April 2015
Describes how to work with Program Generator, Program Design Language, Source Modifications, CASE Programs, Additional Tools, and Source Code Inventory and Database.
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.
Contents

Preface ............................................................................................................................................................ xv

Audience ....................................................................................................................................................... xv

Documentation Accessibility ....................................................................................................................... xv

Related Information ........................................................................................................................................ xv

Conventions .................................................................................................................................................. xv

1 Overview to Computer Aided Software Engineering (CASE)

1.1 System Integration ............................................................................................................................. 1-1
1.1.1 Specifications ................................................................................................................................. 1-2
1.1.2 Fundamentals ................................................................................................................................. 1-2
1.1.3 History of the Program Generator ............................................................................................... 1-2
1.1.4 Evolution of the Program Generator ............................................................................................ 1-2
1.2 Features .............................................................................................................................................. 1-3
1.2.1 What are the Benefits of CASE? ................................................................................................. 1-3
1.3 Terms and Concepts ........................................................................................................................... 1-4
1.3.1 CASE, as an industry term .......................................................................................................... 1-4
1.3.2 CASE, as a JD Edwards World term ......................................................................................... 1-4
1.4 Detailed Information .......................................................................................................................... 1-4
1.4.1 CASE Profile ................................................................................................................................. 1-5
1.4.2 Computer Assisted Design (CAD) ............................................................................................. 1-5
1.4.3 Computer Assisted Programming (CAP) ................................................................................. 1-5
1.4.4 About The Program Generator ................................................................................................. 1-5
1.4.5 About Program Types ................................................................................................................. 1-6
1.4.6 About Master Source Code ....................................................................................................... 1-6
1.4.7 About Program Specifications .................................................................................................... 1-6
1.4.8 Program Types ............................................................................................................................. 1-6
1.5 Menu Overview ................................................................................................................................ 1-7

Part I Foundation

2 Foundation

2.1 About Foundation Information .......................................................................................................... 2-1
3 Work with Prerequisites JD Edwards World Provides

3.1 Working with Prerequisites JD Edwards World Provides ................................................... 3-1
3.1.1 Program Generator Files ................................................................................................ 3-1
3.1.2 Common UDCs ............................................................................................................. 3-2
3.1.3 Source Code for Copy Modules ..................................................................................... 3-5
3.1.4 Source Code for JD Edwards World Files ................................................................. 3-5

4 Work with User-Provided Prerequisites

4.1 Development Libraries ........................................................................................................ 4-1
4.2 Multi-member Source File (JDESRC) ..................................................................................... 4-1
4.3 Job Queues ........................................................................................................................ 4-3
4.4 Project Management ............................................................................................................ 4-3
4.5 CASE Profiles .................................................................................................................. 4-4
4.5.1 Function Exits ............................................................................................................... 4-8
4.6 Object Authorities ................................................................................................................. 4-8
4.6.1 Job Control Authority .................................................................................................. 4-8
4.6.2 Source Library ............................................................................................................... 4-8
4.6.3 Source File .................................................................................................................... 4-8
4.6.4 Job Queues .................................................................................................................... 4-8

Part II Program Generator

5 Overview to Program Generator

5.1 About Program Generator Steps ........................................................................................ 5-1

6 Access Program Generator Specifications

6.1 Accessing Program Generator Specifications ..................................................................... 6-1
6.2 Function Exits ...................................................................................................................... 6-3

7 Define Program Purpose and Type

7.1 Defining Program Purpose and Type .................................................................................. 7-1
7.2 Function Exits .................................................................................................................... 7-7

8 Work with File Specifications

8.1 What Are File Specifications? ............................................................................................... 8-1
8.2 Function Exits .................................................................................................................... 8-7

9 Define General Instructions

9.1 About Special Characters ................................................................................................... 9-1
9.2 Special Characters within Help Instructions ....................................................................... 9-2

10 Define Option and Function Exits

10.1 Defining Option and Function Exits ................................................................................ 10-1
10.2 Function Exit .................................................................................................................... 10-5
11 Work with the Detailed Programming Facility

11.1 About the Detailed Programming Facility ................................................................. 11-1
11.1.1 Available Options ................................................................................................. 11-3
11.1.2 Function Exits ....................................................................................................... 11-3
11.2 About Full Data Field Parameters .......................................................................... 11-4
11.2.1 Primary Uses of Full Data Field Parameters ........................................................ 11-4
11.2.2 Function Exits ....................................................................................................... 11-8
11.3 Loading VC0 Description Fields ............................................................................. 11-9
11.3.1 Example ................................................................................................................ 11-9
11.3.2 Example: User Defined Code ............................................................................... 11-9
11.4 Enabling the Database Update Function for Subfiles ............................................. 11-11
11.5 Creating *ENTRY PLIST Entries ........................................................................... 11-12
11.5.1 Example ................................................................................................................ 11-12
11.6 Protecting Fields from Being Cleared ..................................................................... 11-14
11.6.1 What You Should Know About ........................................................................... 11-15
11.7 Disabling Data Dictionary Edits ............................................................................. 11-16
11.8 Creating a Partial KLST for a File ............................................................................ 11-17
11.8.1 Example ................................................................................................................ 11-17

12 Define Processing Options

12.1 Overview .................................................................................................................... 12-1
12.1.1 What You Should Know About ........................................................................... 12-2
12.2 Example - Interactive Programs Using Processing Options .................................. 12-2
12.3 Example - Report Program Using Processing Options .......................................... 12-4
12.4 Defining Processing Options .................................................................................... 12-4
12.4.1 Function Exits ....................................................................................................... 12-8

Part III Program Design Language

13 Overview to Program Design Language

13.1 Objectives ................................................................................................................ 13-1
13.2 About PDL ............................................................................................................... 13-1

14 About PDL Statements and Syntax

14.1 About PDL Statements ........................................................................................... 14-1
14.1.1 Keywords .............................................................................................................. 14-2
14.1.2 Variables ............................................................................................................. 14-2
14.1.3 Database Files .................................................................................................... 14-2
14.1.4 Operators .......................................................................................................... 14-2
14.1.5 Constants .......................................................................................................... 14-3
14.1.6 Punctuation ....................................................................................................... 14-3
14.2 About Blocks of Statements ................................................................................... 14-3
14.2.1 Keywords and Syntax ......................................................................................... 14-3
18.2 Changing CAP Status ........................................................................................................ 18-2
18.3 Resolving CAP Status Invalid Error .............................................................................. 18-3

19 Work with Model Control Language Programs

19.1 Working with CL Models .............................................................................................. 19-1
19.2 JD Edwards World Model CL Programs ........................................................................ 19-2

