could be continuation after entry of dates on previous doc.
*/
if :doc_table.dcmt_num != '' then
/*      pass a document to frame deletedoc      */
/*      must use select from database table, not form table */
callframe DELETEDOC (
    deletedoc.doc_table :=
        select
            dcmt_num = :doc_table.dcmt_num,
            issue_num = :doc_table.issue_num,
            rqrd_dt = :doc_table.rqrd_dt,
            cer_date = :doc_table.cer_date,
            e_comp_dt = :doc_table.e_comp_dt,
            ruff_dft_dt = :doc_table.ruff_dft_dt,
            dter_date = :doc_table.dter_date,
            matl_list = :doc_table.matl_list,
            req_ml_date = :doc_table.req_ml_date
        from component ) ;
endif ;
} ; /*      end of unloadtable doc_table      */
resume field doc_table.dcmt_num ;
}
'CLEAR' = {
/*      clear screen of all data, put cursor at nomenclature,
/*      awaiting input      */
clear field all ;
resume field doc_table.dcmt_num ;
}
'HELP', key frskey1 =
{
    help_forms (subject = 'data documents',
        file = 'db1:[dbaprdh.help]newdoc.help');
}

```

```
}  
'XIT', key frskey3 =  
{  
    RETURN;  
}  
'STOP', key frskey2 =  
{  
    EXIT;  
}
```

Procedure for NEWPART frame
Written by Norman Stevens

NEWPART

```

/*
initialize (h_nom = vchar(12), h_stem = vchar(7), h_suffix = vchar(3),
h_desc = vchar(32), h_dsn_name = vchar(25), h_desn_engr = vchar(6),
h_qa_name = vchar(25), h_qa_engr = vchar(6), h_pa_name = vchar(34),
h_pa_code = vchar(18), h_prod_code = vchar(3), h_nomen = vchar(8),
h_nomtype = vchar(2), h_prod_name = vchar(25), h_prod_engr = vchar(9),
h_qc_name = vchar(25), h_qc_engr = vchar(9), check_engr = vchar(9),
n_sf_name = vchar(25), n_sf_enum = vchar(6), check_type = vchar(2),
wpn_pgm = vchar(4), test_cfg = vchar(7), h_mod_num = vchar(3),
rcount = integer1, desn_code = vchar(3), no_of_errors = integer1,
answer = vchar(1), h_mfg_code = vchar(18), h_mfg_num = vchar(4),
h_supplier = vchar(24), supplier.sup_code = vchar(3),
supplier.wpn_code = vchar(4), supplier.mod_code = vchar(3)) =

```

```

{
h_nom      := '          ';
h_stem     := '          ';
h_suffix   := '          ';
h_desc     := '          ';
h_desn_engr := '          ';
h_dsn_name := '          ';
h_qa_name  := '          ';
h_qa_engr  := '          ';
h_pa_code  := '          ';
h_prod_code := '          ';
h_pa_name  := '          ';
h_prod_engr := '          ';
h_prod_name := '          ';
h_qc_name  := '          ';
h_qc_engr  := '          ';
h_nomtype  := '          ';
h_nomen    := '          ';
check_engr := '          ';
resume field nomenclature;
}

```

'RETRIEVE' =

```

{
/*
DSL to retrieve the part number and suffix from PREMIS
*/
h_nom := nomenclature;
h_stem := part_num;
h_suffix := unit_suffix;
clear field all;
nomenclature := h_nom;
part_num := h_stem;
unit_suffix := h_suffix;
if nomenclature = '' then
if part_num = '' then
answer := prompt 'You must enter a part number or a nomenclature' +
'. Hit <CR> to continue.';
else
message 'Retrieving Information ...';
newpart := select h_nomtype = c.com_nom_type, h_nomen = c.com_nom_num
from component c where c.unit_stem = :part_num and
c.unit_da_suf = :unit_suffix;
inquire_ingres(rcount = rowcount);

```

```

if rcount = 0 then
    newpart := select wpn_pgm = w.wpn_pgm_code,
        h_mod_num = w.wpn_mod_num, test_cfg = w.wpn_test_cfg
        from weapon w where :part_num = w.unit_stem
            and :unit_suffix = w.unit_da_suf;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        answer := prompt 'This component does not exist in PREMIS' +
            '. Use ADD COMPONENT to add the part to PREMIS';
        resume field nomenclature;
    else
        if test_cfg = '' then
            nomenclature = wpn_pgm + '-' + h_mod_num;
        else
            nomenclature = wpn_pgm + '/' + test_cfg + '-' + h_mod_num;
        endif;
    endif;
else
    nomenclature = :h_nomtype + :h_nomen;
    h_nomtype = ' ';
    h_nomen = ' ';
endif;
endif;
else
    message 'Retrieving Information ... ';
    newpart := select part_num = c.unit_stem, unit_suffix = c.unit_da_suf
        from component c where c.com_nom_type = left(:nomenclature, 2)
        and c.com_nom_num = shift(:nomenclature, -2);
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        newpart := select part_num = c.unit_stem, unit_suffix = c.unit_da_suf
            from component c where c.com_nom_type = left(:nomenclature, 1)
            and c.com_nom_num = shift(:nomenclature, -2);
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            answer := prompt 'Component with that nomenclature is not in ' +
                'PREMIS. Please use ADD COMPONENT to add to PREMIS';
            resume field nomenclature;
        endif;
    endif;
endif;
endif;
/*
    DSL to retrieve the unit description from PREMIS
*/
if rcount != 0 then
    newpart := select unit_desc = u.unit_desc from unit_desc u
        where u.unit_stem = :part_num and u.unit_da_suf = :unit_suffix;
/*
    DSL to retrieve the design agency from PREMIS
*/
newpart := select desn_agency = c.company_name, desn_code = u.dsn_mfg_code
    from company c, unit u where c.mfg_code = u.dsn_mfg_code and
        (u.unit_stem = :part_num and u.unit_da_suf = :unit_suffix);
/*
    DSL to retrieve the design engineer's name from PREMIS
    and concatenate three fields into one
*/
newpart := select h_dsn_name = p.first_name + ' ' + p.middl_init + ' ' +
    p.last_name, desn_orgn = p.asnd_orgn, h_desn_engr = u.da_desn_engr,
    desn_phone = '(' + left(p.phone_num, 3) + ')' +

```

```

left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
from person p, unit u where u.da_desn_engr = p.id_num and
(u.unit_stem = :part_num and u.unit_da_suf = :unit_suffix);
inquire_ingres(rcount = rowcount);
if rcount = 0 then
    desn_engr := '';
    desn_phone := '';
    desn_orgn := '';
    newpart := select desn_orgn = d.division from da_desn_orgn d, unit u
        where :part_num = u.unit_stem and :unit_suffix = u.unit_da_suf
            and u.da_desn_engr = d.division;
else
    desn_engr := h_dsn_name;
    if desn_phone = '()-' then
        desn_phone := '';
    endif;
endif;

```

/*

DSL to retrieve the quality assurance engineer's name
from PREMIS and concatenate three fields into one

*/

```

newpart := select h_qa_name = p.first_name + ' ' + p.middl_init + ' ' +
p.last_name, qa_engr = p.asnd_orgn, h_qa_engr = u.da_qa_engr,
qa_phone = '(' + left(p.phone_num, 3) + ')' +
left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
from person p, unit u where (u.unit_stem = :part_num and
u.unit_da_suf = :unit_suffix) and u.da_qa_engr = p.id_num;
inquire_ingres(rcount = rowcount);
if rcount != 0 then
    qa_engr := h_qa_name;
    if qa_phone = '()-' then
        qa_phone := '';
    endif;
endif;

```

/*

DSL to retrieve the nuclear safety engineer's name
from PREMIS and concatenate three fields into one

*/

```

newpart := select n_sf_name = p.first_name + ' ' + p.middl_init + ' ' +
p.last_name, nuc_sfty_org = p.asnd_orgn, n_sf_enum = u.da_n_sf_engr,
nuc_sfty_phn = '(' + left(p.phone_num, 3) + ')' +
left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
from person p, unit u where (u.unit_stem = :part_num and
u.unit_da_suf = :unit_suffix) and u.da_n_sf_engr = p.id_num;
inquire_ingres(rcount = rowcount);
if rcount != 0 then
    nuc_sfty_egr := n_sf_name;
    if nuc_sfty_phn = '()-' then
        nuc_sfty_phn := '';
    endif;
endif;

```

/*

DSL to retrieve the production agency from PREMIS

*/

```

newpart := select prodagency = c.company_name, h_pa_code = u.prđ_mfg_code
from company c, unit u where c.mfg_code = u.prđ_mfg_code and
(u.unit_stem = :part_num and u.unit_da_suf = :unit_suffix);
h_pa_name := :prodagency;

```

/*

DSL to retrieve the production engineer's name from

PREMIS and concatenate three fields into one

```
*/
newpart := select h_prod_name = p.first_name + ' ' + p.middl_init + ' ' +
  p.last_name, h_prod_engr = u.pa_prod_engr from person p, unit u
  where (u.unit_stem = :part_num and u.unit_da_suf = :unit_suffix)
  and u.pa_prod_engr = p.id_num;
  inquire_ingres(rcount = rowcount);
  if rcount != 0 then
    prodengr := h_prod_name;
  endif;

/*
  DSL to retrieve the quality control engineer's name from
  PREMIS and concatenate three fields into one
*/
newpart := select h_qc_name = p.first_name + ' ' + p.middl_init + ' ' +
  p.last_name, h_qc_engr = u.pa_qc_engr from person p, unit u
  where (u.unit_stem = :part_num and u.unit_da_suf = :unit_suffix)
  and u.pa_qc_engr = p.id_num;
  inquire_ingres(rcount = rowcount);
  if rcount != 0 then
    qc_engr := h_qc_name;
  endif;

/*
  DSL to retrieve the next assembly data from PREMIS
*/
newpart.next_as := select unit_stem = g.unit_stem,
  unit_sffx = g.unit_da_suf from gozinto g
  where g.part_stem = :part_num and g.part_da_suf = :unit_suffix;
unloadable next_as
begin
  newpart := select h_nomtype = c.com_nom_type, h_nomen = c.com_nom_num
    from component c where c.unit_stem = :next_as.unit_stem and
      c.unit_da_suf = :next_as.unit_sffx;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    newpart := select wpn_pgm = w.wpn_pgm_code,
      h_mod_num = w.wpn_mod_num, test_cfg = w.wpn_test_cfg
      from weapon w where :next_as.unit_stem = w.unit_stem
      and :next_as.unit_sffx = w.unit_da_suf;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      next_as.cmpnt_nom = '';
    else
      if test_cfg = '' then
        next_as.cmpnt_nom = wpn_pgm + '-' + h_mod_num;
      else
        next_as.cmpnt_nom = wpn_pgm + '/' + test_cfg + '-' + h_mod_num;
      endif;
    endif;
  else
    next_as.cmpnt_nom := h_nomtype + h_nomen;
  endif;
  newpart := select h_desc = u.unit_desc from unit_desc u
    where :next_as.unit_stem = u.unit_stem
    and :next_as.unit_sffx = u.unit_da_suf;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    next_as.u_desc = '';
  else
    next_as.u_desc := h_desc;
  endif;
end;
```

```

        endif;
    end;

/*
    OSL to retrieve the supplier from PREMIS
*/
supplier := select sup_code = m.mfg_code, wpn_sys = m.wpn_pgm_code,
    wpn_code = m.wpn_pgm_code, mod_code = m.wpn_mod_num,
    mod_num = m.wpn_mod_num, mfg = c.company_name from mfg_inst m, company c
    where m.unit_stem = :part_num and m.unit_da_suf = :unit_suffix
    and m.mfg_code = c.mfg_code
endif;
}

'ADD/REVISE' =
{
no_of_errors := 0;
if part_num = '' then
    if nomenclature = '' then
        answer := prompt 'You must enter either a part number or a nomenclature'+
            '. Hit <CR> to continue';
        no_of_errors := no_of_errors + 1;
    else
        newpart := select part_num = c.unit_stem, unit_suffix = c.unit_da_suf
            from component c where :nomenclature = c.com_nom_type + c.com_nom_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors := no_of_errors + 1;
            answer := prompt 'No component exists with that part number.' +
                ' Use ADD COMPONENT to add it to PREMIS';
        endif;
    endif;
else
    newpart := select nomenclature = c.com_nom_type + c.com_nom_num
        from component c where :part_num = c.unit_stem
        and :unit_suffix = c.unit_da_suf;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        newpart := select wpn_pgm = w.wpn_pgm_code,
            h_mod_num = w.wpn_mod_num, test_cfg = w.wpn_test_cfg
            from weapon w where :part_num = w.unit_stem
            and :unit_suffix = w.unit_da_suf;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors := no_of_errors + 1;
            answer := prompt 'No component exists with that part number.'
                + ' Use ADD COMPONENT to add it to PREMIS';
        else
            if test_cfg = '' then
                nomenclature = wpn_pgm + '-' + h_mod_num;
            else
                nomenclature = wpn_pgm + '/' + test_cfg + '-' + h_mod_num;
            endif;
        endif;
    endif;
endif;
if no_of_errors = 0 then
/*
    OSL to store the unit description in PREMIS
*/
    newpart := select h_desc = u_d.unit_desc from unit_desc u_d

```

```

        where (:part_num = u_d.unit_stem and :unit_suffix = u_d.unit_da_suf);
inquire_ingres(rcount = rowcount);
if rcount = 0 then
    insert into unit_desc (unit_stem, unit_da_suf, unit_desc)
        values (:part_num, :unit_suffix, :unit_desc);
else
    update unit_desc u_d set unit_desc = :unit_desc
        where :part_num = u_d.unit_stem and :unit_suffix = u_d.unit_da_suf;
endif;

```

/*

DSL to store the design engineer's employee number in PREMIS

*/

```

if desn_engr != h_dsn_name then
    if desn_engr = '' then
        h_desn_engr = '';
        desn_code = '';
    else
        newpart := select rcount = count (*) from person p
            where p.last_name = :desn_engr and p.asnd_orgn = :desn_orgn;
        if rcount = 0 then
            newpart := select rcount = count(*) from person p
                where p.asnd_orgn = :desn_orgn
                    and :desn_engr = left(p.first_name, 1) + " " + p.middl_init +
                        " " + p.last_name;
            if rcount = 0 then
                no_of_errors = no_of_errors + 1;
                answer := prompt 'There is no engineer named ' + desn_engr +
                    ' who works for ' + desn_orgn + '. Hit <CR> to continue';
            elseif rcount = 1 then
                newpart := select h_desn_engr = p.id_num from person p
                    where p.asnd_orgn = :desn_orgn
                        and :desn_engr = left(p.first_name, 1) + " " +
                            p.middl_init + " " + p.last_name;
            else
                no_of_errors = no_of_errors + 1;
                answer := prompt 'More than one engr named ' +
                    desn_engr + ' works for ' + desn_orgn + '. Please call '
                    + 'your DBA.';
            endif;
        elseif rcount = 1 then
            newpart := select h_desn_engr = p.id_num from person p
                where p.asnd_orgn = :desn_orgn and :desn_engr = p.last_name;
        else
            no_of_errors = no_of_errors + 1;
            answer := prompt 'More than one engineer named ' +
                desn_engr + ' works for ' + desn_orgn + '. Please enter '
                + 'the initials.';
        endif;
        if rcount = 1 then
            newpart := select check_engr = d.empl_num from da_desn_engr d
                where d.empl_num = :h_desn_engr;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                no_of_errors = no_of_errors + 1;
                answer := prompt 'This engineer is not a valid Design Engineer' +
                    '. Hit <CR> to continue';
            else
                h_desn_engr = :check_engr;
                check_engr = '';
                newpart := select h_dsn_name = p.first_name + ' ' + p.middl_init

```



```

        + ' ' + p.last_name,
        desn_phone = '(' + left(p.phone_num, 3) + ')' +
        left(shift(p.phone_num, -3), 3) + '-',
        + shift(p.phone_num, -6) from person p
        where p.id_num = :h_desn_engr;
    desn_engr := h_dsn_name;
    if desn_phone = '()' then
        desn_phone := '';
    endif;
    newpart := select desn_agency = c.company_name
    from company c where c.mfg_code = :desn_code;
    endif;
endif;
endif;
endif;
if desn_engr = '' then
    if desn_orgn != '' then
        newpart := select h_desn_engr = d.division from da_desn_orgn d
        where :desn_orgn = d.division;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors = no_of_errors + 1;
            answer := prompt 'This design organization is invalid. Hit <CR> ' +
            'to continue.';
            h_desn_engr := '';
        endif;
    endif;
endif;
if desn_code = '' then
    if h_desn_engr = '' then
        desn_code := ''
    else
        newpart := select desn_code = p.mfg_code from person p
        where :h_desn_engr = p.id_num;
    endif;
endif;

```

/*

DSL to store the qa engineer's employee number in PREMIS

*/

```

if qa_engr != h_qa_name then
    if qa_engr = '' then
        h_qa_engr = '';
    else
        newpart := select h_qa_engr = p.id_num from person p
        where p.last_name = :qa_engr and p.asnd_orgn = :qa_engr;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors = no_of_errors + 1;
            answer := prompt 'There is no engineer named ' + qa_engr +
            ' who works for ' + qa_engr + '. Hit <CR> to continue.';
        else
            newpart := select check_engr = q.empl_num from da_qa_engr q
            where q.empl_num = :h_qa_engr;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                no_of_errors = no_of_errors + 1;
                answer := prompt 'This engineer is not a valid QA Engineer' +
                '. Hit <CR> to continue.';
            else
                h_qa_engr = check_engr;
            end if;
        end if;
    end if;
end if;

```

```

        check_engr := '
        newpart := select h_qa_name = p.first_name + ' ' + p.middl_init
        + ' ' + p.last_name,
        qa_phone = '(' + left(p.phone_num, 3) + ')' +
        left(shift(p.phone_num, -3), 3) + '-'
        + shift(p.phone_num, -6)
        from person p where p.id_num = :h_qa_engr;
        qa_engr := h_qa_name;
        if qa_phone = '()-' then
            qa_phone := '';
        endif;
    endif;
endif;
endif;
endif;
/*
    DSL to edit the nuclear safety engineer in PREMIS
*/
if n_sf_name != nuc_sfty_egr then
    if nuc_sfty_egr = '' then
        n_sf_enum := '';
    else
        newpart := select n_sf_enum = p.id_num from person p
        where p.last_name = :nuc_sfty_egr and p.asnd_orgn = :nuc_sfty_org;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors := no_of_errors + 1;
            answer := prompt :nuc_sfty_egr + ' is not found in PREMIS'+
            '. Hit <CR> to continue';
        else
            newpart := select check_engr = n.empl_num from da_n_sf_engr n
            where n.empl_num = :n_sf_enum;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                no_of_errors = no_of_errors + 1;
                answer := prompt 'This engineer is not a valid nuclear safety '+
                'Engineer. Hit <CR> to continue';
            else
                n_sf_enum := check_engr;
                check_engr := '
                newpart := select nuc_sfty_egr = p.first_name + ' ' +
                p.middl_init + ' ' + p.last_name,
                nuc_sfty_phn = '(' + left(p.phone_num, 3) + ')' +
                left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
                from person p where :n_sf_enum = p.id_num;
                n_sf_name := :nuc_sfty_egr;
                if nuc_sfty_phn = '()-' then
                    nuc_sfty_phn := '';
                endif;
            endif;
        endif;
    endif;
endif;
endif;
/*
    DSL to store the production agency in PREMIS
*/
if h_pa_name != prodagency then
    if prodagency = '' then
        h_pa_code = '';
    else

```

```

h_pa_code := prodagency;
newpart := select h_prod_code = c.mfg_code from company c
  where :h_pa_code = c.mfg_code;
inquire_ingres (rcount=rowcount);
if rcount = 0 then
  newpart := select h_prod_code = c_s.mfg_code from company_syn c_s
    where :h_pa_code = c_s.company_name;
  inquire_ingres (rcount=rowcount);
  if rcount = 0 then
    h_pa_code = '';
    no_of_errors = no_of_errors + 1;
    answer := prompt 'The Production Agency entered is invalid' +
      '. Hit <CR> to continue';
  else
    h_pa_code := :h_prod_code;
  endif;
else
  h_pa_code := :h_prod_code;
endif;
if h_pa_code = '' then
  h_pa_name = '';
else
  newpart := select prodagency = c.company_name from company c
    where c.mfg_code = :h_pa_code;
  h_pa_name := :prodagency;
endif;
endif;
/*
  DSL to store the prod engineer's employee number in PREMIS
*/
if prodengr != h_prod_name then
  if prodengr = '' then
    h_prod_engr = '';
  else
    newpart := select rcount = count (*) from person p
      where :prodengr = p.last_name and :h_pa_code = p.mfg_code;
    if rcount = 0 then
      newpart := select rcount = count(*) from person p
        where :h_pa_code = p.mfg_code
          and :prodengr = left(p.first_name, 1) + " " + p.middl_init +
            " " + p.last_name;
      if rcount = 0 then
        no_of_errors = no_of_errors + 1;
        answer := prompt 'There is no engineer named ' + prodengr +
          ' who works for ' + prodagency + '.Hit <CR> to continue';
      elseif rcount = 1 then
        newpart := select h_prod_engr = p.id_num from person p
          where :h_pa_code = p.mfg_code
            and :prodengr = left(p.first_name, 1) + " " +
              p.middl_init + " " + p.last_name;
      else
        no_of_errors = no_of_errors + 1;
        answer := prompt 'More than one engineer named ' + prodengr +
          ' works for ' + prodagency + '. Please call your DBA.';
      endif;
    elseif rcount = 1 then
      newpart := select h_prod_engr = p.id_num from person p
        where :h_pa_code = p.mfg_code and :prodengr = p.last_name;
    else

```

```

no_of_errors = no_of_errors + 1;
answer := prompt 'More than one engineer named ' + prodengr +
' works for ' + prodagency + '. Please enter the initials.';
endif;
if rcount = 1 then
    newpart := select check_engr = p.id_num from pa_prod_engr p
    where p.id_num = :h_prod_engr;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        no_of_errors = no_of_errors + 1;
        answer := prompt 'This engineer is not a valid Product ' +
'Engineer. Hit <CR> to continue.';
    else
        newpart := select h_prod_name = p.first_name + ' ' +
p.middl_init + ' ' + p.last_name from person p
        where p.id_num = :h_prod_engr;
        prodengr := h_prod_name;
    endif;
endif;
endif;
endif;
endif;
/*
    DSL to store the qc engineer's id number in PREMIS
*/
if qc_engr != h_qc_name then
    if qc_engr = '' then
        h_qc_engr = '';
    else
        newpart := select rcount = count (*) from person p
        where :h_pa_code = p.mfg_code and :qc_engr = p.last_name;
        if rcount = 0 then
            newpart := select rcount = count (*) from person p
            where :h_pa_code = p.mfg_code
            and :qc_engr = left(p.first_name, 1) + " " + p.middl_init +
" " + p.last_name;
            if rcount = 0 then
                no_of_errors = no_of_errors + 1;
                answer := prompt 'There is no engineer named ' + qc_engr +
' who works for ' + prodagency + '. Hit <CR> to continue';
            elseif rcount = 1 then
                newpart := select h_qc_engr = p.id_num from person p
                where :h_pa_code = p.mfg_code
                and :qc_engr = p.first_name + " " + p.middl_init + " " +
p.last_name;
            else
                no_of_errors = no_of_errors + 1;
                answer := prompt 'More than one engineer named ' + qc_engr +
' works for ' + prodagency + '. Please call your DBA.';
            endif;
        elseif rcount = 1 then
            newpart := select h_qc_engr = p.id_num from person p
            where :h_pa_code = p.mfg_code and :qc_engr = p.last_name;
        else
            no_of_errors = no_of_errors + 1;
            answer := prompt 'More than one engineer named ' + qc_engr +
' works for ' + prodagency + '. Please enter the initials.';
        endif;
    endif;
    if rcount = 1 then
        newpart := select check_engr = qc.id_num from pa_qc_engr qc
        where qc.id_num = :h_qc_engr;
    endif;

```

```

        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors = no_of_errors + 1;
            answer := prompt 'This engineer is not a valid QC Engineer' +
                '. Hit <CR> to continue.';
        else
            newpart := select h_qc_name = p.first_name + ' ' + p.middl_init
                + ' ' + p.last_name from person p
                where p.id_num = :h_qc_engr;
            qc_engr := h_qc_name;
        endif;
    endif;
endif;
endif;
/*
    OSL to store the unit data in PREMIS
*/
if no_of_errors = 0 then
    newpart := select check_engr = u.da_desn_engr from unit u
        where :part_num = u.unit_stem and :unit_suffix = u.unit_da_suf;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        insert into unit (unit_stem, unit_da_suf, da_desn_engr, dsn_mfg_code,
            da_qa_engr, da_n_sf_engr, prd_mfg_code, pa_prod_engr, pa_qc_engr)
            values (:part_num, :unit_suffix, :h_desn_engr, :dsn_code,
                :h_qa_engr, :n_sf_enum, :h_pa_code, :h_prod_engr, :h_qc_engr);
        message 'The part record for ' + :part_num + ' was added';
        sleep 5;
    else
        update unit u set da_desn_engr = :h_desn_engr,
            dsn_mfg_code = :dsn_code, da_qa_engr = :h_qa_engr,
            da_n_sf_engr = :n_sf_enum, prd_mfg_code = :h_pa_code,
            pa_prod_engr = :h_prod_engr, pa_qc_engr = :h_qc_engr
            where :part_num = u.unit_stem and :unit_suffix = u.unit_da_suf;
        message 'The part record for ' + :part_num + ' was revised';
        sleep 5;
    endif;
endif;
/*
    OSL to store the next assembly information in PREMIS
*/
unloadtable next_as
begin
    if next_as.unit_stem = '' then
        if next_as.cmpnt_nom = '' then
            next_as.cmpnt_nom = '';
        else
            newpart := select h_stem = c.unit_stem, h_suffix = c.unit_da_suf
                from component c
                where :next_as.cmpnt_nom = c.com_nom_type + c.com_nom_num;
            inquire_ingres(rcount=rowcount);
            if rcount = 0 then
                answer := prompt :next_as.cmpnt_nom + ' is not found in PREMIS'+
                    '. Hit <CR> to continue.';
                next_as.unit_stem = '';
                next_as.unit_sffx = '';
            else
                next_as.unit_stem = h_stem;
                next_as.unit_sffx = h_suffix;
            endif;
        end if;
    end if;
end if;

```

```

endif;
else
  newpart := select h_nomtype = c.com_nom_type,
    h_nomen = c.com_nom_num from component c
    where c.unit_stem = :next_as.unit_stem
      and c.unit_da_suf = :next_as.unit_sffx;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    newpart := select wpn_pgm = w.wpn_pgm_code,
      h_mod_num = w.wpn_mod_num, test_cfg = w.wpn_test_cfg
      from weapon w where :next_as.unit_stem = w.unit_stem
        and :next_as.unit_sffx = w.unit_da_suf;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      answer := prompt :next_as.unit_stem + '-' + :next_as.unit_sffx +
        ' is not in PREMIS. Hit <CR> to continue.';
      next_as.unit_stem = '';
      next_as.unit_sffx = '';
    else
      if test_cfg = '' then
        next_as.cmpnt_nom = wpn_pgm + '-' + h_mod_num;
      else
        next_as.cmpnt_nom = wpn_pgm + '/' + test_cfg + '-' +
          h_mod_num;
      endif;
    endif;
  endif;
endif;
if next_as.unit_stem != '' then
  h_desc = '';
  newpart := select h_desc = u.unit_desc from unit_desc u
    where u.unit_stem = :next_as.unit_stem
      and u.unit_da_suf = :next_as.unit_sffx;
  next_as.u_desc = h_desc;
  newpart := select h_stem = g.unit_stem from gozinto g
    where g.unit_stem = :next_as.unit_stem
      and g.unit_da_suf = :next_as.unit_sffx
      and g.part_stem = :part_num and g.part_da_suf = :unit_suffix;
  inquire_ingres(rcount=rowcount);
  if rcount = 0 then
    insert into gozinto (unit_stem, unit_da_suf, part_stem,
      part_da_suf) values (:next_as.unit_stem, :next_as.unit_sffx,
        :part_num, :unit_suffix);
  endif;
endif;
end;
no_of_errors := 0;
unloadtable supplier
begin
  h_mfg_code := '';
  wpn_pgm := '';
  h_mod_num := '';
  if supplier.mfg != '' then
    newpart := select h_mfg_code = c.mfg_code from company c
      where :supplier.mfg = c.mfg_code
        or squeeze(:supplier.mfg) = squeeze(left(c.company_name, 24));
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      newpart := select h_mfg_code = c_s.mfg_code from company_syn c_s
        where :supplier.mfg = c_s.company_name;
    endif;
  endif;

```