Part V CASE Programs

20 Overview to CASE Programs

20.1 Objectives .................................................................................................................. 20-1
20.2 About CASE Programs ............................................................................................... 20-1

21 Overview to Subfile Inquiry Programs

21.1 Program Type Description .......................................................................................... 21-1
21.2 Display File Definition ............................................................................................... 21-1
21.3 CL Program Definition ............................................................................................... 21-1
21.4 File Specifications ...................................................................................................... 21-2
21.5 Detailed Programming Facility ................................................................................... 21-2
21.6 Special Considerations ............................................................................................... 21-2
21.7 Quick Start Generation ............................................................................................... 21-2

22 Overview to Subfile Maintenance Programs

22.1 Program Type Description .......................................................................................... 22-1
22.2 Display File Definition ............................................................................................... 22-1
22.3 CL Program Definition ............................................................................................... 22-2
22.4 File Specifications ...................................................................................................... 22-2
22.5 Detailed Programming Facility ................................................................................... 22-2
22.6 Special Considerations ............................................................................................... 22-2
22.7 Quick Start Generation ............................................................................................... 22-2

23 Create Report Programs

23.1 Understanding RDA Special Use Fields ..................................................................... 23-1
23.2 Creating a Total Format ............................................................................................. 23-3
23.3 Defining a Subheading ............................................................................................... 23-6
23.4 Understanding DREAM Writer Considerations ......................................................... 23-7

Part VI Additional Tools

24 Overview to Additional Tools

24.1 Objectives .................................................................................................................. 24-1
24.2 About Additional Tools ............................................................................................... 24-1
31  Work with Logic Modules
  31.1  Primary Logic Modules................................................................. 31-1
  31.2  Detail Logic Modules ................................................................. 31-2
  31.3  Generation Options ................................................................. 31-2
  31.3.1  Help Instructions Edit/Build.................................................... 31-2
  31.3.2  All Help Instructions.............................................................. 31-2
  31.3.3  Global Program Regeneration .............................................. 31-2
  31.4  Viewing the Logic Module Index ............................................. 31-3
  31.5  Viewing Logic Module Cross Reference ................................... 31-3
  31.6  Viewing Logic Module Op Codes ............................................. 31-4
  31.7  Maintaining the Logic Module File ......................................... 31-5
  31.7.1  Resequence Logic Module.................................................... 31-5
  31.7.2  Remove Logic Module.......................................................... 31-5
  31.8  Creating or Modifying Logic Modules ..................................... 31-6
  31.9  Creating or Modifying Formula Library Entry ....................... 31-7
  31.10 Copying or Moving Program Specifications............................ 31-8
  31.11 Printing Program Generator Specifications........................... 31-9
  31.12 Reviewing Source Modifications .......................................... 31-10
  31.13 Using Program Generator Updates ....................................... 31-11
  31.14 Using CASE Specifications Inquiry ...................................... 31-11

32  Understand Directives
  32.1  Functional Directives................................................................. 32-1
  32.2  Substitution Directives.............................................................. 32-7
  32.3  Exception Directives ................................................................. 32-10
  32.4  Conditional Directives.............................................................. 32-11

33  Work with the Question and Answer System
  33.1  About Simple Question & Answer........................................... 33-1
  33.2  Reviewing Questions in a Master Dialogue.............................. 33-1
  33.3  Adding New Q & A Dialogue................................................... 33-3
  33.4  Working with an Existing Dialogue ........................................ 33-6

34  Create User Defined PDL
  34.1  Creating User Defined PDL....................................................... 34-1

A  Program Generator Checklist
  A.1  Data File Design Aid ................................................................. A-1
  A.2  Screen Design Aid ................................................................. A-1
  A.3  Report Design Aid ................................................................. A-2
  A.4  Program Generator ................................................................. A-2

B  Programming Standards
  B.1  Error Handling ........................................................................ B-1
C  CASE Program Types