```

        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors := no_of_errors + 1;
            set_forms column newpart supplier(underline(mfg) = 1);
        else
            set_forms column newpart supplier(underline(mfg) = 0);
        endif;
    endif;
endif;
if supplier.wpn_sys != '' then
    if supplier.mod_num = '' then
        newpart := select wpn_pgm = w_p.wpn_pgm_code from weapon_pgm w_p
            where :supplier.wpn_sys = w_p.wpn_pgm_code;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors := no_of_errors + 1;
            answer := prompt 'The weapon program is invalid; hit <CR> to '+
                'continue.';
            resume field supplier.wpn_sys;
        endif;
    else
        newpart := select wpn_pgm = w_p.wpn_pgm_code,
            h_mod_num = w_p.wpn_mod_num from weapon_pgm w_p
            where :supplier.wpn_sys = w_p.wpn_pgm_code
            and :supplier.mod_num = w_p.wpn_mod_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            no_of_errors := no_of_errors + 1;
            answer := prompt 'The weapon program is invalid; hit <CR> to '+
                'continue.';
            resume field supplier.wpn_sys;
        endif;
    endif;
endif;
if no_of_errors = 0 then
    if h_mfg_code != '' or wpn_pgm != '' then
        if :supplier.sup_code = '' and :supplier.wpn_code = '' and
            :supplier.mod_code = '' then
/*
    DSL to insert a supplier and/or a weapon program in PREMIS
*/
            newpart := select h_mfg_num = mi.mfg_inst_num from mfg_inst mi
                where :part_num = mi.unit_stem
                and :unit_suffix = mi.unit_da_suf
                and :h_mfg_code = mi.mfg_code
                and :wpn_pgm = mi.wpn_pgm_code
                and :h_mod_num = mi.wpn_mod_num;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                newpart :=
                    select h_mfg_num = vchar(max(int4(mi.mfg_inst_num))+1)
                        from mfg_inst mi;
                insert into mfg_inst (mfg_inst_num, mfg_code, wpn_pgm_code,
                    wpn_mod_num, unit_stem, unit_da_suf) values (:h_mfg_num,
                    :h_mfg_code, :wpn_pgm, :h_mod_num, :part_num, :unit_suffix);
            endif;
        else
/*
    DSL to update a supplier and/or a weapon program in PREMIS
*/

```

```

        newpart := select h_mfg_num = mi.mfg_inst_num from mfg_inst mi
        where :part_num = mi.unit_stem
        and :unit_suffix = mi.unit_da_suf
        and :supplier.sup_code = mi.mfg_code
        and :supplier.wpn_code = mi.wpn_pgm_code
        and :supplier.mod_code = mi.wpn_mod_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 1 then
            update mfg_inst mi set mfg_code = :h_mfg_code,
            wpn_pgm_code = :wpn_pgm, wpn_mod_num = :h_mod_num
            where :h_mfg_num = mi.mfg_inst_num;
        endif;
    endif;
endif;
if h_mfg_code != '' then
    newpart := select h_supplier = c.company_name from company c
    where :h_mfg_code = c.mfg_code;
    supplier.mfg := h_supplier;
endif;
else
    answer := prompt 'The underlined values are invalid; Data not '
    + 'saved; Hit <CR> to continue';
endif;
end;
endif;
}

'INSERT/COMPONENT'=
{
    if part_num = '' then
        answer := prompt 'You must enter a part number to add a component' +
        '. Hit <CR> to continue.';
        resume field part_num;
    else
        newpart := select check_type = c.com_nom_type from component c
        where :part_num = c.unit_stem and :unit_suffix = c.unit_da_suf;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            newpart := select h_nomtype = left(:nomenclature, 2),
            h_nomen = shift(:nomenclature, -2) from component;
            newpart := select check_type = c.type from com_nom_type c
            where :h_nomtype = c.type;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                answer := prompt 'The component type is invalid. Hit <CR> to ' +
                'continue.';
            else
                insert into component(unit_stem, unit_da_suf, com_nom_type,
                com_nom_num) values(:part_num, :unit_suffix, :h_nomtype,
                :h_nomen);
                message 'Component with part number ' + :part_num + ' was ' +
                'added to PREMIS';
                sleep 5;
            endif;
        else
            answer := prompt 'The component is already in PREMIS. Hit <CR> to ' +
            'continue.';
        endif;
    endif;
    resume field unit_desc;
}

```



```

}
'CLEAR' =
{
    clear field all;
    set_forms column newpart supplier (underline(mfg) = 0);
    set_forms column newpart supplier (underline(wpn_sys) = 0,
        underline(mod_num)=0);
    resume field nomenclature;
}
'HELP' , key frskey1 =
{
    help_forms (subject = 'add new part number',
        file = 'db1:[dbaprd.b.help]newpart.help');
}
'XIT', key frskey3 =
{
    RETURN;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

```

/*
                                frame: NOTCOMPDC
DRIVER FOR REPORT TO LIST ALL DATA DOCUMENTS THAT ARE NOT COMPLETED
***** NOT IMPLEMENTED AS OF 9/27/88 *****
                                written by john orman 9/27/88

*/
initialize (
) = {
/*
                                ***** DISPLAY SCREEN ***** */
    redisplay ;
/*
                                ***** SHOW THAT SCREEN NOT IMPLEMENTED ***** */
    message '***** This report has not been implemented *****' ;
    sleep 5 ;
    RETURN ;

'DISPLAY' = {
}
'FILE' = {
}
'PRINTER' =
{
}
'HELP', key frskey1 =
{
    help forms (subject = 'List data documents not completed',
        file = 'db1:[dbaprdc.help]notcompdc.help');
}
'XIT', key frskey3 =
{
    RETURN ;
}
'STOP', key frskey2 =
{
    EXIT ;
}
}

```

/*

Procedure for PARTS frame PARTS
Written by Norman Stevens

*/

```
initialize (rcount = integer1, h_desc = vchar(32), desn_enum = vchar(6),
           qa_enum = vchar(6), qa_org = vchar(6), qa_name = vchar(12),
           desn_org = vchar(6), pa_code = vchar(3), w_pgm = vchar(8),
           answer = vchar(1), h_nomen = vchar(12), cmpnts.suffix = vchar(3)) =
```

```
{
resume field type;
}
```

'LIST by TYPE' =

{

/*

DSL necessary to retrieve the parts of a certain type

*/

```
cmpnts := select subject = c.com_nom_type + c.com_nom_num,
  part_num = c.unit_stem, suffix = c.unit_da_suf from component c
where :type = c.com_nom_type;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
  answer := prompt 'There are no components of this type in PREMIS. Hit '
    + '<CR> to continue';
  resume field type;
else
  unloadtable cmpnts
  begin
    h_desc      := '';
    desn_enum   := '';
    qa_enum     := '';
    pa_code     := '';
    desn_org    := '';
    qa_name     := '';
    qa_org      := '';
    w_pgm       := '';
    parts := select h_desc = u_d.unit_desc from unit_desc u_d
      where :cmpnts.part_num = u_d.unit_stem
        and :cmpnts.suffix = u_d.unit_da_suf;
    cmpnts.description := h_desc;
    parts := select desn_enum = u.da_desn_engr, qa_enum = u.da_qa_engr,
      pa_code = u.prn_mfg_code from unit u
      where :cmpnts.part_num = u.unit_stem
        and :cmpnts.suffix = u.unit_da_suf;
    cmpnts.prod_agency := pa_code;
    if desn_enum != '' then
      parts := select desn_org = p.asnd_orgn from person p
        where :desn_enum = p.id_num;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        desn_org := :desn_enum;
      endif;
      cmpnts.desn_orgn := desn_org;
    endif;
    if qa_enum != '' then
      parts := select qa_name = p.last_name, qa_org = p.asnd_orgn
        from person p where :qa_enum = p.id_num;
      cmpnts.qa_engr_name := qa_name;
      cmpnts.qa_orgn := qa_org;
    endif;
```

```

        parts := select w_pgm = m.wpn_pgm_code + '-' + m.wpn_mod_num
        from mfg_inst m where :cmpnts.part_num = m.unit_stem
        and :cmpnts.suffix = m.unit_da_suf;
        cmpnts.program := w_pgm;
    end;
    resume field cmpnts.part_num;
endif;
}

'PART NUM/LIST' =
{
/*
    DSL necessary to retrieve the parts and sort them by part number
*/
    cmpnts := select subject = c.com_nom_type + c.com_nom_num,
        part_num = c.unit_stem, suffix = c.unit_da_suf from component c;
    unloadtable cmpnts
    begin
        h_desc      := '';
        desn_enum   := '';
        qa_enum     := '';
        pa_code     := '';
        desn_org    := '';
        qa_name     := '';
        qa_org      := '';
        w_pgm       := '';
        parts := select h_desc = u_d.unit_desc from unit_desc u_d
            where :cmpnts.part_num = u_d.unit_stem
            and :cmpnts.suffix = u_d.unit_da_suf;
        cmpnts.description := h_desc;
        parts := select desn_enum = u.da_desn_engr, qa_enum = u.da_qa_engr,
            pa_code = u.prp_mfg_code from unit u
            where :cmpnts.part_num = u.unit_stem
            and :cmpnts.suffix = u.unit_da_suf;
        cmpnts.prod_agency := pa_code;
        if desn_enum != '' then
            parts := select desn_org = p.asnd_orgn from person p
                where :desn_enum = p.id_num;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                desn_org := :desn_enum;
            endif;
            cmpnts.desn_orgn := desn_org;
        endif;
        if qa_enum != '' then
            parts := select qa_name = p.last_name, qa_org = p.asnd_orgn
                from person p where :qa_enum = p.id_num;
            cmpnts.qa_engr_name := qa_name;
            cmpnts.qa_orgn := qa_org;
        endif;
        parts := select w_pgm = m.wpn_pgm_code + '-' + m.wpn_mod_num
            from mfg_inst m where :cmpnts.part_num = m.unit_stem
            and :cmpnts.suffix = m.unit_da_suf;
        cmpnts.program := w_pgm;
    end;
    resume field cmpnts.part_num;
}

'ORDER by SUBJECT' =
{

```

```

/*
DSL necessary to retrieve the parts of a certain type
*/
cmpnts := select distinct subject = c.com_nom_type + c.com_nom_num,
    part_num = c.unit_stem, suffix = c.unit_da_suf from component c
    order by subject asc, part_num asc, suffix asc;
unloadtable cmpnts
begin
    h_desc      := '';
    desn_enum   := '';
    qa_enum     := '';
    pa_code     := '';
    desn_org    := '';
    qa_name     := '';
    qa_org      := '';
    w_pgm       := '';
    parts := select h_desc = u_d.unit_desc from unit_desc u_d
        where :cmpnts.part_num = u_d.unit_stem
            and :cmpnts.suffix = u_d.unit_da_suf;
    cmpnts.description := h_desc;
    parts := select desn_enum = u.da_desn_engr, qa_enum = u.da_qa_engr,
        pa_code = u.prd_mfg_code from unit u
        where :cmpnts.part_num = u.unit_stem
            and :cmpnts.suffix = u.unit_da_suf;
    cmpnts.prod_agency := pa_code;
    if desn_enum != '' then
        parts := select desn_org = p.asnd_orgn from person p
            where :desn_enum = p.id_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            desn_org := :desn_enum;
        endif;
        cmpnts.desn_orgn := desn_org;
    endif;
    if qa_enum != '' then
        parts := select qa_name = p.last_name, qa_org = p.asnd_orgn
            from person p where :qa_enum = p.id_num;
        cmpnts.qa_engr_name := qa_name;
        cmpnts.qa_orgn      := qa_org;
    endif;
    parts := select w_pgm = m.wpn_pgm_code + '-' + m.wpn_mod_num
        from mfg_inst m where :cmpnts.part_num = m.unit_stem
            and :cmpnts.suffix = m.unit_da_suf;
    cmpnts.program := w_pgm;
end;
resume field cmpnts.part_num;
}

'WPN PGM/LIST' =
{
/*
DSL necessary to retrieve the parts in a certain weapon program
*/
cmpnts := select distinct part_num = mi.unit_stem, suffix = mi.unit_da_suf
    from mfg_inst mi where :wpn_pgm = mi.wpn_pgm_code
        and :mod_num = mi.wpn_mod_num order by part_num asc, suffix asc;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
    answer := prompt 'There are no components in this weapon program in ' +
        'PREMIS. Hit <CR> to continue';

```

```

    resume field wpn_pgm;
else
    unloadtable cmpnts
    begin
        h_desc      := '';
        h_nomen     := '';
        desn_enum   := '';
        qa_enum     := '';
        pa_code     := '';
        desn_org    := '';
        qa_name     := '';
        qa_org      := '';
        cmpnts.program := wpn_pgm + '-' + mod_num;
        parts := select h_nomen = c.com_nom_type + c.com_nom_num
            from component c where :cmpnts.part_num = c.unit_stem
            and :cmpnts.suffix = c.unit_da_suf;
        cmpnts.subject := h_nomen;
        parts := select h_desc = u_d.unit_desc from unit_desc u_d
            where :cmpnts.part_num = u_d.unit_stem
            and :cmpnts.suffix = u_d.unit_da_suf;
        cmpnts.description := h_desc;
        parts := select desn_enum = u.da_desn_engr, qa_enum = u.da_qa_engr,
            pa_code = u.prn_mfg_code from unit u
            where :cmpnts.part_num = u.unit_stem
            and :cmpnts.suffix = u.unit_da_suf;
        cmpnts.prod_agency := pa_code;
        if desn_enum != '' then
            parts := select desn_org = p.asnd_orgn from person p
                where :desn_enum = p.id_num;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                desn_org := :desn_enum;
            endif;
            cmpnts.desn_orgn := desn_org;
        endif;
        if qa_enum != '' then
            parts := select qa_name = p.last_name, qa_org = p.asnd_orgn
                from person p where :qa_enum = p.id_num;
            cmpnts.qa_engr_name := qa_name;
            cmpnts.qa_orgn := qa_org;
        endif;
    end;
    resume field cmpnts.part_num;
endif;
}

'CLEAR' =
{
    clear field all;
    resume field type;
}

'HELP', key frskey1 =
{
    help_forms (subject = 'Retrieve parts of a certain type',
        file = 'db1:[dbaprdb.help]parts.help');
}

'XIT', key frskey3 =
{
    RETURN;
}

```

```
'STOP', key frskey2 =  
{  
    EXIT;  
}
```

/*

Procedure for PQTEAM frame
Written by Norman Stevens

PQTEAM

*/

```
Initialize (h_nomtype = vchar(2), h_nomnum = vchar(8), rcount = integer1,
  h_name = vchar(20), h_phone = vchar(13), h_orgn = vchar(26),
  w_pcode = vchar(4), w_modno = vchar(2), w_cfg = vchar(7),
  h_mfg_num = vchar(4), mfg_code = vchar(3), org_type = vchar(1),
  qa_name = vchar(25), qa_enum = vchar(6), check_enum = vchar(9),
  desn_name = vchar(25), desn_enum = vchar(6), err_count = integer1,
  desn_code = vchar(3), n_sf_name = vchar(25), n_sf_enum = vchar(6),
  pa_code = vchar(3), pa_name = vchar(34), prod_id = vchar(9),
  prod_name = vchar(25), qc_name = vchar(25), qc_id = vchar(9),
  a_h_name = vchar(25), a_h_phone = vchar(14), a_h_enum = vchar(9),
  h_eqp_num = vchar(4), h_equip_num = vchar(4), h_equip_type = vchar(2),
  ad_hoc.h_wpn_pgm = vchar(4), ad_hoc.h_mod_num = vchar(2),
  ad_hoc.h_mfg_code = vchar(3), ad_hoc.member_name = vchar(20),
  ad_hoc.member_enum = vchar(9)) =
```

```
{
h_nomtype := ' ';
h_nomnum := ' ';
resume field nomenclature;
}
```

'RETRIEVE' =

{
/*

DSL to retrieve part number and suffix from PREMIS

*/

```
err_count := 0;
if nomenclature = '' then
  if part_num = '' then
    message 'You must enter a part number or a nomenclature';
    sleep 5;
  else
    pqteam := select h_nomtype = c.com_nom_type, h_nomnum = c.com_nom_num
      from component c where c.unit_stem = :part_num and
        c.unit_da_suf = :sffx;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      pqteam := select w_pcode = w.wpn_pgm_code,
        w_cfg = w.wpn_test_cfg, w_modno = w.wpn_mod_num from weapon w
        where w.unit_stem = :part_num and w.unit_da_suf = :sffx;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        message 'No component exist with that part number';
        sleep 5;
      else
        if w_cfg = '' then
          nomenclature := w_pcode + '-' + w_modno;
        else
          nomenclature := w_pcode + '/' + w_cfg + '-' + w_modno;
        endif;
      endif;
    else
      nomenclature := :h_nomtype + h_nomnum;
    endif;
  endif;
else
  pqteam := select part_num = c.unit_stem, sffx = c.unit_da_suf
```



```

        from component c where c.com_nom_type = left(:nomenclature, 2) and
        c.com_nom_num = shift(:nomenclature, -2);
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        pqteam := select part_num = c.unit_stem, sffx = c.unit_da_suf
        from component c where c.com_nom_type = left(:nomenclature, 1)
        and c.com_nom_num = shift(:nomenclature, -2);
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            err_count = 1;
            message 'No part number exist with that nomenclature';
            sleep 3;
        endif;
    endif;
endif;
/*
    DSL to retrieve the weapon program if it is not specified from PREMIS
*/
if err_count = 0 then
    h_mfg_num := '';
    if wpn_pgm = '' then
        if supplier = '' then
            ad_hoc := select h_wpn_pgm = m.wpn_pgm_code, h_mod_num = m.wpn_mod_num,
            h_mfg_code = m.mfg_code from mfg_inst m
            where m.unit_stem = :part_num and m.unit_da_suf = :sffx;
            inquire_ingres(rcount=rowcount);
            if rcount > 1 then
                mfg_code := '';
                err_count := 1;
                message 'This Part is related to more than one weapon and/or supplier'
                + ' . Be more specific';
                sleep 5;
                resume field wpn_pgm;
            elseif rcount = 1 then
                wpn_pgm := ad_hoc[1].h_wpn_pgm;
                mod_num := ad_hoc[1].h_mod_num;
                mfg_code := ad_hoc[1].h_mfg_code;
                pqteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
                where :part_num = m.unit_stem and :sffx = m.unit_da_suf
                and :wpn_pgm = m.wpn_pgm_code and :mod_num = m.wpn_mod_num
                and :mfg_code = m.mfg_code;
            else
                message 'This relationship does not exist in PREMIS. Use ADD/REV' +
                'ISE to add it';
                sleep 5;
                pqteam := select h_mfg_num = vchar(int4(max(mi.mfg_inst_num)) + 1)
                from mfg_inst mi;
            endif;
        else
            pqteam := select mfg_code = c.mfg_code from company c
            where :supplier = c.mfg_code or :supplier = c.company_name;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                pqteam := select mfg_code = c_s.mfg_code from company_syn c_s
                where :supplier = c_s.company_name;
                inquire_ingres(rcount = rowcount);
                if rcount = 0 then
                    mfg_code := '';
                    err_count := 1;
                    message ' The supplier you have entered is invalid';
                endif;
            endif;
        endif;
    endif;
endif;

```

```

        sleep 5;
    endif;
endif;
if mfg_code != '' then
    ad_hoc := select h_wpn_pgm = m.wpn_pgm_code,
        h_mod_num = m.wpn_mod_num from mfg_inst m
        where m.unit_stem = :part_num and m.unit_da_suf = :sffx
        and m.mfg_code = :mfg_code;
    inquire_ingres(rcount = rowcount);
    if rcount > 1 then
        mfg_code := '';
        err_count := 1;
        message 'This part is related to more than one weapon.' +
            ' Please specify weapon program';
        sleep 5;
        resume field wpn_pgm;
    elseif rcount = 1 then
        wpn_pgm := ad_hoc[1].h_wpn_pgm;
        mod_num := ad_hoc[1].h_mod_num;
        pqteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
            where :part_num = m.unit_stem and :sffx = m.unit_da_suf
            and :wpn_pgm = m.wpn_pgm_code and :mod_num = m.wpn_mod_num
            and :mfg_code = m.mfg_code;
    else
        message 'This relationship does not exist in PREMIS. Use ADD' +
            '/REVISE to add it';
        sleep 5;
        pqteam := select h_mfg_num = vchar(int4(max(mi.mfg_inst_num))+1)
            from mfg_inst mi;
    endif;
endif;
endif;
if mfg_code != '' then
    pqteam := select supplier = c.company_name from company c
        where :mfg_code = c.mfg_code;
endif;
else
    if supplier = '' then
        ad_hoc := select h_mfg_code = m.mfg_code from mfg_inst m
            where m.unit_stem = :part_num and m.unit_da_suf = :sffx
            and m.wpn_pgm_code = :wpn_pgm and m.wpn_mod_num = :mod_num;
        inquire_ingres(rcount = rowcount);
        if rcount > 1 then
            mfg_code := '';
            err_count := 1;
            message 'This unit is related to more than one supplier.'
                + ' Please specify supplier.';
            sleep 5;
            resume field supplier;
        elseif rcount = 1 then
            mfg_code := ad_hoc[1].h_mfg_code;
            pqteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
                where :part_num = m.unit_stem and :sffx = m.unit_da_suf
                and :wpn_pgm = m.wpn_pgm_code and :mod_num = m.wpn_mod_num
                and :mfg_code = m.mfg_code;
        else
            message 'This relationship does not exist in PREMIS. Use ADD/REV' +
                'ISE to add it';
            sleep 5;
            pqteam := select h_mfg_num = vchar(int4(max(mi.mfg_inst_num)) + 1)

```

```

        from mfg_inst mi;
    endif;
else
    pqteam := select mfg_code = c.mfg_code from company c
        where :supplier = c.mfg_code or :supplier = c.company_name;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        pqteam := select mfg_code = c_s.mfg_code from company_syn c_s
            where :supplier = c_s.company_name;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            err_count := 1;
            mfg_code := '';
            message 'The supplier you have input is invalid';
            sleep 5;
        endif;
    endif;
    if mfg_code != '' then
        pqteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
            where :part_num = m.unit_stem and :sffx = m.unit_da_suf
                and :wpn_pgm = m.wpn_pgm_code and :mod_num = m.wpn_mod_num
                and :mfg_code = m.mfg_code;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            message 'This relationship does not exist in PREMIS. Use ADD/' +
                'REVISE to add it';
            sleep 5;
            pqteam := select h_mfg_num = vchar(int4(max(mi.mfg_inst_num))+1)
                from mfg_inst mi;
        endif;
    endif;
endif;
if mfg_code != '' then
    pqteam := select supplier = c.company_name from company c
        where :mfg_code = c.mfg_code;
endif;
endif;
if h_mfg_num != '' then
    if eqp_num = '' then
        h_eqp_num := '';
    else
        pqteam := select h_equip_type = left(:eqp_num, 2),
            h_equip_num = shift(:eqp_num, -2) from equipment;
        pqteam := select h_eqp_num = e.equip_type from equipment e
            where :eqp_num = e.equip_type + e.equip_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            insert into equipment(equip_type, equip_num)
                values (:h_equip_type, :h_equip_num);
        endif;
        pqteam := select h_eqp_num = mei.eqp_inst_num from mfg_eqp_inst mei
            where :h_mfg_num = mei.mfg_inst_num
                and :eqp_num = mei.equip_type + mei.equip_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            pqteam := select
                h_eqp_num = vchar(int4(max(mei.eqp_inst_num)) + 1)
                from mfg_eqp_inst mei;
            message 'This Manufacturing Equipment relationship does not '
                + 'exist. Use ADD/REVISE to add it';
        endif;
    endif;
endif;

```

```

        sleep 5;
    endif;
endif;
endif;
endif;
/*
    OSL to retrieve the unit description from PREMIS
*/
if err_count = 0 then
    pqteam := select description = u.unit_desc from unit_desc u
        where u.unit_stem = :part_num and u.unit_da_suf = :sffx;
/*
    OSL to retrieve the product qualification team from PREMIS
*/
    pqteam := select qa_engr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, qa_orgn = p.asnd_orgn, qa_enum = u.da_qa_engr,
        qa_phone = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.da_qa_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    if qa_phone = '()-' then
        qa_phone := '';
    endif;
    qa_name := qa_engr;
    pqteam := select desn_engr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, desn_orgn = p.asnd_orgn, desn_enum = u.da_desn_engr,
        desn_phone = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.da_desn_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    if desn_phone = '()-' then
        desn_phone := '';
    endif;
    desn_name := desn_engr;
    pqteam := select nuc_sfty_egr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, nuc_sfty_org = p.asnd_orgn, n_sf_enum = u.da_n_sf_engr,
        nuc_sfty_phn = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.da_n_sf_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    if nuc_sfty_phn = '()-' then
        nuc_sfty_phn := '';
    endif;
    n_sf_name := nuc_sfty_egr;
    pqteam := select prod_agency = c.company_name, pa_name = c.company_name,
        pa_code = c.mfg_code from company c, unit u
        where (c.mfg_code = u.prn_mfg_code) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    pqteam := select prod_engr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, prod_id = p.id_num,
        prod_phone = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.pa_prod_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    prod_name := prod_engr;
    if prod_phone = '()-' then

```

```

    prod_phone := '';
endif;
pqteam := select qc_engr = p.first_name + ' ' + p.middl_init + ' ' +
    p.last_name, qc_id = p.id_num,
    qc_phone = '(' + left(p.phone_num, 3) + ')' +
        left(shift(p.phone_num, -3), 3) + '-' +
        shift(p.phone_num, -6)
    from person p, unit u where (p.id_num = u.pa_qc_engr) and
        (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
qc_name := qc_engr;
if qc_phone = '()-' then
    qc_phone := '';
endif;
ad_hoc[1].category := 'AD HOC MEMBERS';
ad_hoc[1].engr_name := '';
if h_mfg_num != '' then
    if h_eqp_num = '' then
        ad_hoc := select engr_name = p.first_name + ' ' + p.middl_init + ' ' +
            p.last_name, phone_num = '(' + left(p.phone_num, 3) + ')' +
                left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6),
            member_enum = p.id_num, orgn = p.asnd_orgn,
            category = t.member_title from person p, tms_pq_team t
            where :h_mfg_num = t.mfg_inst_num and t.id_num = p.id_num;
    else
        ad_hoc := select engr_name = p.first_name + ' ' + p.middl_init + ' ' +
            p.last_name, phone_num = '(' + left(p.phone_num, 3) + ')' +
                left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6),
            member_enum = p.id_num, orgn = p.asnd_orgn,
            category = e.member_title from person p, eq_pq_team e
            where :h_eqp_num = e.eqp_inst_num and e.id_num = p.id_num;
    endif;
endif;
unloadtable ad_hoc
begin
    if ad_hoc.phone_num = '()-' then
        ad_hoc.phone_num := '';
    endif;
    ad_hoc.member_name := :ad_hoc.engr_name;
    h_name := ad_hoc.engr_name;
    h_orgn := ad_hoc.orgn;
    pqteam := select h_nomtype = left(:ad_hoc.orgn, 1) from person;
    if h_nomtype != '0' then
        pqteam := select h_orgn = c.company_name from person p, company c
            where :h_name = p.first_name + ' ' + p.middl_init + ' ' +
                p.last_name and p.mfg_code = c.mfg_code;
    endif;
    ad_hoc.engr_name := h_name;
    ad_hoc.orgn := h_orgn;
end;
endif;
}

'ADD/REVISE' =
{
/*
    DSL to ensure that the user has entered a part number
*/
    if part_num = '' then
        if nomenclature = '' then
            message 'You must enter a part number or a nomenclature';

```

```

    sleep 5;
else
    pqteam := select part_num = c.unit_stem, sffx = c.unit_da_suf
    from component c where c.com_nom_type = left(:nomenclature, 2) and
    c.com_nom_num = shift(:nomenclature, -2);
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        pqteam := select part_num = c.unit_stem, sffx = c.unit_da_suf
        from component c where c.com_nom_type = left(:nomenclature, 1)
        and c.com_nom_num = shift(:nomenclature, -2);
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            message 'No part number exist with that nomenclature';
            sleep 5;
        endif;
    endif;
endif;
else
    pqteam := select h_nomtype = c.com_nom_type, h_nomnum = c.com_nom_num
    from component c where c.unit_stem = :part_num and
    c.unit_da_suf = :sffx;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        pqteam := select w_pcode = w.wpn_pgm_code,
        w_cfg = w.wpn_test_cfg, w_modno = w.wpn_mod_num from weapon w
        where w.unit_stem = :part_num and w.unit_da_suf = :sffx;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            message 'No component exist with that part number';
            sleep 5;
        else
            if w_cfg = '' then
                nomenclature := w_pcode + '-' + w_modno;
            else
                nomenclature := w_pcode + '/' + w_cfg + '-' + w_modno;
            endif;
        endif;
    else
        nomenclature := :h_nomtype + h_nomnum;
    endif;
endif;
if part_num != '' then
    err_count := 0;
/*
DSL necessary to edit the weapon program
*/
    pqteam := select w_pcode = wp.wpn_pgm_code from weapon_pgm wp
    where :wpn_pgm = wp.wpn_pgm_code and :mod_num = wp.wpn_mod_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        err_count := err_count + 1;
        message 'The weapon program is invalid. Call your DBA';
        sleep 5;
    endif;
/*
DSL necessary to edit the supplier
*/
    if supplier = '' then
        mfg_code := '';
    else