C.1  Guidelines ................................................................................................. C-2
C.2  A0010 - Interactive Subfile Inquiry ............................................................ C-2
C.2.1  Description ............................................................................................. C-2
C.2.2  Display File Definition ........................................................................... C-2
C.2.3  CL Program Definition ........................................................................... C-2
C.2.4  File Specifications .................................................................................. C-2
C.2.5  Detailed Programming Facility ............................................................... C-3
C.2.6  Special Considerations ........................................................................... C-3
C.2.7  Quick Start Generation ......................................................................... C-3
C.3  A0020 - Interactive Single Record Inquiry ................................................. C-3
C.3.1  Description ............................................................................................. C-3
C.3.2  Display File Definition ........................................................................... C-3
C.3.3  CL Program Definition ........................................................................... C-3
C.3.4  File Specifications .................................................................................. C-3
C.3.5  Special Considerations ........................................................................... C-4
C.3.6  Quick Start Generation ......................................................................... C-4
C.4  B0010 - Interactive Single Record Maintenance ......................................... C-4
C.4.1  Description ............................................................................................. C-4
C.4.2  Display File Definition ........................................................................... C-4
C.4.3  CL Program Definition ........................................................................... C-4
C.4.4  File Specifications .................................................................................. C-4
C.4.5  Detailed Programming Facility ............................................................... C-4
C.4.6  Quick Start Generation ......................................................................... C-4
C.5  C0010 - Batch Report with Totals ............................................................... C-4
C.5.1  Description ............................................................................................. C-5
C.5.2  Printer File Definition ............................................................................ C-5
C.5.3  CL Program Definition ........................................................................... C-5
C.5.4  File Specifications .................................................................................. C-5
C.5.5  Special Considerations ........................................................................... C-5
C.5.6  Quick Start Generation ......................................................................... C-5
C.6  C0020 - Batch Report with Totals and Subheadings .................................. C-5
C.6.1  Description ............................................................................................. C-5
C.6.2  Printer File Definition ............................................................................ C-6
C.6.3  CL Program Definition ........................................................................... C-6
C.6.4  File Specifications .................................................................................. C-6
C.6.5  Special Considerations ........................................................................... C-6
C.6.6  Quick Start Generation ......................................................................... C-6
C.7  C0025 - Batch Report with Totals and Subheadings .................................. C-6
C.7.1  Description ............................................................................................. C-6
C.7.2  Printer File Definition ............................................................................ C-6
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.13</td>
<td>D0060 - Interactive Subfile Maintenance with Action Code, without Options, by Key</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.1</td>
<td>Description</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.2</td>
<td>Display File Definition</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.3</td>
<td>CL Program Definition</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.4</td>
<td>File Specifications</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.5</td>
<td>Detailed Programming Facility</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.6</td>
<td>Special Considerations</td>
<td>C-14</td>
</tr>
<tr>
<td>C.13.7</td>
<td>Quick Start Generation</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14</td>
<td>D0070 - Interactive Subfile Maintenance with Action Code, with Options, by Relative Record Number</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14.1</td>
<td>Description</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14.2</td>
<td>Display File Definition</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14.3</td>
<td>CL Program Definition</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14.4</td>
<td>File Specifications</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14.5</td>
<td>Detailed Programming Facility</td>
<td>C-15</td>
</tr>
<tr>
<td>C.14.6</td>
<td>Special Considerations</td>
<td>C-16</td>
</tr>
<tr>
<td>C.14.7</td>
<td>Quick Start Generation</td>
<td>C-16</td>
</tr>
<tr>
<td>C.15</td>
<td>D0080 - Interactive Subfile Maintenance without Action Code, with Options, by Relative Record Number</td>
<td>C-16</td>
</tr>
<tr>
<td>C.15.1</td>
<td>Description</td>
<td>C-16</td>
</tr>
<tr>
<td>C.15.2</td>
<td>Display File Definition</td>
<td>C-16</td>
</tr>
<tr>
<td>C.15.3</td>
<td>CL Program Definition</td>
<td>C-16</td>
</tr>
<tr>
<td>C.15.4</td>
<td>File Specifications</td>
<td>C-17</td>
</tr>
<tr>
<td>C.15.5</td>
<td>Detailed Programming Facility</td>
<td>C-17</td>
</tr>
<tr>
<td>C.15.6</td>
<td>Special Considerations</td>
<td>C-17</td>
</tr>
<tr>
<td>C.15.7</td>
<td>Quick Start Generation</td>
<td>C-17</td>
</tr>
<tr>
<td>C.16</td>
<td>D0090 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number, Balance</td>
<td>C-17</td>
</tr>
<tr>
<td>C.16.1</td>
<td>Description</td>
<td>C-17</td>
</tr>
<tr>
<td>C.16.2</td>
<td>Display File Definition</td>
<td>C-18</td>
</tr>
<tr>
<td>C.16.3</td>
<td>CL Program Definition</td>
<td>C-18</td>
</tr>
<tr>
<td>C.16.4</td>
<td>File Specifications</td>
<td>C-18</td>
</tr>
<tr>
<td>C.16.5</td>
<td>Detailed Programming Facility</td>
<td>C-18</td>
</tr>
<tr>
<td>C.16.6</td>
<td>Special Considerations</td>
<td>C-18</td>
</tr>
<tr>
<td>C.16.7</td>
<td>Quick Start Generation</td>
<td>C-18</td>
</tr>
<tr>
<td>C.17</td>
<td>D0100 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Key</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.1</td>
<td>Description</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.2</td>
<td>Display File Definition</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.3</td>
<td>CL Program Definition</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.4</td>
<td>File Specifications</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.5</td>
<td>Detailed Programming Facility</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.6</td>
<td>Special Considerations</td>
<td>C-19</td>
</tr>
<tr>
<td>C.17.7</td>
<td>Quick Start Generation</td>
<td>C-19</td>
</tr>
<tr>
<td>C.18</td>
<td>E0010 - Interactive Window</td>
<td>C-20</td>
</tr>
<tr>
<td>C.18.1</td>
<td>Description</td>
<td>C-20</td>
</tr>
<tr>
<td>C.18.2</td>
<td>Display File Definition</td>
<td>C-20</td>
</tr>
<tr>
<td>C.18.3</td>
<td>CL Program Definition</td>
<td>C-20</td>
</tr>
</tbody>
</table>
C.18.4  File Specifications ................................................................. C-20
C.18.5  Define Option and Function Key Exits............................. C-20
C.18.6  Detailed Programming Facility ........................................... C-20
C.18.7  Special Considerations ....................................................... C-21
C.18.8  Quick Start Generation ....................................................... C-21
C.19  X0010 - Batch Update with Report ...................................... C-21
   C.19.1  Description ................................................................. C-21
   C.19.2  Printer File Definition .................................................. C-21
   C.19.3  CL Program Definition ............................................... C-21
   C.19.4  File Specifications ........................................................ C-21
   C.19.5  Special Considerations ................................................ C-21
   C.19.6  Quick Start Generation ................................................ C-22
C.20  X0020 - Batch Update ........................................................... C-22
   C.20.1  Description ................................................................. C-22
   C.20.2  Printer File Definition .................................................. C-22
   C.20.3  CL Program Definition ............................................... C-22
   C.20.4  File Specifications ........................................................ C-22
   C.20.5  Special Considerations ................................................ C-22
   C.20.6  Quick Start Generation ................................................ C-22
C.21  X0030 - Batch Update with Subroutine S001 ....................... C-23
   C.21.1  Description ................................................................. C-23
   C.21.2  Printer File Definition .................................................. C-23
   C.21.3  CL Program Definition ............................................... C-23
   C.21.4  File Specifications ........................................................ C-23
   C.21.5  Special Considerations ................................................ C-23
   C.21.6  Quick Start Generation ................................................ C-23
C.22  X0040 - Batch Update with Report ...................................... C-23
   C.22.1  Description ................................................................. C-23
   C.22.2  Printer File Definition .................................................. C-24
   C.22.3  CL Program Definition ............................................... C-24
   C.22.4  File Specifications ........................................................ C-24
   C.22.5  Special Considerations ................................................ C-24
   C.22.6  Quick Start Generation ................................................ C-24
C.23  Y0010 - Conversion, Two Files with Error Report ................ C-24
   C.23.1  Description ................................................................. C-24
   C.23.2  Printer File Definition .................................................. C-25
   C.23.3  CL Program Definition ............................................... C-25
   C.23.4  File Specifications ........................................................ C-25
   C.23.5  Special Considerations ................................................ C-25
   C.23.6  Quick Start Generation ................................................ C-25
C.24  Y0020 - Conversion, One File Update with Error Report .......... C-25
   C.24.1  Description ................................................................. C-25
   C.24.2  Printer File Definition .................................................. C-26
   C.24.3  CL Program Definition ............................................... C-26
   C.24.4  File Specifications ........................................................ C-26
   C.24.5  Special Considerations ................................................ C-26
   C.24.6  Quick Start Generation ................................................ C-26
C.25  Y0030 - Conversion, One File Write with Error Report...................................................... C-26
C.25.1 Description................................................................................................................... C-26
C.25.2 Printer File Definition.................................................................................................. C-26
C.25.3 CL Program Definition .............................................................................................. C-27
C.25.4 File Specifications ..................................................................................................... C-27
C.25.5 Special Considerations .............................................................................................. C-27
C.25.6 Quick Start Generation .............................................................................................. C-27

D  Source Listings
D.1  Program Status Data Structure - I00DSPROG ............................................................... D-1
D.2  Copy Module - Retrieve Soft Coding Data Structure - I00SC ......................................... D-2
D.3  Item Master Information - P928011 ................................................................................ D-9

E  JD Edwards World Subroutines and Flows
E.1  Subroutines .................................................................................................................. E-1
E.2  Flows ............................................................................................................................ E-2
E.2.1 Interactive Non-Subfile Program .................................................................................. E-3
E.2.2 Subfile Program with Options .................................................................................... E-4
E.2.3 Report Program without Subheadings ....................................................................... E-5
E.2.4 Report Program with Subheadings ............................................................................ E-6

F  Sample Code

G  Functional Servers
G.1  Example: Voucher Processing Functional Server ......................................................... G-2

Index
Welcome to the JD Edwards World Computer Aided Software Engineering Guide.