```

```

pqteam := select mfg_code = c.mfg_code from company c
        where :supplier = c.mfg_code or :supplier = c.company_name;
inquire_ingres (rcount = rowcount);
if rcount = 0 then
    pqteam := select mfg_code = c_s.mfg_code from company_syn c_s
        where :supplier = c_s.company_name;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        err_count := err_count + 1;
        message 'The supplier is INVALID';
        sleep 5;
    else
        pqteam := select supplier = c.company_name from company c
            where :mfg_code = c.mfg_code;
    endif;
endif;
endif;

/*
DSL necessary to edit the quality assurance engineer data
*/
if qa_name != qa_engr then
    if qa_engr = '' then
        qa_enum = '';
    else
        pqteam := select qa_enum = p.id_num from person p
            where p.last_name = :qa_engr and p.asnd_orgn = :qa_orgn;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            err_count = err_count + 1;
            message 'No one named ' + qa_engr + ' works for ' + qa_orgn;
            sleep 5;
        else
            pqteam := select check_enum = q.empl_num from da_qa_engr q
                where :qa_enum = q.empl_num;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                err_count := err_count + 1;
                message 'The QA Engineer is INVALID';
                sleep 5;
            else
                check_enum := '';
                pqteam := select qa_engr = p.first_name + ' ' + p.middl_init
                    + ' ' + p.last_name, qa_phone = '(' + left(p.phone_num, 3)
                    + ')' + left(shift(p.phone_num, -3), 3) + '-' +
                    shift (p.phone_num, -6) from person p
                    where :qa_enum = p.id_num;
                qa_name := qa_engr;
            endif;
        endif;
    endif;
endif;

/*
DSL necessary to edit the design engineer data
*/
if desn_name != desn_engr then
    if desn_engr = '' then
        desn_enum := '';
        desn_orgn := '';
    else

```

```

pqteam := select rcount = count (*) from person p
        where :desn_orgn = p.asnd_orgn and :desn_engr = p.last_name;
if rcount = 0 then
    pqteam := select rcount = count (*) from person p
              where :desn_orgn = p.asnd_orgn
                  and :desn_engr = left(p.first_name, 1) + " " +
                      p.middl_init + " " + p.last_name;
    if rcount = 0 then
        err_count := err_count + 1;
        message 'No one named ' + desn_engr + ' works for ' +
            desn_orgn;
        sleep 5;
    elseif rcount = 1 then
        pqteam := select desn_enum = p.id_num from person p
                  where :desn_orgn = p.asnd_orgn
                      and :desn_engr = left(p.first_name, 1) + " " +
                          p.middl_init + " " + p.last_name;
    else
        err_count := err_count + 1;
        message 'More than one engr named ' + desn_engr + ' works ' +
            'for ' + desn_orgn + '. Please call your DBA.';
        sleep 5;
    endif;
elseif rcount = 1 then
    pqteam := select desn_enum = p.id_num from person p
              where :desn_orgn = p.asnd_orgn and :desn_engr = p.last_name;
else
    err_count := err_count + 1;
    message 'More than one engr named ' + desn_engr + ' works for ' +
        desn_orgn + '. Please enter the initials.';
    sleep 5;
endif;
if rcount = 1 then
    pqteam := select check_enum = d.empl_num from da_desn_engr d
              where :desn_enum = d.empl_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        err_count := err_count + 1;
        message 'The Design Engineer is INVALID';
        sleep 5;
    else
        desn_enum = :check_enum;
        check_enum = '';
        pqteam := select desn_engr = p.first_name + ' ' + p.middl_init +
            ' ' + p.last_name, desn_phone = '(' + left(p.phone_num, 3) + ')'
            + left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6),
            desn_code = p.mfg_code from person p
              where :desn_enum = p.id_num;
        desn_name := desn_engr;
        if desn_phone = '()-' then
            desn_phone = '';
        endif;
    endif;
endif;
endif;
endif;
endif;
/*
OSL necessary to edit the nuclear safety engineer data
*/
if n_sf_name != :nuc_sfty_egr then

```



```

if nuc_sfty_egr = '' then
    n_sf_enum := '';
else
    pqteam := select n_sf_enum = p.id_num from person p
              where p.last_name = :nuc_sfty_egr
              and p.asnd_orgn = :nuc_sfty_org;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        err_count := err_count + 1;
        message 'No one named ' + nuc_sfty_egr + ' works for ' +
            nuc_sfty_org;
        sleep 5;
    else
        pqteam := select nuc_sfty_egr = p.first_name+' ' + p.middl_init +
            ' ' + p.last_name, nuc_sfty_phn = '('+ left(p.phone_num, 3) +
            ')+left(shift(p.phone_num,-3),3)+'-' + shift(p.phone_num, -6)
            from person p where :n_sf_enum = p.id_num;
        n_sf_name := nuc_sfty_egr;
    endif;
endif;
endif;
/*
DSL necessary to edit the production agency
*/
if pa_name != prod_agency then
    if prod_agency = '' then
        pa_code := '';
        pa_name := '';
    else
        pa_name := prod_agency;
        pqteam := select pa_code = c.mfg_code from company c
                  where :pa_name = c.mfg_code or :pa_name = c.company_name;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            pqteam := select pa_code = c_s.mfg_code from company_syn c_s
                      where :pa_name = c_s.company_name;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                pa_code := '';
                err_count := err_count + 1;
                message 'The Production Agency is invalid';
                sleep 5;
            endif;
        endif;
        if pa_code = '' then
            pa_name := '';
        else
            pqteam := select prod_agency = c.company_name from company c
                      where :pa_code = c.mfg_code;
            pa_name := prod_agency;
        endif;
    endif;
endif;
/*
DSL necessary to edit the product engineer data
*/
if prod_engr != :prod_name then
    if prod_engr = '' then
        prod_id := '';
    else

```

```

pqteam := select rcount = count (*) from person p
where :pa_code = p.mfg_code and :prod_engr = p.last_name;
if rcount = 0 then
  pqteam := select rcount = count (*) from person p
  where :pa_code = p.mfg_code
  and :prod_engr = left(p.first_name, 1) + " " +
    p.middl_init + " " + p.last_name;
  if rcount = 0 then
    err_count = err_count + 1;
    message 'There is no one named ' + prod_engr + ' who works ' +
      'for ' + pa_code;
    sleep 5;
  elseif rcount = 1 then
    pqteam := select prod_id = p.id_num from person p
    where :pa_code = p.mfg_code
    and :prod_engr = left(p.first_name, 1) + " " +
      p.middl_init + " " + p.last_name;
  else
    err_count = err_count + 1;
    message 'More than one engineer named ' + prod_engr +
      ' works for ' + pa_code + '. Please call your DBA';
    sleep 5;
  endif;
elseif rcount = 1 then
  pqteam := select prod_id = p.id_num from person p
  where :pa_code = p.mfg_code and :prod_engr = p.last_name;
else
  err_count := err_count + 1;
  message 'More than one engineer named ' + prod_engr + ' works ' +
    'for ' + pa_code + '. Please enter the initials.';
  sleep 5;
endif;
if rcount = 1 then
  pqteam := select check_enum = p.id_num from pa_prod_engr p
  where p.id_num = prod_id;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    err_count = err_count + 1;
    message 'This engineer is not a valid product engineer.';
    sleep 5;
  else
    pqteam := select prod_engr = p.first_name + ' ' + p.middl_init +
      ' ' + p.last_name, prod_phone = '(' + left(p.phone_num, 3) +
      ')' + left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
    from person p where :prod_id = p.id_num;
    prod_name := prod_engr;
    if prod_phone = '()-' then
      prod_phone := '';
    endif;
  endif;
endif;
endif;
endif;
endif;
/*
OSL necessary to edit the quality control engineer data
*/
if qc_engr != :qc_name then
  if qc_engr = '' then
    qc_id := '';
  else

```

```

pqteam := select rcount = count (*) from person p
where :pa_code = p.mfg_code and :qc_engr = p.last_name;
if rcount = 0 then
  pqteam := select rcount = count (*) from person p
  where :pa_code = p.mfg_code
  and :qc_engr = left(p.first_name, 1) + " " +
    p.middl_init + " " + p.last_name;
  if rcount = 0 then
    err_count = err_count + 1;
    message 'There is no one named ' + qc_engr + ' who works ' +
      'for ' + pa_code + '. Hit <CR> to continue.';
    sleep 5;
  elseif rcount = 1 then
    pqteam := select check_enum = p.id_num from person p
    where :pa_code = p.mfg_code
    and :qc_engr = left(p.first_name, 1) + " " +
      p.middl_init + " " + p.last_name;
  else
    err_count = err_count + 1;
    message 'More than one engineer named ' + qc_engr +
      ' works for ' + pa_code + '. Please call your DBA';
    sleep 5;
  endif;
elseif rcount = 1 then
  pqteam := select check_enum = p.id_num from person p
  where :pa_code = p.mfg_code and :qc_engr = p.last_name;
else
  err_count := err_count + 1;
  message 'More than one engineer named ' + qc_engr + ' works ' +
    'for ' + pa_code + '. Please enter the initials.';
  sleep 5;
endif;
if rcount = 1 then
  pqteam := select qc_id = p.id_num from pa_qc_engr p
  where p.id_num = :check_enum;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    err_count = err_count + 1;
    message 'This engineer is not a valid quality control ' +
      'engineer.';
    sleep 5;
  else
    pqteam := select qc_engr = p.first_name + ' ' + p.middl_init +
      ' ' + p.last_name, qc_phone = '(' + left(p.phone_num, 3) +
      ')' + left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
    from person p where :qc_id = p.id_num;
    qc_name := qc_engr;
    if qc_phone = '()' then
      qc_phone := '';
    endif;
  endif;
endif;
endif;
endif;
endif;
/*
OSL necessary to edit ad hoc members
*/
if h_mfg_num = '' then
  message 'You should retrieve data before you try to revise it';
  sleep 5;

```



```

        ad_hoc.category := 'INVALID ORGANIZATION';
    else
        pqteam = select a_h_name = p.first_name + ' ' + p.middl_init
            + ' ' + p.last_name, a_h_phone = '(' +
            left(p.phone_num,3)+')' + left(shift(p.phone_num,-3), 3)
            + '-' + shift(p.phone_num, -6), a_h_enum = p.id_num,
            h_orgn = c.company_name from person p, company c
            where :ad_hoc.egr_name = p.last_name
                and :mfg_code = p.mfg_code;
        if a_h_phone = '()-' then
            a_h_phone := '';
        endif;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            a_h_enum := '';
            ad_hoc.category = 'INVALID MEMBER';
            message :ad_hoc.egr_name + ' is not in PREMIS';
            sleep 5;
        endif;
    endif;
endif;
if a_h_enum != '' then
    ad_hoc.egr_name := a_h_name;
    ad_hoc.phone_num := a_h_phone;
    ad_hoc.orgn := h_orgn;
    if h_eqp_num = '' then
        pqteam := select check_enum = t.id_num from tms_pq_team t
            where :h_mfg_num = t.mfg_inst_num
                and :a_h_enum = t.id_num;
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
            insert into tms_pq_team (id_num, hrd_sft_flag,
                mfg_inst_num, member_title) values (:a_h_enum, 'H',
                    :h_mfg_num, :ad_hoc.category);
        else
            update tms_pq_team t
                set member_title = :ad_hoc.category
                where :h_mfg_num = t.mfg_inst_num
                    and :a_h_enum = t.id_num;
        endif;
    else
        pqteam := select check_enum = e.id_num from eq_pq_team e
            where :h_eqp_num = e.eqp_inst_num
                and :a_h_enum = e.id_num;
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
            insert into eq_pq_team (id_num, hrd_sft_flag,
                eqp_inst_num, member_title) values (:a_h_enum, 'H',
                    :h_eqp_num, :ad_hoc.category);
        else
            update eq_pq_team e
                set member_title = :ad_hoc.category
                where :h_eqp_num = e.eqp_inst_num
                    and :a_h_enum = e.id_num;
        endif;
    endif;
endif;
endif;
endif;
end;

```

```

endif;
/*
OSL necessary to update the unit record
*/
if err_count = 0 then
  pqteam := select check_enum = u.da_qa_engr from unit u
    where :part_num = u.unit_stem and :sffx = u.unit_da_suf;
  inquire_ingres(rcount= rowcount);
  if rcount = 0 then
    insert into unit (unit_stem, unit_da_suf, da_qa_engr, da_desn_engr,
      dsn_mfg_code, da_n_sf_engr, prd_mfg_code, pa_prod_engr,
      pa_qc_engr) values (:part_num, :sffx, :qa_enum, :desn_enum,
        :desn_code, :n_sf_enum, :pa_code, :prod_id, :qc_id);
    message 'Part record was added';
    sleep 5;
  else
    update unit u set da_qa_engr = :qa_enum, da_desn_engr = :desn_enum,
      dsn_mfg_code = :desn_code, da_n_sf_engr = :n_sf_enum,
      prd_mfg_code = :pa_code, pa_prod_engr = :prod_id,
      pa_qc_engr = :qc_id
      where :part_num = u.unit_stem and :sffx = u.unit_da_suf;
    message 'Part record was revised';
    sleep 5;
  endif;
  pqteam := select check_enum = mi.mfg_inst_num from mfg_inst mi
    where :h_mfg_num = mi.mfg_inst_num;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    insert into mfg_inst(mfg_inst_num, mfg_code, wpn_pgm_code,
      wpn_mod_num, unit_stem, unit_da_suf) values (:h_mfg_num,
        :mfg_code, :wpn_pgm, :mod_num, :part_num, :sffx);
  else
    update mfg_inst mi set mfg_code = :mfg_code,
      wpn_pgm_code = :wpn_pgm, wpn_mod_num = :mod_num,
      unit_stem = :part_num, unit_da_suf = :sffx
      where :h_mfg_num = mi.mfg_inst_num;
  endif;
  if h_eqp_num != '' then
    pqteam := select check_enum = mei.eqp_inst_num
      from mfg_eqp_inst mei where :h_eqp_num = mei.eqp_inst_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      insert into mfg_eqp_inst(eq_p_inst_num, mfg_inst_num,
        equip_type, equip_num) values (:h_eqp_num, :h_mfg_num,
          :h_equip_type, :h_equip_num);
    else
      update mfg_eqp_inst mei set mfg_inst_num = :h_mfg_num,
        equip_type = :h_equip_type, equip_num = :h_equip_num
        where :h_eqp_num = mei.eqp_inst_num;
    endif;
  endif;
  qa_enum := '';
  desn_enum := '';
  desn_code := '';
  n_sf_enum := '';
  pa_code := '';
  prod_id := '';
  qc_id := '';
endif;
endif;

```

```

}
'CLEAR' =
{
    clear field all;
    resume field nomenclature;
}
'HELP', key frskey1 =
{
    help_forms (subject = 'Product Qual Team Retrievals',
        file = 'db1:[dbaprdb.help]pqteam.help');
}
'XIT', key frskey3 =
{
    RETURN;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

/*

Procedure for PQTTEAM frame
Written by Norman Stevens

PQTTEAM

*/

```
Initialize (h_nomtype = vchar(2), h_nomnum = vchar(8), rcount = integer1,
            h_name = vchar(20), h_phone = vchar(13), h_orgn = vchar(26),
            w_pcode = vchar(4), w_modno = vchar(2), w_cfg = vchar(7),
            err_count = integer1, h_mfg_num = vchar(4), mfg_code = vchar(3),
            ad_hoc.h_wpn_pgm = vchar(4), ad_hoc.h_mod_num = vchar(2),
            ad_hoc.h_mfg_code = vchar(4)) =
```

```
{
h_nomtype := ' ';
h_nomnum := ' ';
resume field nomenclature;
}
```

'RETRIEVE' =

{
/*

DSL to retrieve part number and suffix from PREMIS

*/

```
if nomenclature = '' then
  if part_num = '' then
    message 'You must enter a part number or a nomenclature';
    sleep 5;
  else
    pqteam := select h_nomtype = c.com_nom_type, h_nomnum = c.com_nom_num
      from component c where c.unit_stem = :part_num and
        c.unit_da_suf = :sffx;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      pqteam := select w_pcode = w.wpn_pgm_code,
        w_cfg = w.wpn_test_cfg, w_modno = w.wpn_mod_num from weapon w
        where w.unit_stem = :part_num and w.unit_da_suf = :sffx;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        message 'No component exist with that part number';
        sleep 5;
      else
        if w_cfg = '' then
          nomenclature := w_pcode + '-' + w_modno;
        else
          nomenclature := w_pcode + '/' + w_cfg + '-' + w_modno;
        endif;
      endif;
    else
      nomenclature := :h_nomtype + h_nomnum;
    endif;
  endif;
else
  pqteam := select part_num = c.unit_stem, sffx = c.unit_da_suf
    from component c where c.com_nom_type = left(:nomenclature, 2) and
      c.com_nom_num = shift(:nomenclature, -2);
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    pqteam := select part_num = c.unit_stem, sffx = c.unit_da_suf
      from component c where c.com_nom_type = left(:nomenclature, 1)
        and c.com_nom_num = shift(:nomenclature, -2);
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
```



```

        message 'No part number exist with that nomenclature';
        sleep 5;
    endif;
endif;
endif;
/*
    DSL to retrieve the weapon program and the supplier if it is
    not specified from PREMIS
*/
err_count := 0;
if wpn_pgm = '' then
    if supplier = '' then
        ad_hoc := select h_wpn_pgm = m.wpn_pgm_code,
            h_mod_num = m.wpn_mod_num, h_mfg_code = m.mfg_code from mfg_inst m
            where m.unit_stem = :part_num and m.unit_da_suf = :sffx;
        inquire_ingres(rcount=rowcount);
        if rcount > 1 then
            mfg_code := '';
            err_count := 1;
            message 'This Part is related to more than one weapon or supplier.'
                + ' Please specify weapon program and/or supplier';
            sleep 5;
            resume field wpn_pgm;
        elseif rcount = 1 then
            wpn_pgm := ad_hoc[1].h_wpn_pgm;
            mod_num := ad_hoc[1].h_mod_num;
            mfg_code := ad_hoc[1].h_mfg_code;
            pqtteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
                where :mfg_code = m.mfg_code and :wpn_pgm = m.wpn_pgm_code
                    and :part_num = m.unit_stem and :sffx = m.unit_da_suf
                    and :mod_num = m.wpn_mod_num;
        else
            mfg_code := '';
            err_count := 1;
            message 'This Part is not related to a weapon program';
            sleep 5;
        endif;
    else
        pqtteam := select mfg_code = c.mfg_code from company c
            where :supplier = c.mfg_code or :supplier = c.company_name;
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
            pqtteam := select mfg_code = c_s.mfg_code from company_syn c_s
                where :supplier = c_s.company_name;
            inquire_ingres (rcount = rowcount);
            if rcount = 0 then
                message 'The supplier you have supplied is invalid';
                sleep 5;
                mfg_code := '';
                err_count := 1;
            endif;
        endif;
    endif;
    if mfg_code != '' then
        ad_hoc := select h_wpn_pgm = m.wpn_pgm_code,
            h_mod_num = m.wpn_mod_num from mfg_inst m
            where m.unit_stem = :part_num and m.unit_da_suf = :sffx
                and m.mfg_code = :mfg_code;
        inquire_ingres(rcount=rowcount);
        if rcount > 1 then
            mfg_code := '';

```

```

err_count := 1;
message 'This Part is related to more than one weapon.' +
  ' Please specify weapon program';
sleep 5;
resume field wpn_pgm;
elseif rcount = 1 then
  wpn_pgm := ad_hoc[1].h_wpn_pgm;
  mod_num := ad_hoc[1].h_mod_num;
  pqtteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
    where :mfg_code = m.mfg_code and :wpn_pgm = m.wpn_pgm_code
      and :part_num = m.unit_stem and :sffx = m.unit_da_suf
      and :mod_num = m.wpn_mod_num;
else
  mfg_code := '';
  err_count := 1;
  message 'This Part is not related to the specified supplier';
  sleep 5;
endif;
endif;
endif;
if mfg_code != '' then
  pqtteam := select supplier = c.company_name from company c
    where :mfg_code = c.mfg_code;
endif;
else
  if supplier = '' then
    ad_hoc := select h_mfg_code = m.mfg_code from mfg_inst m
      where m.unit_stem = :part_num and m.unit_da_suf = :sffx
        and m.wpn_pgm_code = :wpn_pgm and m.wpn_mod_num = :mod_num;
    inquire_ingres(rcount=rowcount);
    if rcount > 1 then
      mfg_code := '';
      err_count := 1;
      message 'This Part is related to more than one supplier.'
        + ' Please specify supplier';
      sleep 5;
      resume field supplier;
    elseif rcount = 1 then
      mfg_code := ad_hoc[1].h_mfg_code;
      pqtteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
        where :mfg_code = m.mfg_code and :wpn_pgm = m.wpn_pgm_code
          and :part_num = m.unit_stem and :sffx = m.unit_da_suf
          and :mod_num = m.wpn_mod_num;
    else
      mfg_code := '';
      err_count := 1;
      message 'This Part is not related to the specified weapon' +
        ' program';
      sleep 5;
    endif;
  else
    pqtteam := select mfg_code = c.mfg_code from company c
      where :supplier = c.mfg_code or :supplier = c.company_name;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
      pqtteam := select mfg_code = c_s.mfg_code from company_syn c_s
        where :supplier = c_s.company_name;
      inquire_ingres (rcount = rowcount);
      if rcount = 0 then
        message 'The supplier you have supplied is invalid';
      end if;
    end if;
  end if;
endif;

```

```

        sleep 5;
        mfg_code := '';
        err_count := 1;
    endif;
endif;
if mfg_code != '' then
    pqtteam := select h_mfg_num = m.mfg_inst_num from mfg_inst m
        where :part_num = m.unit_stem and :sffx = m.unit_da_suf
        and :wpn_pgm = m.wpn_pgm_code and :mod_num = m.wpn_mod_num
        and :mfg_code = m.mfg_code;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        err_count := 1;
        message 'The combination of weapon program, supplier and part '
            + 'number is invalid';
        sleep 5;
    endif;
endif;
endif;
if mfg_code != '' then
    pqtteam := select supplier = c.company_name from company c
        where :mfg_code = c.mfg_code;
endif;
endif;
/*
    DSL to retrieve the unit description from PREMIS
*/
if err_count = 0 then
    pqtteam := select description = u.unit_desc from unit_desc u
        where u.unit_stem = :part_num and u.unit_da_suf = :sffx;
/*
    DSL to retrieve the product qualification team from PREMIS
*/
    pqtteam := select qa_engr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, qa_orgn = p.asnd_orgn,
        qa_phone = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.da_qa_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    if qa_phone = '()-' then
        qa_phone := '';
    endif;
    pqtteam := select desn_engr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, desn_orgn = p.asnd_orgn,
        desn_phone = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.da_desn_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
    if desn_phone = '()-' then
        desn_phone := '';
    endif;
    pqtteam := select nuc_sfty_egr = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name, nuc_sfty_org = p.asnd_orgn,
        nuc_sfty_phn = '(' + left(p.phone_num, 3) + ')' +
            left(shift(p.phone_num, -3), 3) + '-' +
            shift(p.phone_num, -6)
        from person p, unit u where (p.id_num = u.da_n_sf_engr) and
            (:part_num = u.unit_stem and :sffx = u.unit_da_suf);

```

```

if nuc_sfty_phn = '()-' then
  nuc_sfty_phn := '';
endif;
pqteam := select prod_agency = c.company_name from company c, unit u
  where (c.mfg_code = u.prn_mfg_code) and
    (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
pqteam := select prod_engr = p.first_name + ' ' + p.middl_init + ' ' +
  + p.last_name,
  prod_phone = '(' + left(p.phone_num, 3) + ')' +
    left(shift(p.phone_num, -3), 3) + '-' +
    shift(p.phone_num, -6)
  from person p, unit u where (p.id_num = u.pa_prod_engr) and
    (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
if prod_phone = '()-' then
  prod_phone := '';
endif;
pqteam := select qc_engr = p.first_name + ' ' + p.middl_init + ' ' +
  p.last_name,
  qc_phone = '(' + left(p.phone_num, 3) + ')' +
    left(shift(p.phone_num, -3), 3) + '-' +
    shift(p.phone_num, -6)
  from person p, unit u where (p.id_num = u.pa_qc_engr) and
    (:part_num = u.unit_stem and :sffx = u.unit_da_suf);
if qc_phone = '()-' then
  qc_phone := '';
endif;
ad_hoc[1].category := 'AD HOC MEMBERS';
ad_hoc[1].engr_name := '';
if h_mfg_num != '' then
  ad_hoc := select engr_name = p.first_name + ' ' + p.middl_init + ' ' +
    p.last_name, phone_num = '(' + left(p.phone_num, 3) + ')' +
    left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6),
    orgn = p.asnd_orgn, category = t.member_title from person p,
    tms_pq_team t where :h_mfg_num = t.mfg_inst_num
    and t.id_num = p.id_num;
endif;
unloadtable ad_hoc
{ if ad_hoc.phone_num = '()-' then
  ad_hoc.phone_num := '';
endif;
h_name := ad_hoc.engr_name;
h_orgn := ad_hoc.orgn;
pqteam := select h_notype = left(:ad_hoc.orgn, 1) from person;
if h_notype != 'O' then
  pqteam := select h_orgn = c.company_name from person p, company c
    where :h_name = p.first_name + ' ' + p.middl_init + ' ' +
    p.last_name and p.mfg_code = c.mfg_code;
endif;
ad_hoc.engr_name := h_name;
ad_hoc.orgn := h_orgn;
};
endif;
}

'CLEAR' =
{
  clear field all;
  resume field nomenclature;
}
'HELP', key frskey1 =

```

```
{
  help_forms (subject = 'QA PQT Team Screen',
    file = 'db1:[dbaprdb.help]pqtteam.help');
}
'XIT', key frskey3 =
{
  RETURN;
}

'STOP', key frskey2 =
{
  EXIT;
}
```

```

/*
                                frame: PROD
DRIVER FOR REPORT TO LIST ALL PARTS BY PRODUCTION AGENCY      PROD
                                written by john orman   9/21/88

*/
initialize (built_tab = vchar(3),
            old_prod_agy = vchar(15),
            param_prod = vchar(30),
            param_type = vchar(3)) = {
/*
    set flag telling whether all_id_rpt table has been built */
    built_tab = 'no' ;
    if rep_type = 'NEW' then
        param_type = 'new'
    else
        param_type = 'all'
    endif ;
/*
    initialize prod_agy from previous query */
    old_prod_agy = 'XXXXXXXXXXXXXXXX' ;

'DISPLAY' = {
/*
    if there has been a change of production agency, reset
    built_tab

*/
    if prod_agy != :old_prod_agy then
        if built_tab = 'yes' then

/*
            empty all_id_rpt table

*/
            delete from all_id_rpt ;
        endif ;
        old_prod_agy := :prod_agy ;
        built_tab := 'no' ;
    endif ;

/*
    if not already built, build table all_id_rpt
    - needed for report

*/
    if built_tab = 'no' then
        callproc build_prod(
                                prod_agy = :prod_agy,
                                rep_type = :rep_type) ;
        built_tab = 'yes' ;
    endif ;

/*
    create a string for report parameter */
    if :prod_agy = '' then
        param_prod := 'prod_agy = ALL, rep_type = ' +
                        :rep_type
    else
        param_prod := 'prod_agy = ' + :prod_agy +
                        ', rep_type = ' + :rep_type ;
    endif ;

/*
    send report on unit table to screen

*/
    call report (report = 'prod_rpt', mode = 'wrap',
                param = :param_prod) ;
}

'FILE' = {
/*
    if there has been a change of production agency, reset
    built_tab

```

```

*/
if prod_agy != :old_prod_agy then
  if built_tab = 'yes' then

    empty all_id_rpt table

    delete from all_id_rpt ;
  endif ;
  old_prod_agy := :prod_agy ;
  built_tab := 'no' ;
endif ;

/*
  if not already built, build table all_id_rpt
  - needed for report
*/
if built_tab = 'no' then
  callproc build_prod(
    prod_agy = :prod_agy,
    rep_type = :rep_type) ;
  built_tab = 'yes' ;
endif ;

/*
  create a string for report parameter */
if :prod_agy = '' then
  param_prod := 'prod_agy = ALL, rep_type = ' +
    :rep_type ;
else
  param_prod := 'prod_agy = ' + :prod_agy +
    ', rep_type = ' + :rep_type ;
endif ;

/*
  send report on unit table to file unit.rpt
*/
call report (report = 'prod_rpt', mode = 'wrap',
  file = 'prod_agy.rpt',
  param = :param_prod) ;
message 'File containing report is PROD_AGY.RPT' ;
sleep 3 ;
}

'PRINTER' =
{
/*
  if there has been a change of production agency, reset
  built_tab
*/
if prod_agy != :old_prod_agy then
  if built_tab = 'yes' then

    empty all_id_rpt table

    delete from all_id_rpt ;
  endif ;
  old_prod_agy := :prod_agy ;
  built_tab := 'no' ;
endif ;

/*
  if not already built, build table all_id_rpt
  - needed for report
*/
if built_tab = 'no' then
  callproc build_prod(
    prod_agy = :prod_agy,

```

```

                                rep_type = :rep_type) ;
        built_tab = 'yes' ;
    endif ;
/*      create a string for report parameter */
    if :prod_agy = '' then
        param_prod := 'prod_agy = ALL, rep_type = ' +
                        :rep_type ;
    else
        param_prod := 'prod_agy = ' + :prod_agy +
                        ', rep_type = ' + :rep_type ;
    endif ;

/*      create report in file prod.rpt
*/
    call report (report = 'prod_rpt', mode = 'wrap',
                 file = 'prod.rpt',
                 param = :param_prod) ;

/*      send file containing report to appropriate printer
        use special procedure print_dispose
        first parameter to procedure print_dispose if name
        of file to be printed jlo 9/21/88
*/
    call system 'QDB1:[DBAPRDB.SOURCE]print_dispose prod.rpt' ;
    message 'PROD.RPT is being printed on MYPRINTER' ;
    sleep 3 ;
}
/*
'HELP', key frskey1 =
{
    help_forms (subject = 'List all parts by Production Agency',
                 file = 'db1:[dbaprd.b.help]prod.help');
}
*/
'XIT', key frskey3 =
{
    if built_tab = 'yes' then
/*      empty all_id_rpt table
*/
        delete from all_id_rpt ;
    endif ;
    RETURN ;
}
'STOP', key frskey2 =
{
    if built_tab = 'yes' then
/*      empty all_id_rpt table
*/
        delete from all_id_rpt ;
    endif ;
    EXIT ;
}

```


/*

Procedure for QAEMAIN frame
Written by Norman Stevens

QAEMAIN

```
*/  
Initialize (answer = vchar(1)) =  
{  
}  
  
'NEW PART' =  
{  
    callframe NEWPART;  
}  
  
'C D RPT' =  
{  
    callframe CHARDEV;  
}  
  
'TMS CENTER' =  
{  
    callframe TMSCNTR;  
}  
  
'PQ TEAM' =  
{  
    callframe PQTEAM;  
}  
  
'HELP', key frskey1 =  
{  
    help_forms (subject = 'QA DATA ENTRY MENU',  
                file = 'db1:[dbaprdb.help]qaemain.help');  
}  
  
'XIT', key frskey3 =  
{  
    RETURN;  
}  
  
'STOP', key frskey2 =  
{  
    EXIT;  
}
```

/*

Procedure for QAENGR frame
Written by Norman Stevens

*/

Initialize (h_empl_num = varchar(6), h_desc = varchar(16), h_nomen = varchar(10),
 est_date = varchar(11), answer = varchar(1), rcount = integer) =

```
{  
  h_empl_num := '  
  resume field engr_name;  
}
```

'RETRIEVE' = {

```
  qaengr := select h_empl_num = p.id_num from person p  
             where :engr_name = p.last_name and :orgn = p.asnd_orgn;
```

```
  inquire_ingres(rcount=rowcount);
```

```
  if rcount = 0 then
```

```
    answer := prompt 'Engineer named ' + engr_name + ' of '  
                  + orgn + ' is not found in PREMIS. Hit <CR> to continue';
```

```
  else
```

```
    qaengr := select engr_name = p.first_name + ' ' + p.middl_init + ' ' +  
                  p.last_name from person p where p.id_num = :h_empl_num;
```

```
    assignment := select wpn_nomen = m.wpn_pgm_code + '-' + m.wpn_mod_num,  
                        part_no = m.unit_stem, sffx = m.unit_da_suf, dcmt_num = q.doc_num,  
                        pq_type = q.doc_type, rev_num = q.doc_iss_num, status = q.status_code  
                        from unit u, mfg_inst m, qa_engr_rels q, tms_inst t  
                        where :h_empl_num = u.da_qa_engr and m.mfg_inst_num = t.mfg_inst_num  
                        and (u.unit_stem = m.unit_stem and u.unit_da_suf = m.unit_da_suf) and  
                        q.doc_type = t.doc_type and q.doc_num = t.doc_num and q.doc_iss_num  
                        = t.doc_iss_num;
```

```
    unloadtable assignment
```

```
    { h_nomen = '  
      h_desc = ';
```

```
    qaengr := select h_nomen = c.com_nom_type + c.com_nom_num  
                  from component c where c.unit_stem = :assignment.part_no  
                  and c.unit_da_suf = :assignment.sffx;
```

```
    inquire_ingres(rcount=rowcount);
```

```
    if rcount = 0 then
```

```
      qaengr := select h_nomen = w.wpn_pgm_code + w.wpn_test_cfg + '-' +  
                    w.wpn_mod_num from weapon w  
                    where w.unit_stem = :assignment.part_no and  
                    w.unit_da_suf = :assignment.sffx;
```

```
    endif;
```

```
    qaengr := select h_desc = u.unit_desc from unit_desc u  
                  where u.unit_stem = :assignment.part_no and  
                  u.unit_da_suf = :assignment.sffx;
```

```
    qaengr := select est_date = d.est_comp_dt from document d  
                  where :assignment.dcmt_num = d.doc_num  
                  and :assignment.pq_type = d.doc_type  
                  and :assignment.rev_num = d.doc_iss_num;
```

```
    assignment.est_comp_dt := est_date;
```

```
    assignment.cmpnt_nomen := h_nomen;
```

```
    assignment.description := h_desc;
```

```
  };
```

```
endif;
```

```
}
```

'CLEAR' = {

```
  clear field all;
```

```
  resume field engr_name;
```

```
}
```

```
'HELP', key frskey1 =  
{  
    help_forms (subject = 'Retrieve QAENGR',  
        file = 'db1:[dbaprdb.help]qaengr.help');  
}  
'XIT', key frskey3 =  
{  
    RETURN;  
}  
  
'STOP', key frskey2 =  
{  
    EXIT  
;  
}
```

```

/*
      Procedure for QAMAIN frame
      Written by Norman Stevens

*/
Initialize (answer = vchar(1)) =
{
}

'DATA' =
{
    callframe QAEMAIN;
}

'RETRIEVE' =
{
    callframe QARMAIN;
}

'GEN REPTS' =
{
    callframe QAREPORT;
}

'HELP', key frskey1 =
{
    help_forms (subject = 'QA MAIN MENU',
        file = 'db1:[dbaprdb.help]qamain.help');
}

'XIT', key frskey3 =
{
    RETURN;
}

'STOP', key frskey2 =
{
    EXIT;
}

```

/*

Procedure for QAREPORT frame
Written by Norman Stevens

QAREPORT

*/
Initialize (answer = vchar(1)) =
{
}

'NOMENCLATURE' =

{
 call system 'submit db1:[dbaprdb.source]nomen_rpt.com' ;

/*
 The above statement submits a com file which runs the report

*/
 answer := prompt 'NOMEN.RPT will be printed on MYPRINTER. Hit <CR> ' +
 'to continue.' ;
}

'ENGINEER' =

{
 call system 'submit db1:[dbaprdb.source]qa_orgn_rpt.com' ;

/*
 The above statement submits a com file which runs the report

*/
 answer := prompt 'QA_ORGN_.RPT will be printed on MYPRINTER. Hit <CR> ' +
 'to continue.' ;
}

'PART NUM' =

{
 call system 'submit db1:[dbaprdb.source]part_num_rpt.com' ;

/*
 The above statement submits a com file which runs the report

*/
 answer := prompt 'PART_NUM.RPT will be printed on MYPRINTER. Hit <CR> ' +
 'to continue.' ;
}

'WPN PGM' =

{
 call system 'submit db1:[dbaprdb.source]wpn_pgm_rpt.com' ;

/*
 The above statement submits a com file which runs the report

*/
 answer := prompt 'WPN_PGM.RPT will be printed on MYPRINTER. Hit <CR> ' +
 'to continue.' ;
}

'WPQA ORGN' =

{
 call system 'submit db1:[dbaprdb.source]wpn_pgm_qa_rpt.com' ;

/*
 The above statement submits a com file which runs the report

*/
 answer := prompt 'WPN_PGM_QA.RPT will be printed on MYPRINTER. Hit <CR> ' +
 'to continue.' ;
}

'AGENCY' =

```

{
    call system 'submit db1:[dbaprdb.source]pa_rpt.com';
/*
    The above statement submits a com file which runs the report */
    answer := prompt 'PA.RPT will be printed on MYPRINTER. Hit <CR>' +
        'to continue.';
}
'HELP', key frskey1 =
{
    help_forms (subject = 'QA Reports Main Menu',
        file = 'db1:[dbaprdb.help]qareport.help');
}
'XIT', key frskey3 =
{
    RETURN;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

/*

Procedure for QARMAIN frame
Written by Norman Stevens

QARMAIN

*/

'PQ TEAM' =

```
{  
    callframe PQTTEAM;  
}
```

'QA ENGR' =

```
{  
    callframe QAENGR;  
}
```

'DOC HIST' =

```
{  
    callframe DOCHIST;  
}
```

'COMPONENTS' =

```
{  
    callframe WPNSYS;  
}
```

'ENGR RELS' =

```
{  
    callframe ENGRELS;  
}
```

'LIST PARTS' =

```
{  
    callframe PARTS;  
}
```

'HELP', key frskey1 =

```
{  
    help_forms (subject = 'QA RETRIEVAL MENU',  
                file = 'db1:[dbaprdb.help]qarmain.help');  
}
```

'XIT', key frskey3 =

```
{  
    RETURN;  
}
```

'STOP', key frskey2 =

```
{  
    EXIT;  
}
```

```

/*
      Procedure for QUAPLAN frame
      Written by Norman Stevens
*/
Initialize (h_mfg_num = vchar(4), h_supplier = vchar(26), h_wpn_pgm = vchar(4),
            h_pgm_code = vchar(4), h_tcfg = vchar(7), h_modno = vchar(2),
            h_mod_num = vchar(2), h_doc_type = vchar(4), h_type = vchar(2),
            h_rqrd_dt = vchar(11), h_est_c_dt = vchar(11), rcount = integer1,
            h_iss_dt = vchar(11), h_iss_num = vchar(2), h_mi_num = vchar(10),
            h_mfg_code = vchar(3), h_doc_num = vchar(8), h_e_num = vchar(4),
            e_i_num = vchar(4), h_e_type = vchar(2), answer = vchar(1)) =
{
h_mfg_num := '';
h_supplier := ' ';
h_wpn_pgm := ' ';
h_pgm_code := ' ';
h_tcfg := '';
h_modno := ' ';
resume field engr_rels_no;
}

'RETRIEVE' =
{
/*
      DSL necessary to retrieve the issue date and revision number if
      available
*/
      if engr_rels_no = '' then
        answer := prompt 'You must enter an Engineering Release Number. Hit ' +
          '<CR> to continue';
        resume field engr_rels_no;
      endif;
      if rev_num = '' then
        quaplan := select rev_num = max(eer.doc_iss_num) from qa_engr_rels eer
          where :engr_rels_no = eer.doc_num;
      endif;
      quaplan := select issue_date = eer.issue_dt, h_mfg_num = eer.mfg_inst_num,
        h_doc_type = eer.doc_type from qa_engr_rels eer
        where eer.doc_num = :engr_rels_no and eer.doc_iss_num = :rev_num;
      if h_doc_type != 'EER' then
        answer := prompt 'This document is not an EER; Hit <CR> to continue';
      endif;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        answer := prompt 'No information is available on this order yet. Enter ' +
          'any available data. Hit <CR> to continue';
      else
        if h_mfg_num != '' then
/*
          DSL necessary to retrieve the estimated completion date if
          available
*/
          quaplan := select eer_est_c_dt = d.est_comp_dt from document d
            where d.doc_type = :h_doc_type and d.doc_num = :engr_rels_no and
              d.doc_iss_num = :rev_num;
/*
          DSL necessary to retrieve the supplier, the weapon program and the part
          number and suffix
*/
          quaplan := select h_wpn_pgm = m.wpn_pgm_code,

```



```

        h_mod_num = m.wpn_mod_num, part_num = m.unit_stem,
        suffix = m.unit_da_suf from mfg_inst m
        where :h_mfg_num = m.mfg_inst_num;
    quaplan := select h_supplier = c.company_name, h_mfg_code = m.mfg_code
        from company c, mfg_inst m where :h_mfg_num = m.mfg_inst_num and
        m.mfg_code = c.mfg_code;
    supplier = :h_supplier;
    wpn_pgm = :h_wpn_pgm;
    mod_num = :h_mod_num;
    if part_num != '' then
/*
DSL necessary to retrieve production agency and the nomenclature
*/
        quaplan = select prod_agency = c.company_name from company c,
            unit u where u.prp_mfg_code = c.mfg_code
            and (u.unit_stem = :part_num and u.unit_da_suf = :suffix);
        quaplan = select nomenclature = cmpnt.com_nom_type +
            cmpnt.com_nom_num from component cmpnt where
            cmpnt.unit_stem = :part_num and cmpnt.unit_da_suf = :suffix;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            quaplan = select h_pgm_code = w.wpn_pgm_code,
                h_tcfg = w.wpn_test_cfg, h_modno = w.wpn_mod_num
                from weapon w where w.unit_stem = :part_num
                and w.unit_da_suf = :suffix;
            if h_tcfg = 'T' then
                nomenclature = h_pgm_code + '-' + h_modno;
            else
                nomenclature = h_pgm_code + '/' + h_tcfg + '-' + h_modno;
            endif;
        endif;
    endif;
endif;
/*
DSL necessary to retrieve the description of the part
*/
    quaplan := select description = u.unit_desc from unit_desc u
        where u.unit_stem = :part_num and u.unit_da_suf = :suffix;
/*
DSL necessary to retrieve equipment data
*/
    equipment := select equip_num = mei.equip_type + mei.equip_num,
        serial_num = mei.serial_num, doc_type = ei.doc_type,
        er_num = ei.doc_num, iss_num = ei.doc_iss_num
        from mfg_eqp_inst mei, eq_inst ei where mei.mfg_inst_num = :h_mfg_num
        and mei.eqp_inst_num = ei.eqp_inst_num;
    unloadtable equipment
    begin
        quaplan := select h_type = left(:equipment.equip_num, 2) from unit;
        if h_type = 'GA' then
            equipment.eqp_desc = 'GAGE';
        elseif h_type = 'PT' then
            equipment.eqp_desc = 'TESTER';
        endif;
        quaplan := select h_rqrd_dt = d.required_dt,
            h_est_c_dt = d.est_comp_dt from document d
            where d.doc_num = :equipment.er_num
            and d.doc_type = :equipment.doc_type
            and d.doc_iss_num = :equipment.iss_num;
        inquire_ingres(rcount=rowcount);

```

```

if rcount != 0 then
    equipment.qer_rqrd_dt := h_rqrd_dt;
    equipment.est_comp_dt := h_est_c_dt;
endif;
quaplan := select h_iss_dt = eer.issue_dt from qa_engr_rels eer
    where eer.doc_num = :equipment.er_num
        and eer.doc_type = :equipment.doc_type
        and eer.doc_iss_num = :equipment.iss_num;
inquire_ingres(rcount=rowcount);
if rcount != 0 then
    equipment.iss_date := h_iss_dt;
endif;
end;
endif;
}

'ADD/REVISE' = {
if engr_rels_no = '' then
    answer := prompt 'You must enter a document number. Hit <CR> to continue';
    resume field engr_rels_no;
endif;
if :issue_date != '' then
    quaplan := select h_iss_num = er.doc_iss_num from qa_engr_rels er
        where er.doc_num = :engr_rels_no and er.doc_iss_num = :rev_num
        and er.doc_type = 'EER';
    inquire_ingres(rcount=rowcount);
    if rcount = 0 then
        if rev_num = '0' then
            quaplan := select h_iss_num = er.doc_iss_num from qa_engr_rels er
                where er.doc_num = :engr_rels_no and er.doc_iss_num = ''
                and er.doc_type = 'EER';
            inquire_ingres(rcount=rowcount);
            if rcount = 0 then
                insert into qa_engr_rels (doc_type, doc_num, doc_iss_num,
                    issue_dt) values ('EER', :engr_rels_no, :rev_num,
                    :issue_date);
                answer := prompt 'DOCUMENT ADDED to PREMIS. Hit <CR> to ' +
                    'continue';
            else
                update qa_engr_rels er set issue_dt = :issue_date,
                    doc_iss_num = :rev_num where er.doc_type = 'EER'
                    and er.doc_num = :engr_rels_no and er.doc_iss_num = '';
                answer := prompt 'Document revised in PREMIS. Hit <CR> to ' +
                    'continue';
            endif;
        else
            insert into qa_engr_rels (doc_type, doc_num, doc_iss_num, issue_dt)
                values ('EER', :engr_rels_no, :rev_num, :issue_date);
            answer := prompt 'DOCUMENT ADDED to PREMIS. Hit <CR> to continue';
        endif;
    else
        update qa_engr_rels er set issue_dt = :issue_date
            where er.doc_type = 'EER' and er.doc_num = :engr_rels_no
            and er.doc_iss_num = :rev_num;
        answer := prompt 'Document revised in PREMIS. Hit <CR> to continue';
    endif;
endif;
if :eer_est_c_dt != '' then
    quaplan := select h_iss_num = d.doc_iss_num from document d
        where d.doc_num = :engr_rels_no and d.doc_iss_num = :rev_num

```

```

        and d.doc_type = 'EER';
    inquire_ingres(rcount=rowcount);
    if rcount = 0 then
        if rev_num = '0' then
            quaplan := select h_iss_num = d.doc_iss_num from document d
                where d.doc_num = :engr_rels_no and d.doc_iss_num = ''
                    and d.doc_type = 'EER';
            inquire_ingres(rcount=rowcount);
            if rcount = 0 then
                insert into document (doc_type, doc_num, doc_iss_num,
                    est_comp_dt) values ('EER', :engr_rels_no, :rev_num,
                        :eer_est_c_dt);
            else
                update document d set est_comp_dt = :eer_est_c_dt,
                    doc_iss_num = :rev_num where d.doc_type = 'EER'
                        and d.doc_num = :engr_rels_no and d.doc_iss_num = '';
            endif;
        else
            insert into document (doc_type, doc_num, doc_iss_num, est_comp_dt)
                values ('EER', :engr_rels_no, :rev_num, :eer_est_c_dt);
            endif;
        else
            update document d set est_comp_dt = :eer_est_c_dt
                where d.doc_type = 'EER' and d.doc_num = :engr_rels_no
                    and d.doc_iss_num = :rev_num;
            endif;
        inquire_ingres(rcount = rowcount);
        if rcount != 0 then
            h_doc_type = 'EER';
        endif;
    endif;
    if :part_num = '' then
        if nomenclature != '' then
            quaplan := select part_num = c.unit_stem, suffix = c.unit_da_suf
                from component c
                where :nomenclature = c.com_nom_type + c.com_nom_num;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                answer := prompt 'No component with that nomenclature exists in ' +
                    'PREMIS. Hit <CR> to continue';
            endif;
        endif;
    else
        quaplan := select nomenclature = c.com_nom_type + c.com_nom_num
            from component c
            where c.unit_stem = :part_num and c.unit_da_suf = :suffix;
        inquire_ingres(rcount=rowcount);
        if rcount = 0 then
            quaplan := select h_pgm_code = w.wpn_pgm_code,
                h_tcfg = w.wpn_test_cfg, h_modno = w.wpn_mod_num from weapon w
                where w.unit_stem = :part_num and w.unit_da_suf = :suffix;
            inquire_ingres(rcount=rowcount);
            if rcount = 0 then
                part_num = '';
                answer := prompt 'The part number does not exist in PREMIS. Hit ' +
                    '<CR> to continue';
            else
                if h_tcfg = '' then
                    nomenclature = h_pgm_code + '-' + h_modno;
                else

```

```

        nomenclature = h_pgm_code + '/' + h_tcfg + '-' + h_modno;
    endif;
endif;
endif;
endif;
if :part_num != '' then
    quaplan := select description = u_d.unit_desc from unit_desc u_d
        where u_d.unit_stem = :part_num and u_d.unit_da_suf = :suffix;
endif;
if :wpn_pgm != '' then
    if mod_num = '' then
        quaplan := select h_mi_num = w_p.wpn_mi_num from weapon_pgm w_p
            where :wpn_pgm = w_p.wpn_pgm_code;
        inquire_ingres(rcount=rowcount);
        if rcount = 0 then
            answer := prompt wpn_pgm + ' is not a valid weapon ' +
                'program. Hit <CR> to continue';
            wpn_pgm = '';
        endif;
    else
        quaplan := select h_mi_num = w_p.wpn_mi_num from weapon_pgm w_p
            where :wpn_pgm = w_p.wpn_pgm_code and :mod_num = w_p.wpn_mod_num;
        inquire_ingres(rcount=rowcount);
        if rcount = 0 then
            answer := prompt wpn_pgm + '-' + mod_num + ' is not a valid weapon ' +
                'program. Hit <CR> to continue';
            wpn_pgm = '';
            mod_num = '';
        endif;
    endif;
endif;
endif;
if :part_num != '' and :wpn_pgm != '' then
    if supplier = '' then
        h_mfg_code = '';
    elseif supplier = h_supplier then
        h_mfg_code = h_mfg_code;
    else
        quaplan := select supplier = c.company_name,
            h_mfg_code = c.mfg_code from company c
            where :supplier = c.mfg_code or :supplier = c.company_name;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            quaplan := select supplier = c.company_name,
                h_mfg_code = c.mfg_code from company c, company_syn c_s
                where :supplier = c_s.company_name and
                    c_s.mfg_code = c.mfg_code;
            inquire_ingres(rcount = rowcount);
            if rcount = 0 then
                answer := prompt 'The supplier is not found in PREMIS. Hit ' +
                    '<CR> to continue';
                h_mfg_code = '';
            else
                h_supplier := supplier;
            endif;
        else
            h_supplier := supplier;
        endif;
    endif;
endif;
if :rev_num = '' then
    quaplan := select h_mfg_num = er.mfg_inst_num,

```

```

        rev_num = er.doc_iss_num from qa_engr_rels er
        where :engr_rels_no = er.doc_num and er.doc_type = 'EER';
    else
        quaplan := select h_mfg_num = er.mfg_inst_num from qa_engr_rels er
        where :engr_rels_no = er.doc_num and er.doc_type = 'EER'
        and :rev_num = er.doc_iss_num;
    endif;
    inquire_ingres(rcount=rowcount);
    if rcount = 0 then
        quaplan := select h_mfg_num = mi.mfg_inst_num from mfg_inst mi
        where mi.unit_stem = :part_num and mi.unit_da_suf = :suffix and
        mi.wpn_pgm_code = :wpn_pgm and mi.wpn_mod_num = :mod_num and
        mi.mfg_code = :h_mfg_code;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            quaplan := select h_mfg_num = vchar(max(int4(mi.mfg_inst_num)) + 1)
            from mfg_inst mi;
            insert into mfg_inst (mfg_inst_num, unit_stem, unit_da_suf,
            wpn_pgm_code, wpn_mod_num, mfg_code) values(:h_mfg_num,
            :part_num, :suffix, :wpn_pgm, :mod_num, :h_mfg_code);
        else
            update mfg_inst mi set mfg_code = :h_mfg_code where
            mi.mfg_inst_num = :h_mfg_num;
        endif;
        insert into qa_engr_rels (doc_type, doc_num, doc_iss_num,
        mfg_inst_num) values ('EER', :engr_rels_no, :rev_num, :h_mfg_num);
    elseif h_mfg_num = '' then
        quaplan := select h_mfg_num = mi.mfg_inst_num from mfg_inst mi
        where mi.unit_stem = :part_num and mi.unit_da_suf = :suffix and
        mi.wpn_pgm_code = :wpn_pgm and mi.wpn_mod_num = :mod_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            quaplan := select h_mfg_num = vchar(max(int4(mi.mfg_inst_num)) + 1)
            from mfg_inst mi;
            insert into mfg_inst (mfg_inst_num, unit_stem, unit_da_suf,
            wpn_pgm_code, wpn_mod_num) values(:h_mfg_num, :part_num,
            :suffix, :wpn_pgm, :mod_num);
        else
            update mfg_inst mi set mfg_code = :h_mfg_code
            where mi.mfg_inst_num = :h_mfg_num;
        endif;
        update qa_engr_rels er set mfg_inst_num = :h_mfg_num where
        er.doc_type = 'EER' and er.doc_num = :engr_rels_no and
        er.doc_iss_num = :rev_num;
    else
        update mfg_inst mi set mfg_code = :h_mfg_code,
        wpn_pgm_code = :wpn_pgm, wpn_mod_num = :mod_num,
        unit_stem = :part_num, unit_da_suf = :suffix
        where mi.mfg_inst_num = :h_mfg_num;
        update qa_engr_rels er set mfg_inst_num = :h_mfg_num where
        er.doc_type = 'EER' and er.doc_num = :engr_rels_no and
        er.doc_iss_num = :rev_num;
    endif;
endif;
unloadtable equipment
begin
/*
    USL necessary to update document dates
*/
    if :equipment.er_num != '' then

```

```

if equipment.qer_rqrd_dt != '' or equipment.est_comp_dt != '' then
  quaplan := select h_doc_num = d.doc_num from document d
    where :equipment.doc_type = d.doc_type
      and :equipment.er_num = d.doc_num
      and :equipment.iss_num = d.doc_iss_num;
  inquire_ingres (rcount = rowcount);
  if rcount = 0 then
    if :equipment.iss_num = '0' then
      quaplan := select h_doc_num = d.doc_num from document d
        where d.doc_type = :equipment.doc_type
          and d.doc_num = :equipment.er_num
          and d.doc_iss_num = '';
      inquire_ingres (rcount = rowcount);
      if rcount = 0 then
        insert into document(doc_type, doc_num, doc_iss_num,
          est_comp_dt, required_dt) values (:equipment.doc_type,
            :equipment.er_num, :equipment.iss_num,
            :equipment.est_comp_dt, :equipment.qer_rqrd_dt);
      else
        update document d set required_dt = :equipment.qer_rqrd_dt,
          est_comp_dt = :equipment.est_comp_dt,
          doc_iss_num = :equipment.iss_num
          where :equipment.doc_type = d.doc_type
            and :equipment.er_num = d.doc_num
            and d.doc_iss_num = '';
      endif;
    else
      insert into document(doc_type, doc_num, doc_iss_num,
        est_comp_dt, required_dt) values (:equipment.doc_type,
          :equipment.er_num, :equipment.iss_num,
          :equipment.est_comp_dt, :equipment.qer_rqrd_dt);
    endif;
  else
    update document d set required_dt = :equipment.qer_rqrd_dt,
      est_comp_dt = :equipment.est_comp_dt
    where :equipment.doc_type = d.doc_type
      and :equipment.er_num = d.doc_num
      and :equipment.iss_num = d.doc_iss_num;
  endif;
endif;

if equipment.iss_date != '' then
  quaplan := select h_doc_num = qer.doc_num from qa_engr_rels qer
    where :equipment.doc_type = qer.doc_type
      and :equipment.er_num = qer.doc_num
      and :equipment.iss_num = qer.doc_iss_num;
  inquire_ingres (rcount = rowcount);
  if rcount = 0 then
    if equipment.iss_num = '0' then
      quaplan := select h_doc_num = qer.doc_num from qa_engr_rels qer
        where :equipment.doc_type = qer.doc_type
          and :equipment.er_num = qer.doc_num
          and :equipment.iss_num = '';
      inquire_ingres (rcount = rowcount);
      if rcount = 0 then
        insert into qa_engr_rels(doc_type, doc_num, doc_iss_num,
          issue_dt, mfg_inst_num) values (:equipment.doc_type,
            :equipment.er_num, :equipment.iss_num,
            :equipment.iss_date, :h_mfg_num);
      else
        update qa_engr_rels qer set issue_dt = :equipment.iss_date,