Audience
This guide is intended for implementers and end users of JD Edwards World Computer Aided Software Engineering.

Documentation Accessibility
For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support
Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Information
For additional information about JD Edwards World applications, features, content, and training, visit the JD Edwards World pages on the JD Edwards Resource Library located at:
http://learnjde.com

Conventions
The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>boldface</td>
<td>Indicates cautionary information or terms defined in the glossary.</td>
</tr>
<tr>
<td>italic</td>
<td>Indicates book titles or emphasis.</td>
</tr>
</tbody>
</table>
Overview to Computer Aided Software Engineering (CASE)

This chapter contains these topics:
- Section 1.1, "System Integration,"
- Section 1.2, "Features,"
- Section 1.3, "Terms and Concepts,"
- Section 1.4, "Detailed Information,"
- Section 1.5, "Menu Overview."

1.1 System Integration

Computer Aided Software Engineering (CASE) covers the entire application development life cycle, including:
- Design tools
- Code generation
- Automatic documentation generation
- Prototyping
- Repositories
- Other productivity improvement tools

You use these tools to develop, operate, and maintain flexible, business application software.

Application Development Cycle

There are three technical levels in the JD Edwards World Application Development Cycle (A/D Cycle):

<table>
<thead>
<tr>
<th>Level</th>
<th>A/D Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 1</td>
<td>The Application Platform, which represents the Technical Foundation Guide.</td>
</tr>
<tr>
<td>Level 2</td>
<td>The Design Platform, which represents the Advanced Programming Concepts and Skills (APCS) Guide.</td>
</tr>
</tbody>
</table>
1.1.1 Specifications

You define a program using various Program Generator specifications. You perform the following:

- Define program purpose and type
- Specify the files
- Create help text
- Define function exits and options
- Add field-specific logic

You can also create processing options and document Automatic Accounting Instructions (AAIs). After you define the specifications, the Program Generator creates the program, adding the correct validation files and servers to complete the program.

1.1.2 Fundamentals

There are basic building blocks for a program. Program types are basic definitions of the programs. Using the Question and Answer facility, the system determines, based upon your answers, which program type to select. The program generator builds the program using primary and detail logic modules. You can add AAIs to your programs and create Control Language (CL) programs to launch programs from menu options.

1.1.3 History of the Program Generator

- Development started in 1984
- First called Clone
- First program generation was in April, 1985
- Rewrite of all systems (World Systems) done through Clone I & II
- Became known as the KBG (Knowledge Based Generator) in 1991
- Became known as the Program Generator in 1992

1.1.4 Evolution of the Program Generator

**Clone II Programs**

- Dynamic Data Dictionary
- Dynamic totaling and page skipping - created 39 lines of code per field which caused large S002 subroutines

**Clone II.5 Programs**

- Started in 1989
- Dynamic totaling and page skipping - creates 80 lines of code for ALL fields
- No more "?" code generated
- Cursor Sensitive Help (F1)
Features

- F24 Window
- Code for subfile option processing generated

**Current Program Generator Programs**
- More utilization of file servers
- Use of the Program Design Language (PDL) allow you to make modifications instead of making changes through Source Entry Utility

1.2 Features

JD Edwards World provides several tools to help create and customize your programs.
- PDL is available to add field-specific logic to your programs.
- Quick Start asks a few basic questions, and then creates a basic Report Program Generator (RPG) or CL program.
- JD Edwards World provides many different utilities to assist you in creating and maintaining your code.
- PDL enables you to add calculations or comparisons to specific fields within the program.
- Precompiler commands are available to specialize your compile environment

CASE includes the following features:
- Foundation
- Program Generator
- Program Design Language
- Source Modifications
- CASE Programs
- Additional Tools
- Source Inventory and Database

---

**Note:** You cannot use the Program Generator to modify existing JD Edwards World programs.

---

1.2.1 What are the Benefits of CASE?

Every program you create using the Program Generator automatically includes and uses JD Edwards World functionality, such as:
- Data Dictionary
- User defined codes
- Vocabulary overrides
- Action code security
- Business unit security
- Standard function exits
- Function exit and option exit security
■ Cursor sensitive help
■ Program help
■ DREAM Writer
■ Processing options

This functionality is consistent across all applications you generate because it is built into the Program Generator and Master Source.

The Program Generator is the same tool that JD Edwards World uses to generate the JD Edwards World application programs.

You can create simple programs in a short period of time using the Program Generator. Due to the standard structure and subroutines of the programs you generate, it is easier to incorporate complexities in either the Program Specifications or the source code.

You generate the source RPG code from Program Specifications, Program Types, and Master Source Code; therefore, you can regenerate the source as JD Edwards World enhances the functionality of its software. Because the enhanced functionality is in the Master Source Code File, you need to regenerate only the source code using the original Program Specifications.

---

**Note:** The JD Edwards Program Generator output is in RPGIII code. When you complete the program development cycle, you can use RPGIII to RPGIV converters from IBM and third party vendors. As of A7.3.14, A8.1.5 and A9.1, you can use the Software Versions Repository to manage RPGIV programs with the Function Code set to RPGL.

---

1.3 Terms and Concepts

1.3.1 CASE, as an industry term

As an industry term, CASE is an acronym for Computer-Aided Software Engineering. Many suppliers offer tools that implement various aspects of software engineering. These tools are either upper CASE or lower CASE tools.

Upper CASE tools focus on the business process and data models. Products that provide upper CASE capabilities include tools for organizational charts, decomposition diagrams, entity relationship diagrams, and data flow diagrams.

Lower CASE tools, on the other hand, focus on data models and generating source code. An example of a lower CASE product is J.D. Edward World CASE.

1.3.2 CASE, as a JD Edwards World term

As a JD Edwards World term, CASE refers to a set of tools that you use in the software development process. Following are the components of the CASE tools.