```

```

        doc_iss_num = :equipment.iss_num,
        mfg_inst_num = :h_mfg_num
        where :equipment.doc_type = qer.doc_type
              and :equipment.er_num = qer.doc_num
              and qer.doc_iss_num = '';
    endif;
else
    insert into qa_engr_rels(doc_type, doc_num, doc_iss_num,
        issue_dt, mfg_inst_num) values (:equipment.doc_type,
        :equipment.er_num, :equipment.iss_num, :equipment.iss_date,
        :h_mfg_num);
endif;
else
    update qa_engr_rels qer set issue_dt = :equipment.iss_date,
        mfg_inst_num = :h_mfg_num
        where :equipment.doc_type = qer.doc_type
              and :equipment.er_num = qer.doc_num
              and :equipment.iss_num = qer.doc_iss_num;
endif;
endif;
endif;
/*
DSL necessary to update equipment data
*/
if equipment.equip_num != '' then
    quaplan := select h_type = left(:equipment.equip_num, 2),
        h_e_num = shift(:equipment.equip_num, -2) from equipment e;
    quaplan := select h_e_type = e.equip_type from equipment e
        where e.equip_type = :h_type and e.equip_num = :h_e_num
              and e.serial_num = :equipment.serial_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        insert into equipment (equip_num, equip_type, serial_num)
            values (:h_e_num, :h_type, :equipment.serial_num);
    endif;
    e_i_num = '';
    quaplan := select e_i_num = mei.eqp_inst_num from mfg_eqp_inst mei
        where :equipment.equip_num = mei.equip_type + mei.equip_num
              and :equipment.serial_num = mei.serial_num
              and :h_mfg_num = mei.mfg_inst_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
        quaplan := select e_i_num = vchar(max(int4(mei.eqp_inst_num)) + 1)
            from mfg_eqp_inst mei;
        insert into mfg_eqp_inst(equip_num, equip_type, serial_num,
            eqp_inst_num, mfg_inst_num) values (:h_e_num, :h_type,
            :equipment.serial_num, :e_i_num, :h_mfg_num);
    endif;
    if e_i_num != '' then
        quaplan := select h_doc_num = ei.doc_num from eq_inst ei
            where :equipment.er_num = ei.doc_num and
                  :equipment.doc_type = ei.doc_type and
                  :equipment.iss_num = ei.doc_iss_num and
                  :e_i_num = ei.eqp_inst_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            if :equipment.iss_num = '0' then
                quaplan := select h_doc_num = ei.doc_num from eq_inst ei
                    where ei.doc_num = :equipment.er_num
                          and ei.doc_type = :equipment.doc_type

```

```

        and ei.doc_iss_num = ''
        and ei.eqp_inst_num = :e_i_num;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        insert into eq_inst (doc_type, doc_num, doc_iss_num,
            eqp_inst_num, hrd_sft_flag) values (:equipment.doc_type,
            :equipment.er_num, :equipment.iss_num, :e_i_num, 'H');
    else
        update eq_inst ei set doc_iss_num = :equipment.iss_num
            where ei.doc_type = :equipment.doc_type
            and ei.doc_num = :equipment.er_num
            and ei.doc_iss_num = ''
            and ei.eqp_inst_num = :e_i_num;
    endif;
else
    insert into eq_inst (doc_type, doc_num, doc_iss_num,
        eqp_inst_num, hrd_sft_flag) values (:equipment.doc_type,
        :equipment.er_num, :equipment.iss_num, :e_i_num, 'H');
endif;
endif;
endif;
end;
}

'CLEAR' =
{
    clear field all;
    resume field engr_rels_no;
}

'XIT' =
{
    return;
}

'STOP' =
{
    exit;
}

```


/*

Procedure for REPORT frame

written by John Orman

*/

```
'NEW' =  
{  
    CALLFRAME ALLID(ALLID.REP_TYPE := 'NEW PARTS W/O') ;  
}  
'ALL_ID' =  
{  
    CALLFRAME ALLID(ALLID.REP_TYPE := 'ALL PARTS AND') ;  
}  
'WEAPON' =  
{  
    CALLFRAME WEAPON(WEAPON.REP_TYPE := 'ALL PARTS AND') ;  
}  
'PROD' =  
{  
    CALLFRAME PROD(PROD.REP_TYPE := 'ALL') ;  
}  
'INCOMPLETE' =  
{  
    CALLFRAME NOTCOMPDC ;  
}  
'COMPLETE' =  
{  
    CALLFRAME COMPDCC ;  
}  
'HELP', key frskey1 =  
{  
    help_forms (subject = 'DATA ENGR REPORTS MENU',  
        file = 'db1:[dbaprdb.help]report.help');  
}  
'XIT', key frskey3 =  
{  
    RETURN ;  
}  
'STOP', key frskey2 =  
{  
    EXIT ;  
}
```

```

/*
                                Procedure for RPTCHOICE frame
                                Written by Norman Stevens
*/
Initialize (answer = vchar(1)) =
{
}

'DISPLAY' =
{
    call report (report = 'parts_rpt', mode = 'wrap');
}
/*
The above statement runs a report named "parts_rpt" which was created
using RBF and sends the report to the screen.
*/
}

'FILE' =
{
    call report (report = 'parts_rpt', mode = 'column', pagelength = '65',
        file = 'part_num.rpt');
}
/*
The above statement runs a report named "parts_rpt" which was created
using RBF and stores the report in a file named "part_num.rpt".
*/
answer := prompt 'The requested report has been stored under the name ' +
    'PART_NUM.RPT. Hit <CR> to continue.';
}

'PRINT' =
{
    call report (report = 'parts_rpt', mode = 'column', pagelength = '65',
        file = 'part_num.rpt');
}
/*
The above statement runs a report named "parts_rpt" which was created
using RBF and stores the report in a file named "part_num.rpt".
*/
call system '0db1:[dbaprd.b.source]print_dispose part_num.rpt' ;
/*
The above statement causes part_num.rpt to be printed on the printer
which has been assigned to MYPRINTER.
*/
answer := prompt 'PART_NUM.RPT is being printed on MYPRINTER. Hit <CR> ' +
    'to continue.';
}

'XIT' =
{
    return;
}

'STOP' =
{
    exit;
}

```

Procedure for TMSCNTR frame
Written by Norman Stevens

TMSCNTR

```

/*
/
initialize (rcount = integer1, h_stat = vchar(4), h_wpn_code = vchar(4),
h_modno = vchar(3), h_tcfg = vchar(7), mfr_code = vchar(3),
err_count = integer1, com_ctr_enum = vchar(6), h_cont = vchar(25),
h_enum = vchar(6), h_dwg_no = vchar(8), qa_name = vchar(25),
qa_enum = vchar(6), cmpnt_nom = vchar(12), u_desc = vchar(32),
answer = vchar(1), nomenclature = vchar(12), prtno = vchar(7),
h_etype = vchar(2), h_eqnum = vchar(4), h_eqp_num = vchar(4),
subject.mfg_num = vchar(4), subject.ei_num = vchar(4),
subject.cod = vchar(3), h_mfg_num = vchar(4), sfx = vchar(3),
rele_docs.part_no = vchar(7), rele_docs.sffx = vchar(3)) =
{
resume field engr_rels_no;
}

'RETRIEVE' =
{
/*
OSL necessary to retrieve TMS CENTER data from PREMIS
*/
if engr_rels_no = '' then
answer := prompt 'You must enter an engineering release number.' +
'Hit <CR> to continue';
resume field engr_rels_no;
endif;
if rev_num = '' then
tmscntr := select rev_num = max(eer.doc_iss_num) from qa_engr_rels eer
where :engr_rels_no = eer.doc_num;
endif;
message 'Retrieving Information ...';
tmscntr := select doc_type = eer.doc_type, trans_date = eer.trans_com_dt,
message_num = eer.message_num, twx_rcd = eer.twx_receipt,
m11_dir_name = eer.m11_dir_name, file_cab_loc = eer.file_cab_loc,
h_stat = eer.status_code from qa_engr_rels eer
where :engr_rels_no = eer.doc_num and :rev_num = eer.doc_iss_num;
inquire_ingres(rcount=rowcount);
if rcount = 0 then
answer := prompt 'THERE IS NO DOCUMENT with that number in PREMIS; You' +
'may add a new one; Hit <CR> to continue.';
else
/*
OSL necessary to retrieve the part number, supplier, equipment number,
and weapon program from PREMIS
*/
rcount := 0;
tmscntr := select rcount = count (*) from eq_inst ei
where :engr_rels_no = ei.doc_num and :rev_num = ei.doc_iss_num
and :doc_type = ei.doc_type;
if rcount = 0 then
subject := select part_num = mi.unit_stem, suffix = mi.unit_da_suf,
wpn_pgm = mi.wpn_pgm_code, mod_num = mi.wpn_mod_num,
cod = mi.mfg_code, supplier = c.company_name,
mfg_num = ti.mfg_inst_num from mfg_inst mi, tms_inst ti, company c
where (:engr_rels_no = ti.doc_num and :rev_num = ti.doc_iss_num
and :doc_type = ti.doc_type) and ti.mfg_inst_num = mi.mfg_inst_num
and mi.mfg_code = c.mfg_code;
else

```

```

subject := select part_num = mi.unit_stem, suffix = mi.unit_da_suf,
wpn_pgm = mi.wpn_pgm_code, mod_num = mi.wpn_mod_num,
cod = mi.mfg_code, supplier = c.company_name,
eqp_num = mei.equip_type + mei.equip_num,
mfg_num = mei.mfg_inst_num, ei_num = mei.eqp_inst_num
from eq_inst ei, mfg_eqp_inst mei, mfg_inst mi, company c
where (:engr_rels_no = ei.doc_num and :rev_num = ei.doc_iss_num
and :doc_type = ei.doc_type) and ei.eqp_inst_num=mei.eqp_inst_num
and mei.mfg_inst_num = mi.mfg_inst_num
and mi.mfg_code = c.mfg_code;

endif;

/*
DSL necessary to retrieve the nomenclature from PREMIS
*/
unloadtable subject
begin
if subject.part_num != '' then
tmscntr := select nomenclature = c.com_nom_type + c.com_nom_num from
component c where :subject.part_num = c.unit_stem
and :subject.suffix = c.unit_da_suf;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
tmscntr := select h_wpn_code = w.wpn_pgm_code,
h_modno = w.wpn_mod_num, h_tcfg = w.wpn_test_cfg from weapon w
where :subject.part_num = w.unit_stem
and :subject.suffix = w.unit_da_suf;
if h_tcfg = '' then
nomenclature = h_wpn_code + '-' + h_modno;
else
nomenclature = h_wpn_code + '/' + h_tcfg + '-' + h_modno;
endif;
endif;
subject.nomen := :nomenclature;
nomenclature := '';
endif;
end;

/*
DSL necessary to retrieve the qa engineer information from PREMIS
*/
if subject[1].part_num != '' then
tmscntr := select qa_engr_name = p.first_name + ' ' + p.middl_init +
' ' + p.last_name, qa_orgn = p.asnd_orgn, qa_phone = '(' +
left(p.phone_num, 3) + ')' + left(shift(p.phone_num, -3), 3) + '-'
+ shift(p.phone_num, -6) from person p, unit u
where :subject[1].part_num = u.unit_stem
and :subject[1].suffix = u.unit_da_suf
and u.da_qa_engr = p.id_num;
if qa_phone = '()-' then
qa_phone := '';
endif;
else
qa_engr_name := '';
qa_orgn := '';
qa_phone := '';
endif;

/*
DSL necessary to retrieve the results report flag from PREMIS
*/
tmscntr := select rslt_rpt_fl = er.rslt_rpt_fl from eng_eval_rel er
where er.doc_type = :doc_type and er.doc_num = :engr_rels_no and

```

```

:rev_num = er.doc_iss_num;

/*
DSL necessary to retrieve the communication center contact data from PREMIS
*/
tmscntr := select h_cont = p.first_name + ' ' + p.middl_init + ' '
+ p.last_name, com_ctr_orgn = p.asnd_orgn, h_enum = p.id_num,
com_ctr_phon = '(' + left(p.phone_num, 3) + ') ' +
left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
from person p, qa_engr_rels eer where eer.com_ctr_cont = p.id_num and
(eer.doc_type = :doc_type and eer.doc_num = :engr_rels_no and
:rev_num = eer.doc_iss_num);
com_ctr_cont := h_cont;
if com_ctr_phon = '()-' then
com_ctr_phon := '';
endif;

/*
DSL necessary to convert the status from PREMIS
*/
if h_stat = 'ACA' then
status = 'ACCEPTABLE WITH CORR. ACTION';
elseif h_stat = 'ACC' then
status = 'ACCEPTABLE';
elseif h_stat = 'AP' then
status = 'ACTION PENDING';
elseif h_stat = 'CA' then
status = 'CONDITIONALLY ACCEPTABLE';
elseif h_stat = 'FC' then
status = 'FULL CERTIFICATION';
elseif h_stat = 'INST' then
status = 'INSTRUCTIONS';
elseif h_stat = 'NA' then
status = 'NOT ACCEPTABLE';
elseif h_stat = 'NR' then
status = 'NOT REQUIRED';
elseif h_stat = 'P' then
status = 'PLAN';
elseif h_stat = 'PC' then
status = 'PARTIAL CERTIFICATION';
elseif h_stat = 'PEQ' then
status = 'PLANS EQ';
elseif h_stat = 'PQS' then
status = 'QS PLAN';
elseif h_stat = 'PTMS' then
status = 'PLANS TMS';
elseif h_stat = 'R' then
status = 'REQUIRED';
elseif h_stat = 'REQ' then
status = 'RESULTS REPORT EQ';
elseif h_stat = 'REQP' then
status = 'RE-EQ PLAN';
elseif h_stat = 'REQR' then
status = 'RE-EQ RESULTS REPORT';
elseif h_stat = 'RETP' then
status = 'RE-TMS PLAN';
elseif h_stat = 'RETR' then
status = 'RE-TMS RESULTS REPORT';
elseif h_stat = 'REV' then
status = 'REVOKED';
elseif h_stat = 'RQS' then
status = 'RESULTS REPORT QS';

```

```

elseif h_stat = 'RQSP' then
    status = 'RE-QS PLAN';
elseif h_stat = 'RQSR' then
    status = 'RE-QS RESULTS REPORT';
elseif h_stat = 'RTMS' then
    status = 'RESULTS REPORT TMS';
else
    status = h_stat;
endif;
endif;
}

'PART/LIST' =
{
    if subject[1].part_num = '' then
        if subject.nomen = '' then
            answer := prompt 'To list by part, you must enter the nomenclature or'
                + ' the part number. Hit <CR> to continue';
            resume field subject.nomen;
        else
            nomenclature := subject[1].nomen;
            message 'Retrieving Information ... ';
            subject := select part_num = c.unit_stem, suffix = c.unit_da_suf,
                nomen = c.com_nom_type + c.com_nom_num from component c
                where :subject[1].nomen = c.com_nom_type + c.com_nom_num;
            inquire_ingres (rcount = rowcount);
            if rcount = 0 then
                subject := select part_num = c.unit_stem, suffix = c.unit_da_suf,
                    nomen = c.com_nom_type + c.com_nom_num from component c
                    where :subject[1].nomen = c.com_nom_type + ' ' + c.com_nom_num;
                inquire_ingres (rcount = rowcount);
                if rcount = 0 then
                    nomenclature := '';
                    answer := prompt 'No component with this nomenclature exists in'
                        + ' PREMIS. Hit <CR> to continue';
                    resume field subject.nomen;
                endif;
            endif;
        endif;
    else
        message 'Retrieving Information ... ';
        subject := select nomen = c.com_nom_type + c.com_nom_num
            from component c where :subject[1].part_num = c.unit_stem
            and :subject[1].suffix = c.unit_da_suf;
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
            tmscntr := select h_wpn_code = w.wpn_pgm_code, h_tcfg = w.wpn_test_cfg,
                h_modno = w.wpn_mod_num from weapon w
                where :subject[1].part_num = w.unit_stem
                and :subject[1].suffix = w.unit_da_suf;
            inquire_ingres (rcount = rowcount);
            if rcount = 0 then
                answer := prompt 'No part with this nomenclature exists in ' +
                    'PREMIS. Hit <CR> to continue';
                subject[1].part_num := '';
                resume field subject.nomen;
            else
                if h_tcfg = '' then
                    subject[1].nomen := h_wpn_code + '-' + h_modno;
                    nomenclature := h_wpn_code + '-' + h_modno;
                else

```

```

        else
            subject[1].nomen := h_wpn_code + '/' + h_tcfg + '-' + h_modno;
            nomenclature := h_wpn_code + '/' + h_tcfg + '-' + h_modno;
        endif;
    endif;
else
    nomenclature := subject[1].nomen;
endif;
endif;
if :subject[1].part_num != '' then
    tmscntr := select qa_name = p.first_name + ' ' + p.middl_init + ' ' +
        p.last_name from person p, unit u where p.id_num = u.da_qa_engr and
        (:subject[1].part_num = u.unit_stem
        and :subject[1].suffix = u.unit_da_suf);
    tmscntr := select u_desc = u_d.unit_desc from unit_desc u_d
        where :subject[1].part_num = u_d.unit_stem
        and :subject[1].suffix = u_d.unit_da_suf;
    relt_docs := select part_no = mi.unit_stem from mfg_inst mi, tms_inst ti
        where (:subject[1].part_num = mi.unit_stem
        and :subject[1].suffix = mi.unit_da_suf)
        and mi.mfg_inst_num = ti.mfg_inst_num;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
        relt_docs := select part_no = mi.unit_stem from mfg_inst mi,
            eq_inst ei, mfg_eqp_inst mei
            where (:subject[1].part_num = mi.unit_stem
            and :subject[1].suffix = mi.unit_da_suf)
            and mi.mfg_inst_num = mei.mfg_inst_num
            and mei.eqp_inst_num = ei.eqp_inst_num;
        inquire_ingres (rcount = rowcount);
        if rcount = 0 then
            answer := prompt 'There are no documents pertaining to that ' +
                'component in PREMIS';
            resume field subject.part_num;
        else
            relt_docs := select weapon_pgm = mi.wpn_pgm_code + '-' +
                mi.wpn_mod_num, part_no = mi.unit_stem, sffx = mi.unit_da_suf,
                type = ei.doc_type, er_num = ei.doc_num,
                iss_num = ei.doc_iss_num from eq_inst ei, mfg_eqp_inst mei,
                mfg_inst mi where (:subject[1].part_num = mi.unit_stem
                and :subject[1].suffix = mi.unit_da_suf)
                and mi.mfg_inst_num = mei.mfg_inst_num
                and mei.eqp_inst_num = ei.eqp_inst_num;
        endif;
    else
        relt_docs := select weapon_pgm = mi.wpn_pgm_code + '-' +
            mi.wpn_mod_num, part_no = mi.unit_stem, sffx = mi.unit_da_suf,
            type = ti.doc_type, er_num = ti.doc_num, iss_num = ti.doc_iss_num
            from tms_inst ti, mfg_inst mi
            where (:subject[1].part_num = mi.unit_stem
            and :subject[1].suffix = mi.unit_da_suf)
            and mi.mfg_inst_num = ti.mfg_inst_num;
    endif;
    unloadtable relt_docs
begin
    relt_docs.subj          := :nomenclature;
    relt_docs.qa_engr       := :qa_name;
    relt_docs.description := :u_desc;
end;
endif;

```

```

resume field relt_docs.weapon_pgm;
}

'ENGR/LIST' =
{
  if qa_engr_name = '' or qa_orgn = '' then
    answer := prompt 'You must enter the engineer name and organization to ' +
      ' list by engineer. Hit <CR> to continue';
    resume field qa_engr_name;
  endif;
  message 'Retrieving Information ...';
  tmscntr := select qa_enum = p.id_num, qa_name = p.first_name + ' ' +
    p.middl_init + ' ' + p.last_name, qa_phone = '(' + left(p.phone_num, 3)
    + ')' + left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
    from person p where :qa_engr_name = p.last_name and
      :qa_orgn = p.asnd_orgn;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    answer = prompt :qa_engr_name + ' is not found in PREMIS. Hit <CR> to ' +
      'continue';
    resume field qa_engr_name;
  else
    qa_engr_name := qa_name;
    if qa_phone = '()-' then
      qa_phone := '';
    endif;
    relt_docs := select part_no = u.unit_stem from unit u, mfg_inst mi,
      tms_inst ti where :qa_enum = u.da_qa_engr
      and (u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf)
      and mi.mfg_inst_num = ti.mfg_inst_num;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      relt_docs := select part_no = u.unit_stem from unit u, eq_inst ei,
        mfg_inst mi, mfg_eqp_inst mei where :qa_enum = u.da_qa_engr
        and (u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf)
        and mi.mfg_inst_num = mei.mfg_inst_num
        and mei.eqp_inst_num = ei.eqp_inst_num;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        answer := prompt 'There are no documents assigned to that ' +
          'engineer in PREMIS';
        resume field qa_engr_name;
      else
        relt_docs := select weapon_pgm = mi.wpn_pgm_code + '-'
          + mi.wpn_mod_num, part_no = mi.unit_stem, sffx = mi.unit_da_suf,
          er_num = ei.doc_num, type = ei.doc_type, iss_num = ei.doc_iss_num
          from unit u, mfg_inst mi, mfg_eqp_inst mei, eq_inst ei
          where :qa_enum = u.da_qa_engr
          and (u.unit_stem=mi.unit_stem and u.unit_da_suf = mi.unit_da_suf)
          and mi.mfg_inst_num = mei.mfg_inst_num
          and mei.eqp_inst_num = ei.eqp_inst_num;
        endif;
      else
        relt_docs := select weapon_pgm = mi.wpn_pgm_code + '-'
          + mi.wpn_mod_num, part_no = mi.unit_stem, sffx = mi.unit_da_suf,
          er_num = ti.doc_num, type = ti.doc_type, iss_num = ti.doc_iss_num
          from unit u, mfg_inst mi, tms_inst ti
          where :qa_enum = u.da_qa_engr
          and (u.unit_stem=mi.unit_stem and u.unit_da_suf = mi.unit_da_suf)
          and mi.mfg_inst_num = ti.mfg_inst_num;
      endif;
    end if;
  end if;
}

```



```

endif;
unloadtable relt_docs
begin
  tmscntr := select cmpnt_nom = c.com_nom_type + c.com_nom_num
    from component c where c.unit_stem = :relt_docs.part_no and
      c.unit_da_suf = :relt_docs.sffx;
  inquire_ingres (rcount = rowcount);
  if rcount = 0 then
    tmscntr := select h_wpn_code = w.wpn_pgm_code,
      h_tcfg = w.wpn_test_cfg, h_modno = w.wpn_mod_num from weapon w
      where :relt_docs.part_no = w.unit_stem
        and :relt_docs.sffx = w.unit_da_suf;
    if h_tcfg = '' then
      cmpnt_nom := h_wpn_code + '-' + h_modno;
    else
      cmpnt_nom := h_wpn_code + '/' + h_tcfg + '-' + h_modno;
    endif;
  endif;
  tmscntr := select u_desc = u_d.unit_desc from unit_desc u_d
    where :relt_docs.part_no = u_d.unit_stem
      and :relt_docs.sffx = u_d.unit_da_suf;
  relt_docs.description := u_desc;
  if cmpnt_nom = '' then
    relt_docs.subj := relt_docs.part_no;
  else
    relt_docs.subj := cmpnt_nom
  endif;
  relt_docs.qa_engr := qa_name;
  cmpnt_nom := '';
  u_desc := '';
end;
endif;
resume field relt_docs.weapon_pgm;
}

'WPN PGM/LIST' =
{
  if subject[1].wpn_pgm = '' then
    answer := prompt 'You must enter a Weapon Program. Hit <CR> to continue';
    resume field subject.wpn_pgm;
  endif;
  message 'Retrieving Information ...';
  tmscntr := select rcount = count(*) from weapon_pgm w_p
    where :subject[1].wpn_pgm = w_p.wpn_pgm_code
      and :subject[1].mod_num = w_p.wpn_mod_num;
  if rcount = 0 then
    answer := prompt 'The weapon program is invalid. Hit <CR> to continue';
    resume field subject.wpn_pgm;
  else
    relt_docs := select part_no = mi.unit_stem from mfg_inst mi, tms_inst ti
      where (:subject[1].wpn_pgm = mi.wpn_pgm_code
        and :subject[1].mod_num = mi.wpn_mod_num)
        and mi.mfg_inst_num = ti.mfg_inst_num;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
      relt_docs := select part_no = mi.unit_stem
        from mfg_inst mi, eq_inst ei, mfg_eqp_inst mei
        where (:subject[1].wpn_pgm = mi.wpn_pgm_code
          and :subject[1].mod_num = mi.wpn_mod_num)
          and mi.mfg_inst_num = mei.mfg_inst_num
    endif;
  endif;
}

```

```

        and mei.eqp_inst_num = ei.eqp_inst_num;
inquire_ingres (rcount = rowcount);
if rcount = 0 then
    answer := prompt 'There are no documents related to that weapon ' +
        'program in PREMIS';
    resume field subject.wpn_pgm;
else
    relt_docs := select part_no = mi.unit_stem, sffx = mi.unit_da_suf,
        er_num = ei.doc_num, type = ei.doc_type,
        iss_num = ei.doc_iss_num from mfg_inst mi, eq_inst ei,
        mfg_eqp_inst mei where (:subject[1].wpn_pgm = mi.wpn_pgm_code
            and :subject[1].mod_num = mi.wpn_mod_num)
            and mi.mfg_inst_num = mei.mfg_inst_num
            and mei.eqp_inst_num = ei.eqp_inst_num;
endif;
else
    relt_docs := select part_no = mi.unit_stem, sffx = mi.unit_da_suf,
        er_num = ti.doc_num, type = ti.doc_type,
        iss_num = ti.doc_iss_num from mfg_inst mi, tms_inst ti
        where (:subject[1].wpn_pgm = mi.wpn_pgm_code
            and :subject[1].mod_num = mi.wpn_mod_num)
            and mi.mfg_inst_num = ti.mfg_inst_num;
endif;
unloadtable relt_docs
begin
    tmscntr := select cmpnt_nom = c.com_nom_type + c.com_nom_num
        from component c where :relt_docs.part_no = c.unit_stem
        and :relt_docs.sffx = c.unit_da_suf;
inquire_ingres (rcount = rowcount);
if rcount = 0 then
    tmscntr := select h_wpn_code = w.wpn_pgm_code,
        h_tcfg = w.wpn_test_cfg, h_modno = w.wpn_mod_num from weapon w
        where :relt_docs.part_no = w.unit_stem
        and :relt_docs.sffx = w.unit_da_suf;
    if h_tcfg = '' then
        cmpnt_nom := h_wpn_code + '-' + h_modno;
    else
        cmpnt_nom := h_wpn_code + '/' + h_tcfg + '-' + h_modno;
    endif;
endif;
tmscntr := select qa_name = p.first_name + ' ' + p.middl_init + ' ' +
    p.last_name from person p, unit u where p.id_num = u.da_qa_engr
    and :relt_docs.part_no = u.unit_stem
    and :relt_docs.sffx = u.unit_da_suf;
tmscntr := select u_desc = u_d.unit_desc from unit_desc u_d
    where :relt_docs.part_no = u_d.unit_stem
    and :relt_docs.sffx = u_d.unit_da_suf;
relt_docs.weapon_pgm := subject[1].wpn_pgm + '-' + subject[1].mod_num;
if cmpnt_nom = '' then
    relt_docs.subj := relt_docs.part_no;
else
    relt_docs.subj := cmpnt_nom;
endif;
relt_docs.qa_engr := qa_name;
relt_docs.description := u_desc;
qa_name := '';
cmpnt_nom := '';
u_desc := '';
end;
endif;

```

```

resume field relt_docs.weapon_pgm;
}

'GET DOC' =
{
  engr_rels_no := relt_docs.er_num;
  rev_num      := relt_docs.iss_num;

/*
  DSL necessary to retrieve TMS CENTER data from PREMIS
*/
if engr_rels_no = '' then
  answer := prompt 'Use LIST by PART or LIST by ENGR to fill table first' +
    ' Hit <CR> to continue';
  resume field subject.nomen;
else
  if rev_num = '' then
    tmscntr := select rev_num = max(eer.doc_iss_num) from qa_engr_rels eer
      where :engr_rels_no = eer.doc_num;
  endif;
  message 'Retrieving Information ...';
  tmscntr := select doc_type = eer.doc_type, trans_date = eer.trans_com_dt,
    message_num = eer.message_num, twx_rcd = eer.twx_receipt,
    m11_dir_name = eer.m11_dir_name, file_cab_loc = eer.file_cab_loc,
    h_stat = eer.status_code from qa_engr_rels eer
    where :engr_rels_no = eer.doc_num and :rev_num = eer.doc_iss_num;
  inquire_ingres(rcount=rowcount);
  if rcount = 0 then
    answer := prompt 'THERE IS NO DOCUMENT with that number in PREMIS; You '+
      'may add a new one. Hit <CR> to continue';
  else
/*
    DSL necessary to retrieve the part number, supplier, equipment number,
    and weapon program from PREMIS
*/
    tmscntr := select rcount = count (*) from eq_inst ei
      where :engr_rels_no = ei.doc_num and :rev_num = ei.doc_iss_num
        and :doc_type = ei.doc_type;
    if rcount = 0 then
      subject := select part_num = mi.unit_stem, suffix = mi.unit_da_suf,
        wpn_pgm = mi.wpn_pgm_code, mod_num = mi.wpn_mod_num,
        supplier = c.company_name, mfg_num = ti.mfg_inst_num
        from mfg_inst mi, tms_inst ti, company c
        where (:engr_rels_no = ti.doc_num and :rev_num = ti.doc_iss_num
          and :doc_type = ti.doc_type) and ti.mfg_inst_num = mi.mfg_inst_num
          and mi.mfg_code = c.mfg_code;
    else
      subject := select part_num = mi.unit_stem, suffix = mi.unit_da_suf,
        wpn_pgm = mi.wpn_pgm_code, mod_num = mi.wpn_mod_num,
        supplier = c.company_name, eqp_num = mei.equip_type+mei.equip_num,
        mfg_num = mei.mfg_inst_num, ei_num = mei.eqp_inst_num
        from eq_inst ei, mfg_eqp_inst mei, mfg_inst mi, company c
        where (:engr_rels_no = ei.doc_num and :rev_num = ei.doc_iss_num
          and :doc_type = ei.doc_type) and ei.eqp_inst_num = mei.eqp_inst_num
          and mei.mfg_inst_num = mi.mfg_inst_num
          and mi.mfg_code = c.mfg_code;
    endif;
/*
    DSL necessary to retrieve the nomenclature from PREMIS
*/
    unloadtable subject

```

```

begin
if subject.part_num != '' then
  tmscntr := select nomenclature = c.com_nom_type + c.com_nom_num from
    component c where :subject.part_num = c.unit_stem
    and :subject.suffix = c.unit_da_suf;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    tmscntr := select h_wpn_code = w.wpn_pgm_code,
      h_modno = w.wpn_mod_num, h_tcfg = w.wpn_test_cfg from weapon w
      where :subject.part_num = w.unit_stem
      and :subject.suffix = w.unit_da_suf;
    if h_tcfg = '' then
      nomenclature = h_wpn_code + '-' + h_modno;
    else
      nomenclature = h_wpn_code + '/' + h_tcfg + '-' + h_modno;
    endif;
  endif;
  subject.nomen := :nomenclature;
  nomenclature := '';
endif;
end;

```

```

/*
DSL necessary to retrieve the qa engineer information from PREMIS
*/

```

```

if subject[1].part_num != '' then
  tmscntr := select qa_engr_name = p.first_name + ' ' + p.middl_init +
    ' ' + p.last_name, qa_orgn = p.asnd_orgn, qa_phone = '(' +
    left(p.phone_num, 3) + ')' + left(shift(p.phone_num, -3), 3) + '-' +
    shift(p.phone_num, -6) from person p, unit u
  where :subject[1].part_num = u.unit_stem
  and :subject[1].suffix = u.unit_da_suf
  and u.da_qa_engr = p.id_num;
  if qa_phone = '()' then
    qa_phone := '';
  endif;
else
  qa_engr_name := '';
  qa_orgn := '';
  qa_phone := '';
endif;

```

```

/*
DSL necessary to retrieve the results report flag from PREMIS
*/

```

```

tmscntr := select rsits_rpt_fl = er.rsit_rpt_fl from eng_eval_rel er
  where er.doc_type = :doc_type and er.doc_num = :engr_rels_no and
  :rev_num = er.doc_iss_num;

```

```

/*
DSL necessary to retrieve the communication center contact data from PREMIS
*/

```

```

tmscntr := select h_cont = p.first_name + ' ' + p.middl_init + ' '
  + p.last_name, com_ctr_orgn = p.asnd_orgn, h_enum = p.id_num,
  com_ctr_phon = '(' + left(p.phone_num, 3) + ')' +
  left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
  from person p, qa_engr_rels eer where eer.com_ctr_cont = p.id_num and
  (eer.doc_type = :doc_type and eer.doc_num = :engr_rels_no and
  :rev_num = eer.doc_iss_num);
com_ctr_cont := h_cont;
if com_ctr_phon = '()' then
  com_ctr_phon := '';
endif;

```

```

/*
DSL necessary to convert the status from PREMIS
*/

```

```

    if h_stat = 'ACA' then
        status = 'ACCEPTABLE WITH CORR. ACTION';
    elseif h_stat = 'ACC' then
        status = 'ACCEPTABLE';
    elseif h_stat = 'AP' then
        status = 'ACTION PENDING';
    elseif h_stat = 'CA' then
        status = 'CONDITIONALLY ACCEPTABLE';
    elseif h_stat = 'FC' then
        status = 'FULL CERTIFICATION';
    elseif h_stat = 'INST' then
        status = 'INSTRUCTIONS';
    elseif h_stat = 'NA' then
        status = 'NOT ACCEPTABLE';
    elseif h_stat = 'NR' then
        status = 'NOT REQUIRED';
    elseif h_stat = 'P' then
        status = 'PLAN';
    elseif h_stat = 'PC' then
        status = 'PARTIAL CERTIFICATION';
    elseif h_stat = 'PEQ' then
        status = 'PLANS EQ';
    elseif h_stat = 'PQS' then
        status = 'QS PLAN';
    elseif h_stat = 'PTMS' then
        status = 'PLANS TMS';
    elseif h_stat = 'R' then
        status = 'REQUIRED';
    elseif h_stat = 'REQ' then
        status = 'RESULTS REPORT EQ';
    elseif h_stat = 'REQP' then
        status = 'RE-EQ PLAN';
    elseif h_stat = 'REQR' then
        status = 'RE-EQ RESULTS REPORT';
    elseif h_stat = 'RETP' then
        status = 'RE-TMS PLAN';
    elseif h_stat = 'RETR' then
        status = 'RE-TMS RESULTS REPORT';
    elseif h_stat = 'REV' then
        status = 'REVOKED';
    elseif h_stat = 'RQS' then
        status = 'RESULTS REPORT QS';
    elseif h_stat = 'RQSP' then
        status = 'RE-QS PLAN';
    elseif h_stat = 'RQSR' then
        status = 'RE-QS RESULTS REPORT';
    elseif h_stat = 'RTMS' then
        status = 'RESULTS REPORT TMS';
    else
        status = h_stat;
    endif;
endif;
endif;
set_forms field '' (reverse(qa_engr_name) = 0, reverse(qa_orgn) = 0);
resume field doc_type;
}

```

```

'ADD/REVISE' =
{
  err_count = 0;
  if engr_rels_no = '' then
    answer := prompt 'You must supply an engineering release number' +
      '. Hit <CR> to continue';
    err_count = err_count + 1;
    resume field engr_rels_no;
  endif;
  if doc_type = '' then
    answer := prompt 'You must supply a document type. Hit <CR> to continue';
    err_count = err_count + 1;
    resume field doc_type;
  endif;
  unloadtable subject
  begin
    if subject.part_num = '' then
      tmscntr := select prtno = c.unit_stem, sfx = c.unit_da_suf
        from component c
        where :subject.nomen = c.com_nom_type + c.com_nom_num;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        tmscntr := select prtno = c.unit_stem, sfx = c.unit_da_suf
          from component c
          where :subject.nomen = c.com_nom_type + ' ' + c.com_nom_num;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
          answer := prompt 'No component with nomenclature = ' +
            :subject.nomen + ' exists in PREMIS. Hit <CR> to continue';
          err_count = err_count + 1;
        else
          subject.part_num := prtno;
          subject.suffix := sfx;
        endif;
      endif;
    else
      tmscntr := select nomenclature = c.com_nom_type + c.com_nom_num from
        component c where :subject.part_num = c.unit_stem
        and :subject.suffix = c.unit_da_suf;
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        tmscntr := select h_wpn_code = w.wpn_pgm_code,
          h_modno = w.wpn_mod_num, h_tcfg = w.wpn_test_cfg from weapon w
          where :subject.part_num = w.unit_stem
          and :subject.suffix = w.unit_da_suf;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
          answer := prompt 'No part with part number = ' +
            :subject.part_num + ' exists in PREMIS. Hit <CR> to continue';
          err_count = err_count + 1;
        else
          if h_tcfg = '' then
            nomenclature = h_wpn_code + '-' + h_modno;
          else
            nomenclature = h_wpn_code + '/' + h_tcfg + '-' + h_modno;
          endif;
        endif;
      endif;
      subject.nomen := nomenclature;
      nomenclature := '';
    endif;
  end;
}

```

```

endif;
if subject.wpn_pgm != '' then
  if subject.mod_num = '' then
    tmscntr := select rcount = count(*) from weapon_pgm wp
              where :subject.wpn_pgm = wp.wpn_pgm_code;
  else
    tmscntr := select rcount = count(*) from weapon_pgm wp
              where :subject.wpn_pgm = wp.wpn_pgm_code
                  and :subject.mod_num = wp.wpn_mod_num;
  endif;
  if rcount = 0 then
    answer := prompt :subject.wpn_pgm + '-' + :subject.mod_num +
                  'is an invalid weapon program. Hit <CR> to continue';
    err_count = err_count + 1;
  endif;
endif;
endif;
if subject.supplier != '' then
  tmscntr := select mfr_code = c.mfg_code from company c
              where :subject.supplier = c.mfg_code
                  or :subject.supplier = c.company_name;
  inquire_ingres (rcount = rowcount);
  if rcount = 0 then
    tmscntr := select mfr_code = c_s.mfg_code from company_syn c_s
              where :subject.supplier = c_s.company_name;
    inquire_ingres(rcount = rowcount);
    if rcount = 0 then
      answer := prompt :subject.supplier + ' is an invalid supplier. '
                  + 'Hit <CR> to continue';
      err_count = err_count + 1;
    else
      subject.cod := :mfr_code;
    endif;
  else
    subject.cod := :mfr_code;
  endif;
endif;
endif;
if err_count = 0 then
  if subject.mfg_num = '' then
    tmscntr := select h_mfg_num = mi.mfg_inst_num from mfg_inst mi
              where :subject.part_num = mi.unit_stem
                  and :subject.suffix = mi.unit_da_suf
                  and :subject.wpn_pgm = mi.wpn_pgm_code
                  and :subject.mod_num = mi.wpn_mod_num
                  and :subject.cod = mi.mfg_code;
    inquire_ingres (rcount = rowcount);
    if rcount = 0 then
      tmscntr := select h_mfg_num = vchar(max(int4(m.mfg_inst_num))+1)
                    from mfg_inst m;
      insert into mfg_inst (mfg_inst_num, unit_stem, unit_da_suf,
                          wpn_pgm_code, wpn_mod_num, mfg_code) values
        (:h_mfg_num, :subject.part_num, :subject.suffix,
         :subject.wpn_pgm, :subject.mod_num, :subject.cod)
    endif;
  else
    h_mfg_num := subject.mfg_num;
    update mfg_inst mi set unit_stem = :subject.part_num,
                          unit_da_suf = :subject.suffix,
                          wpn_pgm_code = :subject.wpn_pgm,
                          wpn_mod_num = :subject.mod_num, mfg_code = :subject.cod
      where :h_mfg_num = mi.mfg_inst_num;
  endif;
endif;

```

```

endif;
if subject.eqp_num = '' then
  rcount := 0;
  tmscntr := select rcount = count(*) from tms_inst ti
    where :engr_rels_no = ti.doc_num and :rev_num = ti.doc_iss_num
      and :doc_type = ti.doc_type and :h_mfg_num = ti.mfg_inst_num;
  if rcount = 0 then
    insert into tms_inst(doc_type, doc_num, doc_iss_num,
      mfg_inst_num) values (:doc_type, :engr_rels_no, :rev_num,
        :h_mfg_num);
  endif;
else
  tmscntr := select h_etype = left(:subject.eqp_num, 2),
    h_eqnum = shift(:subject.eqp_num, -2) from equipment;
  tmscntr := select rcount = count(*) from equipment e
    where :h_etype = e.equip_type and :h_eqnum = e.equip_num;
  if rcount = 0 then
    insert into equipment (equip_type, equip_num)
      values (:h_etype, :h_eqnum);
  endif;
  if :subject.ei_num = '' then
    rcount := 0;
    tmscntr := select h_eqp_num = mei.eqp_inst_num,
      rcount = count(*) from mfg_eqp_inst mei
      where :h_mfg_num = mei.mfg_inst_num
        and :h_etype = mei.equip_type and :h_eqnum = mei.equip_num;
    if rcount = 0 then
      tmscntr := select h_eqp_num =
        vchar(max(int4(m.eqp_inst_num))+1) from mfg_eqp_inst m;
      insert into mfg_eqp_inst (eqp_inst_num, mfg_inst_num,
        equip_type, equip_num) values(:h_eqp_num, :h_mfg_num,
          :h_etype, :h_eqnum);
    endif;
  else
    h_eqp_num := subject.ei_num;
  endif;
  rcount := 0;
  tmscntr := select rcount = count(*) from eq_inst ei
    where :doc_type = ei.doc_type and :engr_rels_no = ei.doc_num
      and :rev_num = ei.doc_iss_num and :h_eqp_num = ei.eqp_inst_num;
  if rcount = 0 then
    insert into eq_inst(doc_type, doc_num, doc_iss_num, eqp_inst_num)
      values(:doc_type, :engr_rels_no, :rev_num, :h_eqp_num);
  endif;
endif;
endif;
end;
tmscntr := select h_stat = s.eer_status from eer_status s
  where s.eer_status = :status;
inquire_ingres(rcount = rowcount);
if rcount = 0 then
  if status = 'ACCEPTABLE WITH CORR. ACTION' then
    h_stat = 'ACA';
  elseif status = 'ACCEPTABLE' then
    h_stat = 'ACC';
  elseif status = 'ACTION PENDING' then
    h_stat = 'AP';
  elseif status = 'CONDITIONALLY ACCEPTABLE' then
    h_stat = 'CA';
  elseif status = 'FULL CERTIFICATION' then

```



```

        h_stat = 'FC';
    elseif status = 'INSTRUCTIONS' then
        h_stat = 'INST';
    elseif status = 'NOT ACCEPTABLE' then
        h_stat = 'NA';
    elseif status = 'NOT REQUIRED' then
        h_stat = 'NR';
    elseif status = 'PLAN' then
        h_stat = 'P';
    elseif status = 'PARTIAL CERTIFICATION' then
        h_stat = 'PC';
    elseif status = 'PLANS EQ' then
        h_stat = 'PEQ';
    elseif status = 'QS PLAN' then
        h_stat = 'PQS';
    elseif status = 'PLANS TMS' then
        h_stat = 'PTMS';
    elseif status = 'REQUIRED' then
        h_stat = 'R';
    elseif status = 'RESULTS REPORT EQ' then
        h_stat = 'REQ';
    elseif status = 'RE-EQ PLAN' then
        h_stat = 'REQP';
    elseif status = 'RE-EQ RESULTS REPORT' then
        h_stat = 'REQR';
    elseif status = 'RE-TMS PLAN' then
        h_stat = 'RETP';
    elseif status = 'RE-TMS RESULTS REPORT' then
        h_stat = 'RETR';
    elseif status = 'REVOKED' then
        h_stat = 'REV';
    elseif status = 'RESULTS REPORT QS' then
        h_stat = 'RQS';
    elseif status = 'RE-QS PLAN' then
        h_stat = 'RQSP';
    elseif status = 'RE-QS RESULTS REPORT' then
        h_stat = 'RQSR';
    elseif status = 'RESULTS REPORT TMS' then
        h_stat = 'RTMS';
    elseif status = 'TMS' then
        h_stat = 'TMS';
    elseif status = '' then
        h_stat = '';
    else
        h_stat = '';
        answer := 'The status you have input is invalid and will not be ' +
            'stored. Hit <CR> to continue';
    endif;
endif;
if com_ctr_cont != '' then
    if h_cont = com_ctr_cont then
        com_ctr_enum = h_enum
    else
        tmscntr := select com_ctr_enum = p.id_num from person p
            where p.last_name = :com_ctr_cont and p.asnd_orgn = :com_ctr_orgn;
        inquire_ingres(rcount = rowcount);
        if rcount = 0 then
            com_ctr_enum = '';
            answer := prompt com_ctr_cont + ' is not found in PREMIS working ' +
                'for orgn ' + com_ctr_orgn + '. Hit <CR> to continue';
        else
            com_ctr_enum = tmscntr;
        end if;
    end if;
end if;

```

```

err_count = err_count + 1;
else
  tmscntr := select com_ctr_cont = p.first_name + ' ' + p.middl_init
    + ' ' + p.last_name, com_ctr_orgn = p.asnd_orgn,
    com_ctr_phon = '(' + left(p.phone_num, 3) + ')' +
    left(shift(p.phone_num, -3), 3) + '-' + shift(p.phone_num, -6)
  from person p where :com_ctr_enum = p.id_num;
endif;
endif;
endif;
if err_count = 0 then
  tmscntr := select rcount = count(*) from qa_engr_rels eer
    where :enr_rels_no = eer.doc_num and eer.doc_type = :doc_type
    and :rev_num = eer.doc_iss_num;
  if rcount = 0 then
    if rev_num = '0' then
      tmscntr := select rcount = count(*) from qa_engr_rels eer
        where :enr_rels_no = eer.doc_num and eer.doc_type = :doc_type
        and :rev_num = '';
      if rcount = 0 then
        insert into qa_engr_rels (doc_type, doc_num, doc_iss_num,
          message_num, twx_receipt, com_ctr_cont, trans_com_dt,
          file_cab_loc, m11_dir_name, mfg_inst_num, status_code)
          values (:doc_type, :enr_rels_no, :rev_num, :message_num,
            :twx_recd, :com_ctr_enum, :trans_date, :file_cab_loc,
            :m11_dir_name, :h_mfg_num, :h_stat);
        answer := prompt 'Document added to PREMIS. Hit <CR> to ' +
          'continue';
      else
        update qa_engr_rels er set message_num = :message_num,
          twx_receipt = :twx_recd, com_ctr_cont = :com_ctr_enum,
          trans_com_dt = :trans_date, file_cab_loc = :file_cab_loc,
          m11_dir_name = :m11_dir_name, mfg_inst_num = :h_mfg_num,
          status_code = :h_stat, doc_iss_num = :rev_num
          where er.doc_num = :enr_rels_no and er.doc_type = :doc_type
          and er.doc_iss_num = '';
        update eq_inst ei set doc_iss_num = :rev_num
          where ei.doc_num = :enr_rels_no and ei.doc_type = :doc_type
          and ei.doc_iss_num = '';
        answer := prompt 'Document revised in PREMIS. Hit <CR> to ' +
          'continue';
      endif;
    else
      insert into qa_engr_rels (doc_type, doc_num, doc_iss_num,
        message_num, twx_receipt, com_ctr_cont, trans_com_dt,
        file_cab_loc, m11_dir_name, status_code)
        values (:doc_type, :enr_rels_no, :rev_num, :message_num,
          :twx_recd, :com_ctr_enum, :trans_date, :file_cab_loc,
          :m11_dir_name, :h_stat);
      answer := prompt 'Document added to PREMIS. Hit <CR> to continue';
    endif;
  else
    update qa_engr_rels er set message_num = :message_num,
      twx_receipt = :twx_recd, com_ctr_cont = :com_ctr_enum,
      trans_com_dt = :trans_date, file_cab_loc = :file_cab_loc,
      m11_dir_name = :m11_dir_name,
      status_code = :h_stat where er.doc_num = :enr_rels_no and
      er.doc_type = :doc_type and :rev_num = er.doc_iss_num;
    answer := prompt 'Document revised in PREMIS. Hit <CR> to continue';
  endif;
endif;

```

```

if rsits_rpt_fl != '' then
  tmscntr := select rcount = count (*) from eng_eval_rel eer
    where eer.doc_num = :engr_rels_no and eer.doc_type = :doc_type
      and eer.doc_iss_num = :rev_num;
  if rcount = 0 then
    if rev_num = '0' then
      tmscntr := select h_dwg_no = eer.doc_num from eng_eval_rel eer
        where eer.doc_num = :engr_rels_no and eer.doc_type = :doc_type
          and eer.doc_iss_num = '';
      inquire_ingres(rcount = rowcount);
      if rcount = 0 then
        insert into eng_eval_rel(doc_type, doc_num, doc_iss_num,
          rsit_rpt_fl) values (:doc_type, :engr_rels_no, :rev_num,
            :rsits_rpt_fl);
      else
        update eng_eval_rel eer set rsit_rpt_fl = :rsits_rpt_fl,
          doc_iss_num = :rev_num where eer.doc_num = :engr_rels_no
            and eer.doc_type = :doc_type and :rev_num = '';
      endif;
    else
      insert into eng_eval_rel(doc_type, doc_num, doc_iss_num,
        rsit_rpt_fl) values (:doc_type, :engr_rels_no, :rev_num,
          :rsits_rpt_fl);
      endif;
    else
      update eng_eval_rel eer set rsit_rpt_fl = :rsits_rpt_fl
        where eer.doc_num = :engr_rels_no
          and eer.doc_type = :doc_type and :rev_num = eer.doc_iss_num;
      endif;
    endif;
  endif;
}

'CLEAR' =
{
  clear field all;
  set_forms field '' (reverse(qa_engr_name) = 1, reverse(qa_orgn) = 1);
  resume field engr_rels_no;
}

'HELP', key frskey1 =
{
  help_forms (subject = 'Retrieve TMS Center data',
    file = 'db1:[dbaprdb.help]tmscntr.help');
}

'XIT', key frskey3 =
{
  RETURN;
}

'STOP', key frskey2 =
{
  EXIT;
}

```

/*

Procedure for TOPFRAME frame
Written by Norman Stevens

TOPFRAME

*/

Initialize =

{
}

'DATA' =

{
 CALLFRAME DATA;
}

'QA' =

{
 CALLFRAME QAMAIN;
}

'COMPANY' =

{
 CALLFRAME COMPMAIN;
}

'HELP', key frskey1 =

{
 help_forms (subject = 'TOPFRAME',
 file = 'db1:[dbaprd.b.help]topframe.help');
}

'STOP', key frskey2 =

{
 EXIT;
}

/*

FRAME: UNITDOC

UNITDOC

DSL to fill UNITDOC form

Return add_it flag to indicate whether document was
successfully added to the unit_doc database table

Written by John Orman 7/14/88

*/

initialize (

```
    add_it = vchar(3),
    unit_count = integer,
    row_count = integer,
    rcount = integer4
  ) = {
    add_it = 'NO' ;
    resume field unit_table ;
  }
```

'ADD_UNITS' = {

/* ***** DOCUMENTS ***** */

/* add documents to unit_doc table

insert part numbers

/* cannot use unloadtable unit_table - cannot nest unloadtable commands,
and PREMIS is using unloadtable in frame NEWDOC to get
to UNITDOC */

/* find number of rows entered in unit_table */

inquire_forms table unitdoc (unit_count = datarows(unit_table)) ;

/* cannot access more rows than are displayed on form -
else will get range error */

if unit_count <= 5 then

/* loop through rows in unit_table */

row_count := 1 ;

while row_count <= unit_count do

/* loop through rows of unit_table */

/* extract document type and number */

if unit_table[row_count].unit_stem != '' then

/* user has entered all mandatory fields */

/* update unit_doc table */

/* know that document is not in unit_doc table */

insert into unit_doc (doc_type, doc_num, doc_iss_num,
unit_stem, unit_da_suf)

values (

:doc_type,

:doc_num,

:doc_iss_num,

:unit_table[row_count].unit_stem,

:unit_table[row_count].unit_da_suf) ;

/* see if found part in component table */

inquire_ingres (rcount = rowcount) ;

if rcount != 0 then

/* successful insertion */

add_it = 'YES' ;

message 'Unit ' + :unit_table[row_count].unit_stem + '-' +
:unit_table[row_count].unit_da_suf +
' was inserted into the document database' ;

sleep 3 ;

else

/* unsuccessful insertion. */

message 'Error in inserting unit ' +
:unit_table[row_count].unit_stem + '-' +
:unit_table[row_count].unit_da_suf +
' into the document database' ;

```

        sleep 3 ;
    endif ;
else
/*      the unit stem was blank */
    message 'The unit number must not be blank' ;
    sleep 3 ;
    endif ;
    row_count := :row_count + 1 ;
endwhile ;
else
/*      more entries than can be accessed by indexing */
    message 'Only 5 part numbers may be entered' ;
    sleep 3 ;
    message 'No units were added to the document database' ;
    sleep 3 ;
endif ;
    RETURN add_it;
}
'HELP', key frskey1 =
{
    help_forms (subject = 'add a document to the database',
        file = 'db1:[dbaprd.b.help]unitdoc.help');
}
'XIT', key frskey3 =
{
    RETURN add_it;
}
'STOP', key frskey2 =
{
    EXIT;
}

```

/*

frame: WEAPON WEAPON
DRIVER FOR REPORT TO LIST ALL PARTS BY WEAPON
***** NOT IMPLEMENTED AS OF 9/27/88 *****

written by John Orman 9/27/88

```
*/
initialize (
) = {
/*          ***** DISPLAY SCREEN ***** */
    redisplay ;
/*          ***** SHOW THAT SCREEN NOT IMPLEMENTED ***** */
    message '*** This report has not been implemented ***' ;
    sleep 5 ;
    RETURN ;
}
'DISPLAY' = {
}
'FILE' = {
}
'PRINTER' =
{
}
'HELP', key frskey1 =
{
    help_forms (subject = 'List parts by weapon',
        file = 'db1:[dbaprdb.help]weapon.help');
}
'XIT', key frskey3 =
{
    RETURN ;
}
'STOP', key frskey2 =
{
    EXIT ;
}
```

/*

Procedure for WPNSYS frame
Written by Norman Stevens

WPNSYS

*/

initialize (rcount = integer1, h_nomen = vchar(15), h_wpgm = vchar(4),
 h_modno = vchar(2), h_cfg = vchar(7), h_desc = vchar(32)) =

{
 resume field part_nom;
}

'RETRIEVE' =

{

/*

OSL necessary to retrieve the part number from PREMIS

*/

if part_nom = '' then
 if wpn_pgm = '' then
 message 'You must enter a part nomenclature or a weapon nomenclature';
 sleep 5;

 else

 wpnsys := select part_num = w.unit_stem, suffix = w.unit_da_suf
 from weapon w where :wpn_pgm = w.wpn_pgm_code
 and :mod = w.wpn_mod_num and :cfg = w.wpn_test_cfg;
 inquire_ingres(rcount=rowcount);

 if rcount = 0 then

 message 'No weapon in PREMIS has that nomenclature';
 sleep 5;

 else

 subas := select part_no = g.part_stem, sffx = g.part_da_suf,
 type = eer.doc_type, doc_num = eer.doc_num,
 rev_num = eer.doc_iss_num, rels_dt = eer.issue_dt,
 status = eer.status_code from gozinto g, mfg_inst m,
 qa_engr_rels eer, tms_inst t
 where (:part_num = g.unit_stem and :suffix = g.unit_da_suf)
 and (g.part_stem = m.unit_stem and g.unit_da_suf = m.unit_da_suf)
 and m.mfg_inst_num = t.mfg_inst_num and eer.doc_type =
 t.doc_type and eer.doc_num
 = t.doc_num and eer.doc_iss_num = t.doc_iss_num;

 endif;

 endif;

 else

 wpnsys := select part_num = c.unit_stem, suffix = c.unit_da_suf
 from component c where :part_nom = c.com_nom_type + c.com_nom_num;
 inquire_ingres(rcount=rowcount);

 if rcount = 0 then

 message 'No component in PREMIS has that nomenclature';
 sleep 5;

 else

 subas := select part_no = g.part_stem, sffx = g.part_da_suf,
 type = eer.doc_type, doc_num = eer.doc_num,
 rev_num = eer.doc_iss_num, rels_dt = eer.issue_dt,
 status = eer.status_code from gozinto g, mfg_inst m,
 qa_engr_rels eer, tms_inst t
 where (:part_num = g.unit_stem and :suffix = g.unit_da_suf)
 and (g.part_stem = m.unit_stem and g.unit_da_suf = m.unit_da_suf)
 and m.mfg_inst_num = t.mfg_inst_num and eer.doc_type =
 t.doc_type and eer.doc_num
 = t.doc_num and eer.doc_iss_num = t.doc_iss_num;

 endif;

 endif;


```

/*
DSL necessary to retrieve the part number's of components
*/
unloadtable subas
begin
  h_nomen := '';
  wpnsys := select h_nomen = c.com_nom_type + c.com_nom_num
    from component c where :subas.part_no = c.unit_stem
    and :subas.sffx = c.unit_da_suf;
  inquire_ingres(rcount = rowcount);
  if rcount = 0 then
    wpnsys := select h_wpgm = w.wpn_pgm_code, h_modno = w.wpn_mod_num,
      h_cfg = w.wpn_test_cfg from weapon w
      where w.unit_stem = :subas.part_no and w.unit_da_suf = :subas.sffx;
    if h_cfg = '' then
      h_nomen := h_wpgm + '-' + h_modno;
    else
      h_nomen := h_wpgm + '/' + h_cfg + '-' + h_modno;
    endif;
  endif;
  subas.nomen := h_nomen;
  wpnsys := select h_desc = u.unit_desc from unit_desc u
    where u.unit_stem = :subas.part_no and u.unit_da_suf = :subas.sffx;
  subas.description := h_desc;
end ;
}

'CLEAR' =
{
  clear field all;
  resume field part_nom;
}

'HELP', key frskey1 =
{
  help_forms (subject = 'Retrieve part numbers of components',
    file = 'db1:[dbaprdb.help]wpnsys.help');
}

'XIT', key frskey3 =
{
  RETURN;
}

'STOP', key frskey2 =
{
  EXIT;
}

```

6. Other Procedures (VAX/VMS DCL and INGRES SQL, etc.)

The frames, documented in the previous section, call special VAX/VMS command files (COM) and Standard Query Language (SQL) files to accomplish the necessary functions in the application. In addition, special files are used to define and run reports from the application. These files are documented in the following section and are arranged in alphabetical order.