1.4 Detailed Information
1.4.1 CASE Profile

See CASE Profiles *JD Edwards World Advanced Programming Concepts and Skills Guide* for information on setting up the CASE profile. There are several methods to access the CASE profile:

- Choose CASE profile from the Computer Aided Design menu (G92)
- Choose Software Versions Repository from the Computer Aided Design menu (G92), and then choose Repository Services (F6) to access the Repository Services window.

You enter the CASE program source generation file and compiler options on this screen.

1.4.2 Computer Assisted Design (CAD)

CAD includes the following:

- Data Dictionary
- User Defined Codes
- File Design Aid (FDA)
- Screen Design Aid (SDA)
- Report Design Aid (RDA)

1.4.3 Computer Assisted Programming (CAP)

CAP includes the following:

- Program Generator
  - Program Purpose and Type
  - File Specifications
  - General Instructions (Help)
  - Option and Function Key Exits
  - Detailed Programming Facility
  - Processing Options
- CL Generator
  - Model CL programs (J98MODEL1 through J98MODEL8)
  - Quick Start CL Generator
- Quick Start Application Tool

1.4.4 About The Program Generator

The Program Generator is the JD Edwards World tool that generates source code for both RPG programs and CL programs. In many respects the Program Generator is a very simple tool that combines three functions and produces the source code as a result of combining the functions. The three functions are:

- Program Types
- Master Source Code
- Program Specifications
1.4.5 About Program Types

The Program Generator builds software depending on the program type you choose. The program types combine the features of:

- Interactive, for example, screen
- Batch, for example report or conversion
- Single record or multi-record

These program types contain a list of individual definitions which you combine to form a functional program. JD Edwards World refers to individual definitions as primary logic modules and the system uses them to build the source code for the program type. The system stores each primary logic module in the Master Source Code File. These logic modules are the components of all JD Edwards World program types.

1.4.6 About Master Source Code

The Master Source Code File consists of over 11,000 lines of RPG source code. Some lines are pure RPG source code. Others contain some RPG code and some JD Edwards World directives, which the Program Generator interprets and replaces with RPG code. The interpretation of the directives is based on the Program Specifications that you establish for generating a specific program type.

1.4.7 About Program Specifications

To generate a program, you must first complete the Program Generator Specifications. These specifications are the details of your program that the Program Generator uses to complete the RPG code it builds from the master source directives. There are six specifications. You must determine:

- A program type
- The files the program uses

After you specify this information, you can generate complete source code that you can then compile and execute.

1.4.8 Program Types

There are five categories of program types:

**Interactive**

- Can be either update or inquiry
- Can contain Action Codes
- Can contain a subfile
- Can interface with DREAM Writer to provide run-time options

**Window**

- Normally includes cursor sensitive helps (F1)
- Fits inside current interactive program

**Report**

- Provides for accumulated values (totals)
■ Interfaces with DREAM Writer
■ Can contain sub-headings

Server
■ Name includes a prefix of X
■ Updates master files
■ Can contain a report

Conversion
■ Use this program type to convert data from one file to another
■ Can contain a report

JD Edwards World currently provides 25 pre-defined program types with the Program Generator. The CASE Guide includes information to modify existing program types and to create of your own program types.

1.5 Menu Overview

JD Edwards World systems are menu driven. System functions are organized according to their function and frequency of use. The options on these screens illustrate the flow to the functions in this guide.

Figure 1–1 Master Directory screen
Figure 1–2  Advanced & Technical Operations screen

Figure 1–3  Computer Assisted Programming (CAP) screen
Figure 1–4  Model Program Design screen
This part contains these chapters:

- Chapter 2, "Foundation,"
- Chapter 3, "Work with Prerequisites JD Edwards World Provides,"
- Chapter 4, "Work with User-Provided Prerequisites."
This chapter contains the topic:
- Section 2.1, "About Foundation Information."

2.1 About Foundation Information

Before the Program Generator can successfully generate source code, a number of foundation items need to be in place. JD Edwards World provides some of these and you must verify they exist. You must perform additional prerequisites.

JD Edwards World provides the following prerequisites:
- Program Generator Files
- Common User Defined Codes
- Source Code for Copy Modules
- Source Code for JD Edwards World Files

You provide the following prerequisites:
- Development Libraries
- Multi-member Source File copied from F93002 (8 fields, 142 char record)
- Job Queues
- Project Management
- CASE Profiles
- Object Authorities
This chapter contains the topic:
- Section 3.1, "Working with Prerequisites JD Edwards World Provides."

### 3.1 Working with Prerequisites JD Edwards World Provides

The following are prerequisites JD Edwards World provides. You must verify their existence.

- Program Generator Files
- Common User Defined Codes (UDCs)
- Source Code for Copy Modules
- Source Code for JD Edwards World Files

#### 3.1.1 Program Generator Files

The Program Generator files follow. Each has a specific function when the system generates a program. Some of these files include data; while others have no data. You need to verify that the files exist in your CASE environment and that they appropriately contain data or not.

**Program Generator**

The following two files are database files and include data.

- Program Types (F93000)
- Master Source (F93001)

**Source Modifications/Helps**

The Help/Modification Master (F93002) file is a multi-member source file, and includes an empty F93002 member.

**Program Generator Specifications**

The following files are database files and do not include data.

- Program Purpose and Type (F93101)
- File Specifications (F93102)
- File Formats (F93103)
■ Selection/Function Exits (F93104)
■ Detail Field Definitions (F93105)
■ Automatic Accounting Instructions (F93106)

The DREAM Writer Master Parameter (F98301) file is a database file and includes processing options.

Program Design Language (PDL)
The following are database files. The Generation Operation Codes (F93108) file includes data; the other two files do not include data.
■ Generation Operation Codes (F93108)
■ Data Item Formulas (F93109)
■ Calculation Parameters (F93110)

Q&A Dialogue
The following files are database files and include data.
■ Dialogue Master (F00501)
■ Dialogue Detail (F00502)
■ Dialogue Questions (F00510)
■ Dialogue Responses (F00511)
■ Dialogue Text (F00512)

3.1.2 Common UDCs
The Program Generator uses the following four UDCs:
■ Logic Modules, 93/LM. Identifies the pieces of code within the Master Source Inventory file (F93001) that the system uses to create your RPG program.
Common Subroutine Copy Members, 93//C. Lists all of the copy modules on the system. Description-2 field contains any additional copy modules that are necessary to make the common subroutine function properly. For example, C0012 requires copy module E0012.
<table>
<thead>
<tr>
<th>Alphanumeric Code</th>
<th>Type of Copy Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Copy the member into the F specifications</td>
</tr>
<tr>
<td>E</td>
<td>Copy the member into the E specifications</td>
</tr>
<tr>
<td>I</td>
<td>Copy the member into the I specifications</td>
</tr>
<tr>
<td>C</td>
<td>Copy the member into the C specifications</td>
</tr>
</tbody>
</table>

- Servers, 93//X. This is a partial list of server programs and the associated copy member for each.