/*

Report: ALL_ID_RPT.REP Procedure
written by John Orman

*/

/*
*/

Last Modified: 16-sep-88 18:27

```
.NAME    all_id_rpt form
.DATA    all_id_rpt
.SORT    nomenclature : a
         ,part_number : a

.HEADER      report
             .BEGIN RBFSETUP
               .FORMFEEDS
               .FORMAT nomenclature(c15)
               .POSITION nomenclature(0,15)
               .FORMAT part_number(c9)
               .POSITION part_number(15,11)
               .FORMAT prod_agency(c3)
               .POSITION prod_agency(29,11)
               .FORMAT desn_agency(c3)
               .POSITION desn_agency(43,11)
               .FORMAT data_engnr(c30)
               .POSITION data_engnr(57,30)
             .END RBFSETUP
             .BEGIN RBFPTOP
               .LEFT
               .PRINT current_date
               .RIGHT
               .PRINT current_time
               .NEWLINE 1
             .END RBFPTOP
             .BEGIN RBFTITLE
               .NEED 4
               .NEWLINE 1
               .BEGIN BLOCK
                 .TAB 30
                 .UNDERLINE
                 .PRINT "Report of ", $rep_type(c0), " parts by nomenclature"
                 .NOUNDERLINE
                 .NEWLINE
               .END BLOCK
               .BEGIN BLOCK
                 .TAB 30
                 .PRINT "Parts assigned to data engineer = ", $data_eng(c0)
                 .NEWLINE
               .END BLOCK
               .NEWLINE 1
             .END RBFTITLE
             .BEGIN RBFHEAD
               .NEED 3
               .NEWLINE 1
               .BEGIN BLOCK
                 .WITHIN nomenclature
                 .TOP
                 .LINESTART
                 .UNDERLINE
                 .PRINT "Nomenclature"
                 .NOUNDERLINE
```

```

        .NEWLINE
        .TOP
    .END WITHIN
    .WITHIN part_number
        .TOP
        .LINESTART
        .UNDERLINE
            .PRINT "Part_number"
        .NOUNDERLINE
        .NEWLINE
        .TOP
    .END WITHIN
    .WITHIN prod_agency
        .TOP
        .LINESTART
        .UNDERLINE
            .PRINT "Prod_agency"
        .NOUNDERLINE
        .NEWLINE
        .TOP
    .END WITHIN
    .WITHIN desn_agency
        .TOP
        .LINESTART
        .UNDERLINE
            .PRINT "Desn_agency"
        .NOUNDERLINE
        .NEWLINE
        .TOP
    .END WITHIN
    .WITHIN data_engnr
        .TOP
        .LINESTART
        .UNDERLINE
            .PRINT "Data_engnr"
        .NOUNDERLINE
        .NEWLINE
        .TOP
    .END WITHIN
    .END BLOCK
    .NEWLINE 1
.END RBFHEAD

.HEADER
    .BEGIN RBFPTOP
        .LEFT
        .PRINT current_date
        .UNDERLINE
        .CENTER
        .PRINT "Report of ", $rep_type(c0), " parts by nomenclature"
        .NOUNDERLINE
        .RIGHT
        .PRINT current_time
        .NEWLINE 1
        .BEGIN BLOCK
            .TAB 26
            .PRINT "Parts assigned to data engineer = ", $data_eng(c0)
            .NEWLINE
        .END BLOCK
    .END RBFPTOP

```

```

.BEGIN RBFHEAD
  .NEED 3
  .NEWLINE 1
  .BEGIN BLOCK
    .WITHIN nomenclature
      .TOP
      .LINESTART
      .UNDERLINE
      .PRINT "Nomenclature"
      .NOUNDERLINE
      .NEWLINE
      .TOP
    .END WITHIN
    .WITHIN part_number
      .TOP
      .LINESTART
      .UNDERLINE
      .PRINT "Part_number"
      .NOUNDERLINE
      .NEWLINE
      .TOP
    .END WITHIN
    .WITHIN prod_agency
      .TOP
      .LINESTART
      .UNDERLINE
      .PRINT "Prod_agency"
      .NOUNDERLINE
      .NEWLINE
      .TOP
    .END WITHIN
    .WITHIN desn_agency
      .TOP
      .LINESTART
      .UNDERLINE
      .PRINT "Desn_agency"
      .NOUNDERLINE
      .NEWLINE
      .TOP
    .END WITHIN
    .WITHIN data_engnr
      .TOP
      .LINESTART
      .UNDERLINE
      .PRINT "Data_engnr"
      .NOUNDERLINE
      .NEWLINE
      .TOP
    .END WITHIN
  .END BLOCK
  .NEWLINE 1
.END RBFHEAD

.FOOTER
  .BEGIN RBFPBOT
    .NEWLINE 1
    .CENTER
    .PRINT "-",page_number(f4)," -"
    .NEWLINE 1
  .END RBFPBOT

```

.DETAIL

.BEGIN BLOCK

```
.WITHIN nomenclature
  .TOP
  .LINESTART
  .PRINT nomenclature
  .NEWLINE
.END WITHIN
.WITHIN part_number
  .TOP
  .LINESTART
  .PRINT part_number
  .NEWLINE
.END WITHIN
.WITHIN prod_agency
  .TOP
  .LINESTART
  .PRINT prod_agency
  .NEWLINE
.END WITHIN
.WITHIN desn_agency
  .TOP
  .LINESTART
  .PRINT desn_agency
  .NEWLINE
.END WITHIN
.WITHIN data_engnr
  .TOP
  .LINESTART
  .PRINT data_engnr
  .NEWLINE
.END WITHIN
```

.END BLOCK

.FOOTER

report

```
.NEED 7
.NEWLINE 1
.TAB 0
.PRINT "=====", "=====
.NEWLINE 1
.PRINT "Grand Totals: REPORT"
.NEWLINE 1
.NEWLINE 1
.BEGIN BLOCK
  .PRINTLINE "N:"
  .WITHIN part_number
    .TOP
    .LINESTART
    .PRINT count(part_number)
    .NEWLINE
    .TOP
  .ENDWITHIN
.END BLOCK
.TAB 0
.PRINT "=====", "=====
.NEWLINE 1
.NEWLINE 1
```



```

UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
AND
UNIT.DA_DATA_ENGR = DA_EMPL.EMPL_NUM
AND
DA_EMPL.ID_NUM = PERSON.ID_NUM
AND NOT EXISTS
  (SELECT *
   FROM UNIT_DOC,
        DATA_DOCUMNT
   WHERE
     UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
     AND
     UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
     AND
     UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
     AND
     UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
     AND
     UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

\G\R
/*

SOME WEAPONS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
CHANGE MADE BY JOHN ORMAN 9/14/89

*/

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )
SELECT
  WEAPON.WPN_PGM_CODE +
  '/' +
  WEAPON.WPN_TEST_CFG +
  '-' +
  WEAPON.WPN_MOD_NUM,
  WEAPON.UNIT_STEM +
  WEAPON.UNIT_DA_SUF
FROM
  WEAPON
WHERE NOT EXISTS
  (SELECT * FROM UNIT WHERE
    UNIT.UNIT_STEM = WEAPON.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF)
AND NOT EXISTS
  (SELECT *
   FROM UNIT_DOC,
        DATA_DOCUMNT
   WHERE
     UNIT_DOC.UNIT_STEM = WEAPON.UNIT_STEM
     AND
     UNIT_DOC.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
     AND
     UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
     AND
     UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
     AND
     UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

\G\R
/*

NOW INSERT ALL COMPONENTS INTO TABLE

*/

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,

```



```

                                PART_NUMBER,
                                PROD_AGENCY,
                                DESN_AGENCY,
                                DATA_ENGR )
SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ', ' +
    PERSON.MIDDL_INIT +
    ', ' +
    PERSON.LAST_NAME
FROM
    UNIT,
    COMPONENT,
    DA_EML,
    PERSON
WHERE
    UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EML.EMPL_NUM
    AND
    DA_EML.ID_NUM = PERSON.ID_NUM
    AND NOT EXISTS
        (SELECT *
         FROM UNIT_DOC,
              DATA_DOCUMNT
         WHERE
             UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
             AND
             UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
             AND
             UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
             AND
             UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
             AND
             UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

\G\R
/*

SOME COMPONENTS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
CHANGE MADE BY JOHN ORMAN 9/14/89

*/

```

INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )
SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF
FROM
    COMPONENT
WHERE NOT EXISTS
    (SELECT * FROM UNIT WHERE
     UNIT.UNIT_STEM = COMPONENT.UNIT_STEM

```

```

AND
UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF)
AND NOT EXISTS
( SELECT *
  FROM UNIT_DOC,
       DATA_DOCUMNT
  WHERE
    UNIT_DOC.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT_DOC.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT_DOC.DOC_NUM = DATA_DOCUMNT.DOC_NUM
    AND
    UNIT_DOC.DOC_TYPE = DATA_DOCUMNT.DOC_TYPE
    AND
    UNIT_DOC.DOC_ISS_NUM = DATA_DOCUMNT.DOC_ISS_NUM ) ;

```

```

\G\R
\Q
$ !
$ GOTO TABLE_BUILT
$ ALL_ID:
$ !
$ !
$ !
$ !
$ SQL UPREMIS
/*

```

WEAPONS

```

*/
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGNR )

```

```

SELECT
  WEAPON.WPN_PGM_CODE +
  '/' +
  WEAPON.WPN_TEST_CFG +
  '-' +
  WEAPON.WPN_MOD_NUM,
  WEAPON.UNIT_STEM +
  WEAPON.UNIT_DA_SUF,
  UNIT.PRD_MFG_CODE,
  UNIT.DSN_MFG_CODE,
  PERSON.FIRST_NAME +
  ',' +
  PERSON.MIDDL_INIT +
  ',' +
  PERSON.LAST_NAME
FROM
  UNIT,
  WEAPON,
  DA_EEMPL,
  PERSON
WHERE
  UNIT.UNIT_STEM = WEAPON.UNIT_STEM
  AND
  UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF
  AND
  UNIT.DA_DATA_ENGR = DA_EEMPL.EMPL_NUM
  AND

```

DA_EMPL.ID_NUM = PERSON.ID_NUM ;

\G\R
/*

SOME WEAPONS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT
CHANGE MADE BY JOHN ORMAN 9/14/89

*/

```
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )
SELECT
    WEAPON.WPN_PGM_CODE +
    '/' +
    WEAPON.WPN_TEST_CFG +
    '-' +
    WEAPON.WPN_MOD_NUM,
    WEAPON.UNIT_STEM +
    WEAPON.UNIT_DA_SUF
FROM
    WEAPON
WHERE NOT EXISTS
    (SELECT * FROM UNIT WHERE
      UNIT.UNIT_STEM = WEAPON.UNIT_STEM
      AND
      UNIT.UNIT_DA_SUF = WEAPON.UNIT_DA_SUF) ;
```

\G\R
/*

NOW INSERT ALL COMPONENTS INTO TABLE

*/

```
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER,
                             PROD_AGENCY,
                             DESN_AGENCY,
                             DATA_ENGNR )
SELECT
    COMPONENT.COM_NOM_TYPE +
    COMPONENT.COM_NOM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF,
    UNIT.PRD_MFG_CODE,
    UNIT.DSN_MFG_CODE,
    PERSON.FIRST_NAME +
    ',' +
    PERSON.MIDDLE_INIT +
    ',' +
    PERSON.LAST_NAME
FROM
    UNIT,
    COMPONENT,
    DA_EMPL,
    PERSON
WHERE
    UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
    AND
    UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF
    AND
    UNIT.DA_DATA_ENGR = DA_EMPL.EMPL_NUM
    AND
    DA_EMPL.ID_NUM = PERSON.ID_NUM ;
```

\G\R
/*

SOME COMPONENTS MAY NOT BE IN UNIT TABLE - ADD THEM TO REPORT

*/

```
INSERT INTO ALL_ID_RPT      (NOMENCLATURE,
                             PART_NUMBER )
```

```
SELECT
```

```
    COMPONENT.COM_NDM_TYPE +
    COMPONENT.COM_NDM_NUM,
    COMPONENT.UNIT_STEM +
    COMPONENT.UNIT_DA_SUF
```

```
FROM
```

```
    COMPONENT
```

```
WHERE NOT EXISTS
```

```
    (SELECT * FROM UNIT WHERE
      UNIT.UNIT_STEM = COMPONENT.UNIT_STEM
      AND
      UNIT.UNIT_DA_SUF = COMPONENT.UNIT_DA_SUF) ;
```

```
|G|R
```

```
|Q
```

```
$ !
```

```
$ TABLE_BUILT:
```

```
$ !
```

```
$ ALL_ID_RPT TABLE BUILT - RUN REPORT
```

```
$ !
```

```
$ report -fall_ID.RPT -v61 UPREMIS ALL_ID_RPT (data_eng = "ALL",rep_type = "'REP_TYPE'")
```

```
$ !
```

```
$ SEND TO MYPRINTER
```

```
$ !
```

```
$ ODB1:[DBAPRDB.SOURCE]PRINT_DISPOSE ALL_ID.RPT
```

```
$ !
```

```
$ EXIT
```

```

/*      CREATE TALL.SQL Procedure
Create used by company report
Tables: ALLCODES
      by
Donna Eaton
*/

```

```

DROP ALLCODES; \G
create table allcodes as
select MFG_CODE=c.mfg_code,
      COMPANY_NAME=c.company_name, STATUS=' ',
      ALT_CODE=' ',
      OLD_NAME=' ',
      STRT_ADDR1= a.strt_addr1,
      STRT_ADDR2 =a.strt_addr2,
      COUNTRY_NAME = a.country_name,
      COMPANY_NAM2 = c.company_nam2,
      NOTES = C.NOTES,
      CITY_NAME=a.city_name,
      STATE_CODE=a.state_code,
      ZIP_CODE=a.zip_code
from company c, address a
where
      A.ADDRESS_ID=C.MFG_CODE

```

```

;
INSERT INTO ALLCODES
select MFG_CODE=h.current_code,
      COMPANY_NAME=c.company_name, STATUS='was',
      ALT_CODE= h.old_mfg_code,
      OLD_NAME = h.old_name, STRT_ADDR1=' ',
      CITY_NAME=' ',
      STATE_CODE=' ',
      STRT_ADDR2 = ' ',
      COUNTRY_NAME = ' ',
      COMPANY_NAM2 = ' ',
      NOTES = ' ',
      ZIP_CODE=' '
from company c, address a, company_hist h
where
      H.current_code=C.MFG_CODE AND
      A.ADDRESS_ID=C.MFG_CODE

```

```

;
INSERT INTO ALLCODES
select MFG_CODE=H.old_mfg_code,
      COMPANY_NAME=h.old_name, STATUS='use',
      ALT_CODE=h.current_code,
      OLD_NAME=c.company_name, STRT_ADDR1=' ',
      CITY_NAME=' ',
      STATE_CODE=' ',
      STRT_ADDR2 = ' ',
      COUNTRY_NAME = ' ',
      COMPANY_NAM2 = ' ',
      NOTES = ' ',
      ZIP_CODE=' '
from company c, address a, company_hist h
where
      H.CURRENT_CODE=C.MFG_CODE AND
      A.ADDRESS_ID=C.MFG_CODE

```

COMPANY_C.RW
 COMPANY_SORTED BY CODES
 written by Caleb C. Crump
 and Donna S. Eaton

This report uses as input the temporary table created as allcodes.
 The table allcodes is a select of all company records
 with column STATUS = "n";
 plus a union of the
 history records showing the current codes and old code and name
 with column STATUS = "was";
 plus a union of the
 history records showing the old code and old
 name with a pointer to the current code and name
 with column STATUS = "use".

The command file to run this report is FULLCODES.COM
 The sql file to be included by FULLCODES.COM is CREATETALL.SQL
 All of this is executed from PREMIS vI.1 automatically from the
 Company Menu choice for Company Reports. Both reports are
 produced and printed when Company Reports is chosen.

This report is the same as company_n except for the
 name, sort and output lines.
 The title page and 1st comment line must also be changed.
 The drawing number in the page header must also be changed.

```

*/
/*change the next 3 lines to make the company_n report */
.name company_c
.sort mfg_code
.output c_all.lis

/*
.data allcodes
.lm 0
.rm 130
.pl 66
.formfeeds

/* title page
.header report

.nl 12
.ulc ""
.ul .ce .pr "Report of Company Sites"
.nl4
.ce .pr "by"
.nl4
/* change the following line for the company_n Company Name report */
.ce .pr "Company Code"

/*
.nl 8
.ulc ""
.ul .ce .pr current_date
.noul
.np 1

.header page

.nl 2
/* change the following line for the company_n Company Name report */

```

```

/*
    .rt .pr "AJ373115-001"

    .nl 1
    .rt .pr page_number
    .nl 1
    .rt .pr "Issue A"
    .nl 2
    .ulc "_"
    .ul
    .tab 0          .pr "CODE",
    .tab 10         .pr "COMPANY NAME(/AND DIVISION)",
    .tab 55         .pr "ADDRESS(/ADDRESS EXTENSION)",
    .tab 90         .pr "CITY(/COUNTRY)",
    .tab 111        .pr "STATE ",
    .tab 117        .pr "ZIP"
    .noul
    .nl 2

    .detail
    .need 3

    .if status = "" .then
    .tab 0          .pr mfg_code (c3),
    .tab 10         .pr company_name (c34),
    .tab 55         .pr strt_addr1 (c34),
    .tab 90         .pr city_name (c20),
    .tab 111        .pr state_code (c2),
    .tab 117        .pr zip_code (c9)
        .if company_nam2 != "" or strt_addr2 != ""
            or country_name != ""
        .then
        .nl
        .tab 10     .pr company_nam2(c34),
        .tab 55     .pr strt_addr2 (c34),
        .tab 90     .pr country_name (c20)
        .endif
    .endif

    .if status = "was" .then
    .tab 5          .pr status (c3),
    .tab 15         .pr alt_code (c3),
    .tab 20         .pr old_name (c34)
    .endif

    .if status = "use" .then
    .tab 0          .pr mfg_code (c3),
    .tab 10         .pr company_name (c34),
    .tab 52         .pr status (c3),
    .tab 57         .pr alt_code (c3),
    .tab 62         .pr old_name (c34)
    .endif

    .nl 1

```

/* End of Report

*/

```

/*
COMPANY_N.RW
COMPANY_SORTED BY NAME
written by Caleb C. Crump
and Donna S. Eaton

```

This report uses as input the temporary table created as allcodes.
The table allcodes is a select of all company records
with column STATUS = "";
plus a union of the
history records showing the current codes and old code and name
with column STATUS = "was";
plus a union of the
history records showing the old code and old
name with a pointer to the current code and name
with column STATUS = "use".

The command file to run this report is FULLCODES.COM
The sql file to be included by FULLCODES.COM is CREATETALL.SQL
All of this is executed from PREMIS v1.1 automatically from the
Company Menu choice for Company Reports. Both reports are
produced and printed when Company Reports is chosen.

This report is the same as company_n except for the
name, sort and output lines.
The title page and 1st comment line must also be changed.
The drawing number in the page header must also be changed.

```

*/
/*change the next 3 lines to make the company_n report */
.name company_n
.sort company_name
.output c_all_name.lis
/*
.data allcodes
.lm 0
.rm 130
.pl 66
.formfeeds
/* title page */
.header report

.nl 12
.ulc ""
.ul .ce .pr "Report of Company Sites"
.nl4
.ce .pr "by"
.nl4
/* change the following line for the company_n Company Name report */
.ce .pr "Company Name"
/*
.nl 8
.ulc ""
.ul .ce .pr current_date
.noul
.np 1

.header page

.nl 2
/* change the following line for the company_n Company Name report */

```



```

.rt .pr "AJ373115-002"
/*
.nl 1
.rt .pr page_number
.nl 1
.rt .pr "Issue A"
.nl 2
.ulc "-"
.ul
.tab 0 .pr "CODE",
.tab 10 .pr "COMPANY NAME(/AND DIVISON)",
.tab 55 .pr "ADDRESS(/ADDRESS EXTENSION)",
.tab 90 .pr "CITY(/COUNTRY)",
.tab 111 .pr "STATE ",
.tab 117 .pr "ZIP"
.noul
.nl 2
.detail
.need 3

.if status = "" .then
.tab 0 .pr mfg_code (c3),
.tab 10 .pr company_name (c34),
.tab 55 .pr strt_addr1 (c34),
.tab 90 .pr city_name (c20),
.tab 111 .pr state_code (c2),
.tab 117 .pr zip_code (c9)
.if company_nam2 != "" or strt_addr2 != ""
or country_name != ""
.then
.nl
.tab 10 .pr company_nam2(c34),
.tab 55 .pr strt_addr2 (c34),
.tab 90 .pr country_name (c20)
.endif
.endif

.if status = "was" .then
.tab 5 .pr status (c3),
.tab 15 .pr alt_cou (c3),
.tab 20 .pr old_name (c34)
.endif

.if status = "use" .then
.tab 0 .pr mfg_code (c3),
.tab 10 .pr company_name (c34),
.tab 52 .pr status (c3),
.tab 57 .pr alt_code (c3),
.tab 62 .pr old_name (c34)
.endif

.nl 1

```

/* End of Report

```

$ |
$ |
$ SET VERIFY
$ |
$ |
$ | ***** PREMIS *****
$ | RUN INGRES SQL TO CREATE A TABLE (ALLCODES)
$ | FOR INPUT TO THE COMPANY CODES REPORTS. THEN
$ | RUN INGRES REPORT TO PRODUCE THE REPORTS
$ | C_ALL.LIS AND C_ALL_NAME.LIS AND PRINT THEM.
$ | SET UP BY DONNA EATON 9/15/89
$ |
$ SQL UPREMIS -C3 -T3 >FULLCODES.LLIS
\I DB1:[DBAPRDB.SOURCE]CREATETALL.SQL
\G
$ DEFINE/PROC ING SET UPREMIS "SET LOCKMODE SESSION WHERE READLOCK = NOLOCK"
$ REPORT UPREMIS -R COMPANY_C
$ REPORT UPREMIS -R COMPANY_N
$ ODB1:[DBAPRDB.SOURCE]PRINT DISPOSE C_ALL.LIS
$ ODB1:[DBAPRDB.SOURCE]PRINT DISPOSE C_ALL_NAME.LIS

```

/*

NOMEN.VIEW Code
by
Norman Stevens

Report file created: NOMEN.RPT, prints at MYPRINTER
Report is RBF report: NOMEN_RPT
Procedure is: NOMEN_RPT.COM, called in frame QAREPORT.

*/

```
create view nomen_view (nomenclature, part_number, qa_orgn, qa_Engineer, pA,  
  wpn_Pgm, description) as select c.com_nom_type + c.com_nom_num,  
  u.unit_stem, p.asnd_orgn, p.last_name, n.doe_code,  
  mi.wpn_pgm_code + '-' + mi.wpn_mod_num, u_d.unit_desc  
from unit u, component c, unit_desc u_d, person p, mfg_inst mi, nwc_agency n  
where u.unit_stem = c.unit_stem and u.unit_da_suf = c.unit_da_suf  
  and u.unit_stem = u_d.unit_stem and u.unit_da_suf = u_d.unit_da_suf  
  and u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf  
  and u.da_qa_engr = p.id_num and u.prn_mfg_code = n.mfg_code;
```


/*

PA.VIEW
by
Norman Stevens

Report file created: PART PA.RPT, prints at MYPRINTER
Report is RBF report: PA_RPT
Procedure is: PA_RPT.COM, called in frame QAREPORT.

*/

```
create view pa_view (pa, wpn_pgm, nomenclature, part_number, qa_orgn,  
  qa_engr_name, da_desn_orgn, da_desn_engr, description) as select  
  m.company_name, mi.wpn_pgm_code + '-' + mi.wpn_mod_num,  
  c.com_nom_type + c.com_nom_num, u.unit_stem, p.asnd_orgn, p.last_name,  
  d.asnd_orgn, d.last_name, u_d.unit_desc  
from unit u, component c, unit_desc u_d, person p, person d, mfg_inst  
mi, company m  
where u.unit_stem = c.unit_stem and u.unit_da_suf = c.unit_da_suf  
  and u.unit_stem = u_d.unit_stem and u.unit_da_suf = u_d.unit_da_suf  
  and u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf  
  and u.da_qa_engr = p.id_num and u.da_desn_engr = d.id_num  
  and u.prd_mfg_code = m.mfg_code;
```


/*

PART_NO.VIEW
by
Norman Stevens

Report file created: PART_NUM.RPT, prints at MYPRINTER
Report is RBF report: PART_NUM.RPT
Procedure is: PART_NO_RPT.COM, called in frame QAREPORT.

*/

```
create view part_no_view (part_number, nomenclature, qa_orgn, qa_engineer,  
pa, wpn_pgm, description) as select u.unit_stem,  
c.com_nom_type + c.com_nom_num, p.asnd_orgn, p.last_name, n.doe_code,  
mi.wpn_pgm_code + '-' + mi.wpn_mod_num, u_d.unit_desc  
from unit u, component c, unit_desc u_d, person p, mfg_inst mi,  
nwc_agency n  
where u.unit_stem = c.unit_stem and u.unit_da_suf = c.unit_da_suf  
and u.unit_stem = u_d.unit_stem and u.unit_da_suf = u_d.unit_da_suf  
and u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf  
and u.da_qa_engr = p.id_num and u.prn_mfg_code = n.mfg_code;
```

PART_NO_RPT.COM Procedure
by Norman Stevens

This procedure is called from frame QAREPORT and
generates the part number report.
(PART_NUM_RPT is RBF with no source code)

```
$ define/proc ing_set_upremis "set lockmode session where readlock = nolock"
$ report -fpart_num.rpt -v61 upremis part_no_rpt
$ 0db1:[dbaprd.b.source]print_dispose part_num.rpt
$ exit
```



```

$ | PRINT_DISPOSE.COM
$ |
$ |
$ | SET VERIFY
$ |
$ | PROCEDURE TO DISPOSE PRINTOUT TO PRINTER DEFINED BY SYMBOL MYPRINTER
$ | P1 HAS FILE NAME
$ | THIS PROCEDURE IS CALLED BY REPORT CREATORS
$ | PROD.OSQ AND ALLID.OSQ IN THE PREMIS INGRES APPLICATION
$ | IF EACH COMMAND IS EXECUTED IN PREMIS WITH CALL SYSTEM,
$ | SCREEN WILL FLASH OFF FOR EACH CALL SYSTEM,
$ | AND UNDEFINED SYMBOL MYPRINTER WILL CAUSE AN
$ | INTERRUPT REQUIRING THE USER TO HIT RETURN TO CONTINUE
$ | SET UP BY JOHN ORMAN 9/21/88
$ |
$ | FOR T56LAB QUEUE, MYPRINTER = "SAV94::TXA3:"
$ | GET USER NAME FOR POSSIBLE LATER USE - MUST TRUNCATE TRAILING BLANKS
$ | USERNAME == F$EXTRACT(0,8,F$GETJPI("", "USERNAME"))
$ | LEN = 'F$LOCATE(" ", USERNAME)'
$ | USERNAME == F$EXTRACT(0,LEN, USERNAME)
$ | SEE IF MYPRINTER DEFINED - WILL GET UNDEFINED SYMBOL WARNING IF NOT DEFINED
$ | THIS IS SAME WAY MYPRINTER IS USED IN PTD MIRACLE SYSTEM
$ | OUT_DEV = ""
$ | OUT_DEV = MYPRINTER
$ | IF OUT_DEV .EQS. "" THEN GOTO NO_PRINTER_DEFINED
$ | PRINTER_DEFINED:
$ | MYPRINTER IS DEFINED, SEND FILE TO PRINTER WITH COPY,
$ | INSERT USER NAME INTO PRINTOUT NAME
$ | COPY/LOG 'P1' 'MYPRINTER' 'USERNAME' '_P1'
$ | GOTO FINISHED
$ |
$ | NO_PRINTER_DEFINED:
$ | MYPRINTER NOT DEFINED, SEND FILE TO T56LAB PRINTER
$ |
$ | PRINT/NAME=PREMIS/QUEUE=T56LAB/TRAILER=ONE/DELETE 'P1'
$ |
$ | FINISHED:
$ |
$ | EXIT

```

QA_ORGN.VIEW
by
Norman Stevens

Report file created: QA_ORGN.RPT, prints at MYPRINTER
Report is RBF report: QA_ORGN_RPT
Procedure is: QA_ORGN_RPT.COM, called in frame QAREPORT.