*Figure 3–3  General User Defined Codes (93//X) screen*

- Program Types, 93/PT. These are all the program types within the CASE tool.
3.1.3 Source Code for Copy Modules

The program generator requires that the source code for the Common Subroutine Copy Members be in the CASE environment. Copy modules are in file JDECPY in library JDFSRC.

3.1.4 Source Code for JD Edwards World Files

Source code for JD Edwards World database files must also be in the CASE environment. File source is in file JDESRC in library JDFSRC.
This chapter contains these topics:

- Section 4.1, "Development Libraries,"
- Section 4.2, "Multi-member Source File (JDESRC),"
- Section 4.3, "Job Queues,"
- Section 4.4, "Project Management,"
- Section 4.5, "CASE Profiles,"
- Section 4.6, "Object Authorities."

There are several prerequisites that the user must provide. These prerequisites include the program developer being signed on to a JD Edwards World environment including QGPL in the library list and additional prerequisites.

### 4.1 Development Libraries

You must provide three types of libraries for CASE:

- Source, which contains the Development Source File.
- Object, which contains the CASE generated programs and device files, as well as your non-CASE developed programs and device files.
- Data, which contains any CASE generated database files.

While it might be customary to create three different libraries for these purposes, this task is optional. Either of the following scenarios is acceptable.

<table>
<thead>
<tr>
<th>Unique Libraries</th>
<th>Common Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source = DEVSRC</td>
<td>Source = DEVLIB</td>
</tr>
<tr>
<td>Object = DEVOBJ</td>
<td>Object = DEVLIB</td>
</tr>
<tr>
<td>Data = DEVDTA</td>
<td>Data = DEVLIB</td>
</tr>
</tbody>
</table>

### 4.2 Multi-member Source File (JDESRC)

To use the Program Generator to develop application software, the program source file record must:

- Be 142 bytes long to allow for the Program Generator serial number.
- Contain eight specific fields. For example, DSPFFD F93002.
To create the multi-member source file JDESRC

1. Enter the following on the Command Line:

   CPYF FROMFILE(F93002) TOFILE(LIBRARY/XXXXXX) MBROPT(*NONE) CRTFILE(*YES)

   You use the F93002 file because it is in the correct format to generate the program.

   The To file can be any name and you can use JDESRC. However, JD Edwards World pristine source resides in JDESRC, so you can only use the JDESRC name if it is in a different library than the pristine JD Edwards World source library (JDFSRC).

   Do not use CRTSRCPF as it has only three fields in it, Date, Time, and Data, and the Program Generator requires extra fields.

2. Enter the following on the Command Line:

   RMVM FILE(LIBRARY/JDESRC) MBR(F93002)

After you create the JDESRC file, you can remove the empty member that the system adds.

Figure 4–1  Copy File (CPYF) screen
After you create the JDESRC file, you can remove the empty member that the system adds.

If the system incorrectly creates your JDESRC file, you receive the CPD3105 error message for incorrect source file format, due to either:

- The wrong length
- Improper formatting

### 4.3 Job Queues

By default, the system submits jobs to generate the program to the CLONE job queue, and the jobs to compile the program to the COMPILE job queue. If you want to use these default job queues, then you must create them and attach them to an existing subsystem.

If you want to use different job queues, or existing job queues, then you must override the defaults, either in the CASE Profile for *PUBLIC or the CASE Profile for specific users.

### 4.4 Project Management

You must determine the following regarding Project Management:

1. Whether to manage CASE generated programs (or any development work) using the JD Edwards World Software Action Request System (SAR). JD Edwards World includes the SAR system as part of System 00, General Back Office, under the name of Work Order Processing.

- If you are going to use the JD Edwards World SAR System, you create a SAR before starting development or use the number of an existing SAR for development.

- If you are not going to use the JD Edwards World SAR system, you can disable the function that allows the system to validate the SAR number by entering *NONE* in the SAR Number field on the CASE Profiles screen.

2. If you use the JD Edwards World SAR System to manage software development, then you must determine whether to use SAR logging.

SAR Logging is a process that allows you to associate a SAR number with all of the components of the software development project. These include the Data Dictionary, UDCs, Files, Programs, Vocabulary Overrides, DREAM Writer, and Menus. SAR Logging allows you to identify what components you must move from your development environment to a testing environment and a production environment.

If you use SAR Logging, you must determine what method you use to link a SAR number with each piece of the development work. There are two ways to associate a SAR number with development:

- Use a default SAR number, which you use with all development work until you change the default number.

- Enter the SAR number as you perform the development work.

The results of your decisions reside in your *PUBLIC CASE Profiles.

---

4.5 CASE Profiles

CASE profiles are user-defined values that can pertain to individual users or to one default *PUBLIC user profile. The system:

- Stores information in the CASE Profiles file (F98009).

- Uses the profiles to define the overall CASE operating environment.

You define various processing control parameters, including:

- Default development libraries

- Compile job queue

- Program Generator source generation job queue

- Compile print options

- SAR logging options

The system overrides the *PUBLIC default values with the individual CASE profile values.

- You must complete all fields when entering information for *PUBLIC.

- You maintain default CASE Profile values in a record with the User ID *PUBLIC. Enter CASE Profile values for individual users only if you want to override the *PUBLIC values.

- You can leave all fields blank except for the specific values you want to override when entering values for individual users.
The system uses the values in the record for User ID *PUBLIC as the defaults for all users unless individual user profiles have been set up.

You cannot delete the *PUBLIC record.

The system uses the values in the SAR Number and SAR Delivery fields to determine what type of SAR logging should occur.

<table>
<thead>
<tr>
<th>Value in the SAR Delivery field</th>
<th>SAR Logging</th>
</tr>
</thead>
<tbody>
<tr>
<td>*NONE</td>
<td>No SAR logging at all.</td>
</tr>
<tr>
<td>*LOG</td>
<td>The system does not include a SAR number as part of the SAR logging.</td>
</tr>
<tr>
<td>*DFT</td>
<td>The system uses the SAR number you specify for the SAR logging.</td>
</tr>
<tr>
<td>*PROMPT</td>
<td>The system prompts you for a SAR number and revision notes when it creates an entry to the SAR log.</td>
</tr>
</tbody>
</table>

To access CASE profiles

**Navigation**

From Computer Assisted Design (G92), choose Case Profiles

Alternatively, on the Software Versions Repository screen, choose Repository Services. On the Repository Services window, choose Case Profiles and then click Enter.