```
*/  
create view qa_orgn_view (qa_orgn, qa_engr_name, wpn_pgm, nomenclature,  
part_number, pa, description) as select p.asnd_orgn, p.last_name,  
mi.wpn_pgm_code + '-' + mi.wpn_mod_num, c.com_nom_type + c.com_nom_num,  
u.unit_stem, n.doe_code, u_d.unit_desc  
from unit u, component c, unit_desc u_d, person p, mfg_inst mi,  
nwc_agency n  
where u.unit_stem = c.unit_stem and u.unit_da_suf = c.unit_da_suf  
and u.unit_stem = u_d.unit_stem and u.unit_da_suf = u_d.unit_da_suf  
and u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf  
and u.da_qa_engr = p.id_num and u.prn_mfg_code = n.mfg_code;
```

QA_ORGN_RPT.COM Procedure
by Norman Stevens

This procedure is called from frame QAREPORT and
generates the QA Organization/Engineer report.
(QA_ORGN_RPT is RBF with no source code)

```
$ define/proc ing_set_upremis "set lockmode session where readlock = nolock"  
$ report -fqa_orgn.rpt -v61 upremis qa_orgn_rpt  
$ 0db1:[dbaprd.b.source]print_dispose qa_orgn.rpt  
$ exit
```

/*

WPN_PGM.VIEW
by
Norman Stevens

Report file created: WPN_PGM.RPT, prints at MYPRINTER
Report is RBF report: WPN_PGM_RPT
Procedure is: WPN_PGM_RPT.CDM, called in frame QAREPORT.

*/

```
create view wpn_pgm (wpn_pgm, nomenclature, part_number, qa_orgn,  
qa_engr_name, pa, description) as select  
mi.wpn_pgm_code + '-' + mi.wpn_mod_num, c.com_nom_type + c.com_nom_num,  
u.unit_stem, p.asnd_orgn, p.last_name, n.doe_code, u_d.unit_desc  
from unit u, component c, unit_desc u_d, person p, mfg_inst mi,  
nwc_agency n  
where u.unit_stem = c.unit_stem and u.unit_da_suf = c.unit_da_suf  
and u.unit_stem = u_d.unit_stem and u.unit_da_suf = u_d.unit_da_suf  
and u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf  
and u.da_qa_engr = p.id_num and u.prn_mfg_code = n.mfg_code;
```

```

$ !           WPN_PGM_RPT.COM Procedure
$ !           by Norman Stevens
$ !
$ !           This procedure is called from frame QAREPORT and
$ !           generates the weapon program/nomenclature report.
$ !           (WPN_PGM_RPT is RBF with no source code)
$ !
$ define/proc ing_set_upremis "set lockmode session where readlock = nolock"
$ report -fwpn_pgm.rpt -v61 upremis wpn_pgm_rpt
$ 0db1:[dbaprd.b.source]print_dispose wpn_pgm.rpt
$ exit

```

/*

WPN_PGM_QA.VIEW

by

Norman Stevens

Report file created: WPN_PGM_QA.RPT, prints at MYPRINTER

Report is RBF report: WPN_PGM_QA.R

Procedure is: WPN_PGM_QA_RPT.COM, called in frame QAREPORT.

*/

```
create view wpn_pgm_qa_v (wpn_pgm, qa_orgn, qa_engr_name,
  nomenclature, part_num, pa, description) as select
  mi.wpn_pgm_code + '-' + mi.wpn_mod_num, p.asnd_orgn, p.last_name,
  c.com_nom_type + c.com_nom_num, u.unit_stem, n.doe_code,
  u_d.unit_desc
  from unit u, component c, unit_desc u_d, person p, mfg_inst mi,
  nwc_agency n
  where u.unit_stem = c.unit_stem and u.unit_da_suf = c.unit_da_suf
    and u.unit_stem = u_d.unit_stem and u.unit_da_suf = u_d.unit_da_suf
    and u.unit_stem = mi.unit_stem and u.unit_da_suf = mi.unit_da_suf
    and u.da_qa_engr = p.id_num and u.prn_mfg_code = n.mfg_code;
```

```

$ |           WPN_PGM_QA_RPT.COM Procedure
$ |           by Norman Stevens
$ |
$ |   This procedure is called from frame QAREPORT and
$ |   generates the weapon program/QA organization report.
$ |   (WPN_PGM_QA.R is RBF with no source code)
$ |
$ define/proc ing_set_upremis "set lockmode session where readlock = nolock"
$ report -fwpn_pgm_qa.rpt -v61 upremis wpn_pgm_qa_r
$ 0db1:[dbaprdb.source]print_dispose wpn_pgm_qa.rpt
$ exit

```

Appendix I -- PREMIS Data Dictionary

PREMIS Data Dictionary

Variable Name	data type	size	tables	description
ADDRESS_ID	character	09	ADDRESS	The Identifier for the ADDRESS record. It will be the same as the MFG_CODE in the COMPANY record. The three character code is assigned by the control clerk in charge of the record copy for company codes.
ALT_MFG_CODE	character	02	COMPANY	The two-character identifier for companies used as record copy prior to the current three character codes.
ASND_ORGN	character	06	PERSON NOMEN_VIEW	The organization to which an employee is assigned.
CER_DT	date		DATA_DOCUMENT	The Date that the document first officially meets the design agency's intent and becomes part of the drawing set.
CITY_NAME	character	20	ADDRESS NWC_COM_VIEW	The name of the city in which the company is located.
COM_CTR_CONT	character	06	QA_ENGR_RELS	The employee number of the communication center employee who accepted the transmission of a quality assurance engineering release.
COM_NOM_NUM	character	08	COMPONENT	The last eight characters of the component nomenclature.
COM_NOM_TYPE	character	02	COMPONENT	The first two characters of the component nomenclature. Valid values include CA, CF, CT, H, MC, SA, SB, SS, TC, and blank.
COMP_REQ	character	09	COMPANY	The ID NUM of the person who requested that the company be added to the database.
COMPANY_NAM2	character	34	COMPANY	The second line of the name of the company.
COMPANY_NAME	character	34	COMPANY COMPANY_SYN	The name of the company, which could be an NWC Agency or a Supplier.
CONFIG	character	04	UNIT	The material type. Valid values include JT, TYPE, TRA, and WR.
COUNTRY_NAME	character	20	ADDRESS	The name of the country in which a company is located. This field is only used when the country is not the U. S. A.
CURRENT_CODE	character	03	COMPANY_HIST	The code now used as a replacement for the historical company.
DA_DATA_ENGR	character	06	UNIT	The employee number of the data engineer responsible for this unit.
DA_DESN_ENGR	character	06	UNIT	The employee number of the engineer who designed the component.
DA_N_SF_ENGR	character	06	UNIT	The employee number of the Sandia Engineer responsible for nuclear safety.
DA_PROD_ENGR	character	06	C_DA_PRD_EGR	The employee number of the Sandia Engineer who acts as a product engineer for components produced by Sandia.
DA_QA_ENGR	character	06	UNIT	The employee number of the quality assurance engineer assigned to this unit.
DA_REL_ENGR	character	06	UNIT	The employee number of the reliability engineer responsible for the reliability of this unit.

Variable Name	data type	size	tables	description
DESCRIPTION	character	30	NOMEN VIEW PART NO VIEW QA ORGN VIEW WPN PGM VIEW WPN_PGM_QA_V	The UNIT_DESC for a particular component used only in reports.
DOC_ISS_NUM	character	02	CHAR DEV RPT DATA_DOCUMENT DOC ROUGH DT DOC_RQ_ML_DT DOCUMENT ENG EVAL REL EQ_ACT_EER EQ_INST QA_ENGR RELS QER PROBLEMS QL_EVAL RELS TMS ACTIVITY TMS_INST UNIT UNIT_DOC	The issue code for the document. It is usually one character. It is also referred to as the revision.
DOC_NUM	character	08	CHAR DEV RPT DATA_DOCUMENT DOC ROUGH DT DOC_RQ_ML_DT DOCUMENT ENG EVAL REL EQ_ACT_EER EQ_INST QA_ENGR RELS QER PROBLEMS QL_EVAL RELS TMS ACTIVITY TMS_INST UNIT UNIT_DOC	The eight-character document number which in many cases corresponds to the UNIT_STEM.
DOC_TYPE	character	04	CHAR DEV RPT DATA_DOCUMENT DOC ROUGH DT DOC_RQ_ML_DT DOCUMENT ENG EVAL REL EQ_ACT_EER EQ_INST QA_ENGR RELS QER PROBLEMS QL_EVAL RELS TMS ACTIVITY TMS_INST UNIT UNIT_DOC	A four-character abbreviation indication the type of document. Valid values include AB, DF, DR, EER, PS, QER, RD, REVL, SAND, SB, SC, and SIER.

Variable Name	data type	size	tables	description
DOE_CODE	character	02	NWC_AGENCY NWC_COM_VIEW	The two-digit code assigned by the Department of Energy.
DSN_MFG_CODE	character	03	UNIT	The manufacturing code of the design agency responsible for the design of the unit.
DTDR_DT	date		DATA_DOCUMENT	Drawing Transfer Engineering Release Date. The date that the design agency releases the document to the production agency.
E_DESN_ENGR	character	06	WEAPON	The employee number of the lead electrical design engineer for a weapon.
EER_STATUS	character	04	EER_STATUS	The status of a quality assurance document.
EMPL_NUM	character	06	DA_CT_DS_EGR DA_DATA_ENGR DA_DESN_ENGR DA_ELEC_ENGR DA_EEMPL DA_ENGR DA_MECH_ENGR DA_N_SF_ENGR DA_PROD_ENGR DA_QA_ENGR DA_REL_ENGR	The employee number of the Sandia Employee.
ENTRY_DATE	date		COMPANY	The date that the company was added to the database or the date that the MFG_CODE of the company was last changed.
EQUIP_NUM	character	04	EQUIPMENT GAGE MFG_EQP_INST TESTER	The equipment number used to identify a piece of equipment.
EQUIP_TYPE	character	02	EQUIPMENT GAGE MFG_EQP_INST TESTER	The equipment type. Valid values include but are not limited to AE, GA, GL, MT, PT, SE, TE, and UA.
EQP_INST_NUM	character	04	EQ_ACT_EER EQ_ACTIVITY EQ_INST EQ_PQ_TEAM MFG_EQP_INST	A computer-generated number used to identify a specific manufacturer equipment instance.
EST_COMP_DT	date		DOCUMENT	The estimated completion date for a document.
EST_DRAFT_DT	date		CHAR_DEV_RPT	The estimated completion date for the final draft of a characteristic and development report.
EST_PUB_DT	date		CHAR_DEV_RPT	The estimated publication date for a characteristic and development report.

Variable Name	data type	size	tables	description
FILE_CAB_LOC	character	09	QA_ENGR_RELS FILE_CABINET	The location of the hardcopy of the quality assurance document.
FINL_DRFT_DT	date		CHAR_DEV_RPT	The date the final draft of the characteristic and development report is completed.
FIRST_NAME	character	12	PERSON	The first name of the person.
FRST_PROD_DT	date		MFG_INST	The date a particular manufacturing instance was first produced.
HRD_SFT_FLAG	character	01	EQ_ACT_EER EQ_ACTIVITY EQ_INST EQ_PQ_TEAM TMS_ACTIVITY TMS_INST TMS_PQ_TEAM	Tells whether the item being tested is hardware or software.
ID_NUM	character	09	DA_EMPL EQ_PQ_TEAM PA_DATA_ENGR PA_EMPL PA_PROD_ENGR PA_QC_ENGR PERSON TMS_PQ_TEAM	The identification number of employees who are stored in PREMIS, but are non-Sandians. In some cases, Sandia Employee Numbers (Ennnnn) will also appear in these fields.
ISSUE_DT	date		QA_ENGR_RELS	The date on which a quality assurance engineering release is issued.
LAST_NAME	character	15	PERSON	The last name of the person.
M_DESN_ENGR	character	06	WEAPON	The employee number of the lead mechanical design engineer for a weapon.
M11_DIR_NAME	character	02	QA_ENGR_RELS	The name of the mess-11 directory in which a quality assurance engineering release is located.
MAIL_STOP	character	06	PERSON	The mail stop of the person.
MEMBER_TITLE	character	16	EQ_PQ_TEAM TMS_PQ_TEAM	The title of the ad hoc member of the Product Qualification Team.
MESSAGE_NUM	character	05	QA_ENGR_RELS	The message number assigned when the document was transmitted to the COMM Center.
MFG_CODE	character	03	COMPANY COMPANY_SYN DESIGN_AGENCY MANUFACTURER MFG_INST NWC_AGENCY NWC_COM_VIEW PERSON PROD_AGENCY SUPPLIER	The three character manufacturer's code which uniquely identifies all NWC agencies and suppliers used by them.

Variable Name	data type	size	tables	description
PRD_MFG_CODE	character	03	UNIT	The manufacturing code of the production agency responsible for the production of the unit.
PROBLEM_DESC	character	30	QER_PROBLEMS	A description of one of the problems identified by a quality evaluation release.
PUBLISHED_DT	date		CHAR_DEV_RPT	The date that the characteristic and development report is published.
QA_ACTV_TYPE	character	04	EQ_ACTIVITY TMS_ACTIVITY	Tells what type of quality assurance work is being done. Valid values include CERT, EQ, INST, QS, REQ, RTMS, RTPV, TMS, and TPV.
QA_ORGN	character	06	NOMEN_VIEW PART NO VIEW QA ORGN_VIEW WPN_PGM_VIEW WPN_PGM_QA_V	The organization number of the division to which the QA Engineer for a particular component is assigned. This field is used only in reports.
QA_ENGINEER	character	08	NOMEN_VIEW PART NO VIEW QA ORGN_VIEW WPN_PGM_VIEW WPN_PGM_QA_V	The LAST NAME of the QA Engineer for a particular component. This field is used only in reports.
PA	character	02	NOMEN_VIEW PART NO VIEW QA ORGN_VIEW WPN_PGM_VIEW WPN_PGM_QA_V	The DOE_CODE of the production agency for a particular component. This field is used only in reports.
PROV_CODE	character	02	PROVINCE	The four-character abbreviation for the Canadian province in which the vendor is located.
R_DRAFT_DT	date		DOC_ROUGH_DT	The date that the rough draft of the document is complete. This is optional.
REQ_ML_DT	date		DOC_RQ_ML_DT	The date that the design agency asks the production agency to place the document on the material list.
REQUIRED_DT	date		DOCUMENT	The date the completion of the document is required.
RSLT_RPT_FL	character	01	ENG_EVAL_REL	A flag that indicates whether the results report has been filed or not.
SERIAL_NUM	character	02	EQUIPMENT GAGE MFG EQP_INST TESTER	The serial number used to uniquely identify a piece of equipment.
SOLVED_DT	date		QER_PROBLEMS	The date on which a problem identified by a quality evaluation release must be solved.
SOURCE	character	34	COMPANY	The record or person that served as the source for verification of the data for a new COMPANY record
STATE_CODE	character	02	ADDRESS STATE	The two character abbreviation for the state in which the agency is located.

Variable Name	data type	size	tables	description
MFG_INST_NUM	character	04	MFG_EQP_INST MFG_INST TMS_ACTIVITY TMS_INST TMS_PQ_TEAM	A computer-generated number used to identify a specific manufacturing instance.
MIDDLE_INIT	character	01	PERSON	The middle initial of the person.
ML_DT	date		DATA_DOCUMNT	The date the document is placed on the Material List.
NEW_MFG_CODE	character	03	COMPANY_HIST	The value that was stored in MFG_CODE in the COMPANY record when a change occurred for this historical company.
NOMENCLATURE	character	10	NOMEN_VIEW PART_NO_VIEW QA_ORGN_VIEW WPR_PGM_VIEW WPN_PGM_QA_V	The combination of COM_NOM_TYPE and COM_NOM_NUM used only in reports.
NOTES	character	34	COMPANY	Information stored to clarify the records in the COMPANY records.
OLD_MFG_CODE	character	03	COMPANY_HIST	The value which was stored in MFG_CODE in the COMPANY record before the last change of MFG_CODE.
OLD_NAME	character	34	COMPANY_HIST	The value which was stored in COMPANY_NAME in the COMPANY record before the last change in the COMPANY_NAME of the record.
PA_DATA_ENGR	character	09	C_PA_DTA_EGR	The ID number for the production agency employee who assists the design agency data engineer.
PA_PROD_ENGR	character	09	UNIT	The ID number of the production agency employee who is the product engineer for this unit.
PA_QC_ENGR	character	09	UNIT	The ID number of the production agency engineer who is responsible for the quality control for the unit.
PART_DA_SUF	character	03	GOZINTO	The Sandia suffix for a component which is a part of a larger unit.
PART_NUMBER	character	07	NOMEN_VIEW PART_NO_VIEW QA_ORGN_VIEW WPR_PGM_VIEW WPN_PGM_QA_V	The UNIT_STEM used only in reports.
PART_STEM	character	06	GOZINTO	The base part number for a component which is a part of a larger unit.
PERCENT_COMP	integer	02	CHAR_DEV_RPT	The percent complete of the characteristic and development report.
PHONE_NUM	character	10	COMPANY PERSON	The phone number of the person or company.

Variable Name	data type	size	tables	description
STATUS_CODE	character	01	QA_ENGR_RELS	The status of a quality assurance engineering release.
STRT_ADDR1	character	24	ADDRESS	The first line of the street address of the company.
STRT_ADDR2	character	24	ADDRESS	The second line of the street address of the company.
TRANS_COM_DT	date		QA_ENGR_RELS	The date on which a quality assurance engineering release is transmitted to the communication center.
TWX_RECEIP	numeric	01	QA_ENGR_RELS	The number of twx's received.
UNIT_DA_SUF	character	03	C DA PRD EGR C-PA-DTA-EGR COMPONENT GOZINTO MFG INST UNIT UNIT_DESC UNIT_DOC WEAPON	2 or 3 character suffix given to a component or weapon by Sandia to make identification unique.
UNIT_DESC	character	30	UNIT_DESC	A description of the unit.
UNIT_STEM	character	07	C DA PRD EGR C-PA-DTA-EGR COMPONENT GOZINTO MFG INST UNIT UNIT_DESC UNIT_DOC WEAPON	7-digit part number used to identify components and weapons
WPN_MI_NUM	character	10	WEAPON_PGM	The MI number of a weapon.
WPN_MOD_NUM	character	02	MFG INST QA_ENGR_RELS WEAPON WEAPON_NOM WEAPON_PGM	Weapon modification number.
WPN_PGM	character	08	NOMEN VIEW QA_ENGR_RELS WEAPON WEAPON_NOM WEAPON_PGM	The concatenation of the WPN_PGM_CODE and WPN_MOD_NUM used only in reports.
WPN_PGM_CODE	character	04	MFG INST WEAPON WEAPON_NOM WEAPON_PGM	Weapon Program Code such as W88
WPN_TEST_CFG	character	07	WEAPON WEAPON_NOM	The test configuration of the weapon.
ZIP_CODE	character	09	ADDRESS	The company's zip code.

Appendix II -- PREMIS MAIN MENU HELP

HELP for the MAIN MENU

HELP MENU KEYS:

..... EXPLANATION of HELP MENU Keys (see also the Keys(2) choice from the prior menu)	
PF1 (F1 on PC's) ("MENU key")	Go to the menu line or/and once there, see the remaining menu keys (i.e., scroll the line).
^ followed by letter	Means type CTRL and Letter Key at the same time.
Keypad , Keypad * on some Pc's	Note: VTERM on PC - NUMLOCK must be on. Used as page down or Nextpage
Keypad -	Note: VTERM on PC - NUMLOCK must be on! Used as page up or Previous page key
^ (F) Find	Look for a specific word or phrase (which you type after the colon).

PREMIS CHOICES:

Now that you are at the main menu for the PProduct Evaluation Management Information System (PREMIS), you have four choices:

DATA engineering activities	You can choose to use the screens which have been designed to allow the data engineers to update and retrieve information.
Quality Assurance activities	You can choose to use the screens which have been designed to allow quality assurance personnel to update and retrieve information concerning product qualification.
Company code activities	You can choose to use the screens to access the 3-digit manufacturer or agency codes.
HELP	You can choose this to read HELP screens.
STOP (exit PREMIS)	You can return to system level (\$) and use other applications or logoff.

In order to choose ...	enter ... or	enter ...
DATA engineering activities	1 on the numeric keypad	D or DATA and <CR>
Quality Assurance activities	2 on the numeric keypad	Q and <CR>
Company code activities	3 on the numeric keypad	C or COMPANY <CR>
HELP (PF2)	4 on the numeric keypad	H or HELP and <CR>
STOP (exit PREMIS)	5 on the numeric keypad	S or STOP and <CR>

GENERAL - MAKING MENU CHOICES:

General Terms	Definition or explanation
KEYPAD CHOICE	<p>This is the choice which uses the numeric keypad located on the far right of most keyboards. When exercising the keypad choice, do NOT hit <CR>. On all menu screens, the KEYPAD CHOICES will be located on the far left and they will be underlined for your convenience. On update and retrieval screens you will also have choices which will appear at the bottom of the screen. The keypad choices will be in parentheses.</p> <p>EXAMPLE: RETRIEVE(1)</p> <p>In the example RETRIEVE is the MENU CHOICE and 1 is the KEYPAD CHOICE. Do NOT enter the parentheses.</p>
MENU CHOICE	<p>In order to exercise the menu choice the cursor must be at the bottom of the screen. On MENU screens, it will already be there. On update or retrieval screens, if you want to exercise the MENU CHOICE, you must hit the F1 KEY or the PF1 key. In order to exercise a menu choice, you must enter as many of the letters in the MENU CHOICE as it takes to be unique. On menu screens, the minimum number of letters will be highlighted and underlined. On update and retrieval screens, you must make this determination yourself. It is necessary to hit the return key after using the MENU CHOICE.</p> <p>EXAMPLE: DATA(1) QA(2) HELP(PF2) STOP(PF4)</p> <p>If you enter D and <CR>, you have exercised the DATA menu choice correctly.</p>
STANDARD FUNCTION KEYS PF2, PF3, PF4	<p>Standard function keys (F or PF) always appear for HELP (PF2), XIT(PF3), and STOP (PF4).</p>
MENU KEY (STANDARD FUNCTION KEY 1: F1 OR PF1)	<p>The MENU KEY is either the PF1 key if you have a DEC terminal (It is located on the keypad on the right side of the keyboard.), or the F1 key if you have a typical PC keyboard. (It is located on the left side of the keyboard.)</p> <p>On all screens, a menu appears at the bottom of the screen. If you see the > symbol at the bottom, all the choices are being shown. If not, more choices will be displayed if you hit the menu key. The original choices will be redisplayed if you hit the menu key again.</p>

SCREEN TYPE	Explanation
MENU	Menu screens contain only choices of things to do

RETRIEVAL	Retrieval screens are used only to access information. They cannot be used to alter data.
UPDATE	Update screens can be used to add or modify data. It is always advisable to retrieve any data which may already be stored in PREMIS before adding or modifying data.

FIELDS (DATA ITEMS):

FIELD TYPE	Explanation
KEY	<p>A field is considered to be a key if it can be used to access other data. Key fields will be displayed in reverse video and highlighted. If you want to access data about a component, you can enter the part number or the nomenclature. Both are key fields. On some screens, you may be required to enter data in more than one key field.</p> <p>EXAMPLE: On the product qualification screen, you may be required to enter not only the nomenclature or subject, but also the weapon program and the supplier.</p>
Update	Fields which you can alter or update are in reverse video. You can add or modify the data in these fields.
Display only	<p>These fields are highlighted. You can not tab to these fields. You can only affect their contents indirectly.</p> <p>EXAMPLE: If you alter the design engineer, his phone number will also be altered.</p>
Table Field	<p>A field with vertical bars on it which shows more than one value in it by displaying more than one row.</p> <p>There can be many more values in a table field than those on the screen at the time. You can page through these data values by using two keys. The , (* on PCs) key on the keypad pages forward and the - key on the keypad pages backward.</p>
(GET DETAIL)	<p>A table field can be used like a menu of choices.</p> <p>Some screens have been designed so that you can move the cursor to a row about which you want more information and then press a menu key to make choices of actions for that row.</p>

CURSOR MOVEMENT:

There are several ways to move from one field to another on the screen.

These include:

- * the TAB key,
- * the RETURN key,
- * CTRL P,
- * the SPACE bar,
- * the DOWN arrow, and
- * the UP arrow.
- * the keypad , or keypad *
- * the keypad -

Use the ...	when you wish to ...	Example:
RETURN key	move forward one field and delete the remaining information on the screen.	The quality assurance engineer is incorrect and you wish to revise it. If you enter JONES and <CR>, any remaining characters from the previous entry will be deleted.
the SPACE bar	move forward one field and there is only one or two spaces left in the field.	You have entered the organization and only one space is left in the field. Once the field is full, cursor automatically moves to the next field.
TAB key (forward)	move forward one field and preserve the data in the current field.	The quality assurance engineer is correct and you wish to add/revise the design engineer.
(backward) Use CTRL P	move backward one field and preserve the data in the current field.	The cursor is on the field ahead of the one you wish to revise.
the DOWN arrow (F8 on VTERM)	move the cursor down one row in a table field.	You wish to enter an additional member of the AD HOC PQ team.
the UP arrow (F7 on VTERM)	move the cursor up one row in a table field.	You wish to choose the previous row in a table field.
the keypad , (* on PCs)	display the next screenful of data values in a table field.	You wish to move forward 1 screen in a table field.
the - key on the keypad.	display the previous screenful of data in a table field.	You wish to move backwards 1 screen in a table field.
CTRL G	Does a screen dump of current screen to file: PREMISSCREENDUMP.LIS This file is deleted at logon	You wish to print this screen. You must issue PRINT PREMISSCREENDUMP or LASERSCREEN for output before you logoff.

Appendix III -- Screen Definition Standards

When creating an application, it is necessary to have standards so that the application will be consistent in its design and implementation.

MENU SCREENS:

Menu screens are screens whose only function is to branch to other screens. Retrieval screens and update screens will have menu choices on the bottom line, but they do not fit the category menu screens.

Menu screens shall have the keypad choice on the left hand margin, and the menu choice on the right hand margin. Each keypad choice will have the attribute "underlined". The unique portion of each menu choice will have the attributes "underlined" and "brightness change". The rest of the menu choice will not have any attributes. A short description of the screen to which the program will branch if either the menu choice or the keypad choice is exercised will appear at the middle. The keypad choices for version 1 will assume that term_ingres is defined "VT100F". Other terminals may be supported in later versions. At the bottom of each menu screen will appear the following two lines to inform the user how to procede:

ENTER the KEYPAD CHOICE using the numeric keypad or
ENTER the MENU CHOICE and <CR>

The labels KEYPAD CHOICE and MENU CHOICE should be used to indicate what they are.

RETRIEVAL SCREENS:

Retrieval screens can only be used to retrieve data based on some key. The field which is used to input the key will not have the attribute "display only", but it will have the attribute "reverse video" as well as the attribute "brightness change". The data itself will be made distinguishable from the titles and trim by giving the screen fields the attribute "brightness change". This makes the data stand out. All screen fields except those used as keys should also be given the attribute "display only". This prevents the user from making unauthorized updates. More importantly, it saves the user from having to tab through these fields when moving the cursor.

Whenever a data item can occur more than once, a TABLE FIELD will be created. (For example a component can be a part of several other parts, so next assembly is a TABLE FIELD). TABLE FIELDS will be displayed without the lines between the occurrences to conserve space on the screen. If only one occurrence is expected, a regular FIELD should be used.

All retrieval screens will be equipped with an option to clear the screen and move the cursor to the first or preferred key field.

UPDATE SCREENS:

On many update screens, there are fields which can be updated, and fields which cannot. If a field can be updated, it will be given the attribute "reverse video". This will serve to inform the user that the field can be updated. It will also show the user the maximum length of the screen field. If the field will be used to compare against a data-base variable which is uppercase, the screen field should be given the attribute "force uppercase". If a field cannot be updated, it should be given the attributes "display only" and "brightness change". If a field is used to input the key value, it should be given the attributes "reverse video" and "brightness change".

If a field can be updated, every attempt should be made to edit the data in the field to ensure validity before the update is made. If the data is invalid, the user should be informed with an appropriate message. The user must be responsible for the correctness of the data, but the programmer should be responsible for validity and integrity. The program can tell that September 30 is a valid issue date, but it is up to the user to know the correct issue date. The programmer can ensure referential integrity, by ensuring that a document cannot be connected to a unit which is not in the UNIT table, etc. If the unit exists in the table, then it is up to the user to connect the correct unit to the correct document. The programmer can ensure entity integrity, by ensuring that the primary key is unique before inserting a new record in a table. This can be accomplished by doing a select on the table in question and exercising the inquire_ingres option.

Every attempt should be made to help the user locate the proper record. The user will not be required to know employee numbers. The program will search for them based on data supplied by the user. If the user knows the nomenclature and not the part number, the program should locate the part number based on the nomenclature.

Every update screen should be coded in such a way that a minimum number of key strokes will allow the user to clear the screen and start over. This will keep down the frustration level if the user gets caught making a lot of errors.

REFERENCES

1. Relational Technology, Inc., "INGRES/APPLICATIONS: Applications-by-Forms (SQL) User's Guide," Release 5.0 for VAX/VMS, INGRES Corporation, Alameda, CA, August, 1986.
2. Hofsted, K. and Skagestein, G. "Nijssen Information Analysis Method as a Basis for Construction of Information Systems," "Nord-Data 81" Conference, Copenhagen 1981, published in "Data," 718-1981.

END

DATE FILMED

01 / 08 / 91