*Figure 4–3 Software Versions Repository screen*

The program attempts to locate the CASE profile for your User ID. An error message displays on the CASE Profiles screen if your user ID is not set up.

1. On Case Profiles, locate the *PUBLIC user ID.
2. Enter your Used ID, complete any of the fields and click Change to create your record.
Field | Explanation
--- | ---
Source File | The Source File Name field contains the name of the file where the source for an object exists. In the program generator File Specifications this name is defaulted to "JDESRC". In combination with the source library name it identifies where the program generator places the generated source code. The system uses this file to find the source for each data file, display file or report file which it must analyze to create the data field parameters. As used in the automated installation processing file this is the source file of an object at the time the object was created.

Source Library | The default library where source will be stored. The source file specified must reside within this library.

Object Library | The default library where compiled objects will be stored.

CL Source File | The default library where source for CL programs will be stored. This file must reside within the specified source library.

Data File Library | The default data file library specifies the test (or development) library for physical and logical files. This library is used as the default object library for the Software Versions Repository when copying source code for physical or logical files.

SAR Number | An abbreviation for software action request (SAR).
- *NONE = the SAR number will not be validated in any of the CAD/CAP programs and can be left blank.
- If a SAR number is entered, it is used in conjunction with the SAR Delivery Type of *DFT (default).
- If the SAR Number is left blank, you must enter a valid SAR number when using the CAD/CAP tools.

Version ID | The software version number to be defaulted in the Software Versions Repository file.
<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Explanation</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Code</td>
<td>This code determines the status of the software as well as where it resides in production. It will specify that the software is in production, in development, or in release.</td>
</tr>
<tr>
<td>Compile Job Queue</td>
<td>Specifies which job queue will be used when submitting programs to compile. This job queue is used for programs with function code of RPG, CBL, PLI, C and SYSC.</td>
</tr>
<tr>
<td>Prog Gen Job Queue</td>
<td>Specifies which job queue will be used when submitting jobs to the Program Generator. These jobs include the source code generation and the source code monitor from SEU.</td>
</tr>
<tr>
<td>Compile Target Release</td>
<td>Used by various System CRT commands (RPG, CLP, COBOL, C) to compile an object compatible with a specified target release.</td>
</tr>
<tr>
<td></td>
<td>■ A value of *CURRENT compiles an object compatible to the release of the machine at compile time.</td>
</tr>
<tr>
<td></td>
<td>■ A value of *PRV compiles an object compatible with both one release back and the current release.</td>
</tr>
<tr>
<td>Print Option</td>
<td>Used to designate whether or not a report will be generated when an object is compiled.</td>
</tr>
<tr>
<td></td>
<td>■ 0 = no print</td>
</tr>
<tr>
<td></td>
<td>■ 1 = print</td>
</tr>
<tr>
<td></td>
<td>■ 2 = print and hold spool file</td>
</tr>
<tr>
<td></td>
<td>■ 3 = print only, does not generate execution object (applies to COBOL and RPG only)</td>
</tr>
<tr>
<td></td>
<td>■ 4 = print when compile or creation fails</td>
</tr>
<tr>
<td>Cross-Reference Listing</td>
<td>Enter Y or N. Specifies whether a cross-reference listing will be generated for variables and fields in a program’s compile listing.</td>
</tr>
<tr>
<td>SAR File Library</td>
<td>Specifies which library the Software Action Request (SAR) file (F4801) being used for software development exists in. If left blank, the user’s library list will be used. You may specify *NONE in the SAR number field (MSAR) if you do not want any SAR number editing.</td>
</tr>
<tr>
<td>SAR Delivery Type</td>
<td>Associated with SAR logging, which tracks all modifications to JD Edwards World software. For example, it will track when User Defined Codes are modified.</td>
</tr>
<tr>
<td></td>
<td>■ *NONE = no logging.</td>
</tr>
<tr>
<td></td>
<td>■ *LOG = log to SAR number 00000000 (no SAR number is used for logging).</td>
</tr>
<tr>
<td></td>
<td>■ *DFT = log to a default SAR number (specified in the SAR Number field).</td>
</tr>
<tr>
<td></td>
<td>■ *PROMPT = log and prompt the user for the SAR number to be used and allow the user to enter the revision notes.</td>
</tr>
<tr>
<td>Source Gen Opt (Future)</td>
<td>Specifies whether to generate source interactively or in batch for programs with this option (for example, Fast Path Application Tool).</td>
</tr>
<tr>
<td></td>
<td>Allowed values are as follows:</td>
</tr>
<tr>
<td></td>
<td>1. generate source on-line (interactively)</td>
</tr>
<tr>
<td></td>
<td>2. generate source in batch</td>
</tr>
<tr>
<td>Helps Maint Opt (Future)</td>
<td>Enter a user defined code, 92/HL.</td>
</tr>
</tbody>
</table>
4.5.1 Function Exits

Choose Redisplay Previously Changed Member (F9) to locate the last record to which you made changes.

4.6 Object Authorities

The system checks the user’s authorities to some objects at different steps in generating programs using CASE. Therefore, it is necessary that you review these authorities initially.

4.6.1 Job Control Authority

On the user’s IBM User Profile, you must set the Special Authority parameter to *JOBCTL. This authority is necessary when entering the CASE Specifications.

4.6.2 Source Library

Ensure that the user has Object Management authority to the Source Library for software development.

4.6.3 Source File

Ensure that the user has Object Management authority to the Source File for software development.

4.6.4 Job Queues

Ensure that the user has authorization to use the job queues for generating source code and compiling programs.
This part contains these chapters:

- Chapter 5, "Overview to Program Generator,"
- Chapter 6, "Access Program Generator Specifications,"
- Chapter 7, "Define Program Purpose and Type,"
- Chapter 8, "Work with File Specifications,"
- Chapter 9, "Define General Instructions,"
- Chapter 10, "Define Option and Function Exits,"
- Chapter 11, "Work with the Detailed Programming Facility,"
- Chapter 12, "Define Processing Options."
This chapter contains the topic:

- Section 5.1, "About Program Generator Steps."

5.1 About Program Generator Steps

You perform the following tasks to create a program using the Program Generator:

- Define Program Generator Specifications
- Define Program Purpose and Type
- Work with File Specifications
- Define General Instructions
- Define Option and Function Key Exits
- Work with the Detailed Programming Facility
- Define Processing Options
This chapter contains these topics:
- Section 6.1, "Accessing Program Generator Specifications,"
- Section 6.2, "Function Exits."

### 6.1 Accessing Program Generator Specifications

You use the specification screens in the Program Generator to create a program.
- You must enter two specifications:
  - Program Purpose and Type
  - File Specifications

The system allows a third specification, Detailed Programming Facility, which it creates after you enter the File Specifications.
- Optional specifications include:
  - General Instructions
  - Option and Function Exits
  - Processing Options
  - Automatic Accounting Instructions

The system only requires source for files and common copy modules during the specifications and generation steps. The system does not require objects you define externally until you compile the program.

This section contains the following tasks:
- To access the Program Generator
- To access Program Generator Options

#### To access the Program Generator

The Software Versions Repository screen serves as the portal screen to all JD Edwards World Design tools including the Program Generator.

#### Navigation

From Computer Assisted Design (G92), choose Software Versions Repository

1. Locate a member from the Software Versions Repository. For example, locate P92801.
2. Enter 10 (Design) next to the environment in the following field:
   - Option
     The Program Generator Specification screen displays.

   To access Program Generator Options
   Enter 1 in the following field for the appropriate option on the Define Generator Specifications screen.
   - Option
Function Exits

Figure 6–3 Define Generator Specification (Options) screen

<table>
<thead>
<tr>
<th>Specification</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Purpose and Type</td>
<td>Defines what kind of program you are designing and the status of the program generation (CAP status).</td>
</tr>
<tr>
<td>File Specifications</td>
<td>Allows the user to enter the database files to be used by the program you are designing.</td>
</tr>
<tr>
<td>Define General Instructions</td>
<td>Allows the user to enter program-specific help instructions.</td>
</tr>
<tr>
<td>Define Option and Function Key Exits</td>
<td>Allows the user to define special program exits.</td>
</tr>
<tr>
<td>Detailed Programming Facility</td>
<td>Allows the user to specify data field definition parameters for fields included in the screen, the report, and the master files.</td>
</tr>
<tr>
<td>Define Processing Options</td>
<td>Allows the user to define processing options the program can use.</td>
</tr>
</tbody>
</table>

6.2 Function Exits

Parameter Validation Monitor (F2)
Choose Parameter Validation Monitor after you enter all of the Program Generator specifications to determine if the monitor program can detect any pre-defined errors.

- This program verifies important features that are pertinent to generating source code by the Program Generator
- This program does not verify whether you regenerate the file specifications after you change your video file

The monitor program verifies that you:

- Specify $$ fields in the TOTAL formats of the report file for the a generated report program if the report includes a total column
- Define the SH#RRN field for interactive subfile programs processing by relative record number
- Define the File Information Data Structure for interactive programs processing by relative record number
- Choose a keyed master file for programs processing by relative record number
- Define a field as mandatory entry N for transaction processor programs (subfiles)
- Define a hidden field for interactive transaction processor programs
- Define the master file key fields as output
- Attach a validation file to fields that are set up to use next numbers

**Repository Functions (F6)**
Choose Repository Functions (F6) to access a window of JD Edwards World repositories. This is the same window you access from the Software Versions Repository and CASE Profiles screens.

**Software Search (F9)**
Choose Software Search (F9) to access the Software Search facility. On the Software Search facility, enter a program name (generic*) to view all program names that meet or are greater than the search criteria.

**Automatic Accounting Instructions (F13)**
Choose Automatic Accounting Instructions (F13) to access the Automatic Accounting Instructions screen. Use this screen for reference only as you cannot generate code from this screen. Data you enter on this screen appears in the AAP portion of the Help instructions you generate.

**Select All Functions (F21)**
Choose Select All Functions (F21) to access all of the Program Generator definition screens.

**Delete All Specifications (F23)**
Choose Delete All Specifications (F23) to delete all of the Program Generator specifications for the program. This removes the Pxxxxx and Hxxxxx members from the F93002 file.
7 Define Program Purpose and Type

This chapter contains these topics:

- Section 7.1, "Defining Program Purpose and Type,"
- Section 7.2, "Function Exits."

7.1 Defining Program Purpose and Type

Defining the program purpose and the program type is the first step in creating a program using the CASE Program Generator. Program types specify the basic function of the program. There are program types for:

- Interactive maintenance programs
- Programs with subfiles
- Report programs
- Conversion programs

The system:

- Stores information in the General Purpose/Type Parameters file (F93101)
- Creates the Pxxxxx member in the Additional Help/Modifications Master file (F93002)
- Creates a data item in the Data Item Master file (F9200)

The Program Purpose and Type screen includes:

- Software Action Request (SAR) number for the program
- Install system value
- Additional information from the Software Version Repository

The Dialogue Selection screen is a series of questions you use to determine the Program Type.

To define program purpose and type

1. On Define Generator Specification, enter 1 in the following field next to Program Purpose and Type.
   - Option
2. On Program Purpose and Type, complete the following fields.

   - **Program ID**
   - **Title**
   - **Purpose**
   - **Product Code**
   - **SAR Number**
   - **CAP Status**
   - **Program Type**
   - **Lockout Action Codes**
Figure 7–2  Program Purpose and Type screen

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program ID</td>
<td>RPG program name specified in the Software Versions Repository. The system adds a data item by this name, with a glossary group of P to the data dictionary as part of this program definition.</td>
</tr>
<tr>
<td>Title</td>
<td>The title defaults to the description in the Software Versions Repository and should not be changed. When help instructions are generated, this title appears as the Help program title. Serves as the alpha description for the data item previously mentioned.</td>
</tr>
<tr>
<td>Purpose</td>
<td>The full glossary of terms (Data Dictionary) definition. When entering the definition, be aware that the text is wrapped around to fit within 65 characters when being printed for Help Instructions. Therefore, if you are indenting certain parts of the definition, keep the entire line to within 65 characters to retain your desired indentation.</td>
</tr>
<tr>
<td>Product Code</td>
<td>Defaults to the system specified in the Software Versions Repository.</td>
</tr>
<tr>
<td>SAR Number</td>
<td>Defaults to the SAR entered in the Software Versions Repository.</td>
</tr>
</tbody>
</table>
Defining Program Purpose and Type

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAP Status</td>
<td>This field indicates whether the source code for a program can be generated using the program generator. The default for this field is &quot;Y&quot; (Yes, can be generated). The generation program will only generate source code for programs having a code of &quot;Y&quot;. This should be set to &quot;N&quot; (No, cannot be generated) if the Program Generator should not be used to generate the program source or if the source generation process is complete and the program has moved into production.</td>
</tr>
<tr>
<td>Screen-specific information</td>
<td>This field also indicates whether the source code for a program can be modified using the program generator. The five additional serial number fields are still included in the source file (142 characters). When the source generation process is complete and the program has moved into a production source file (92 characters)</td>
</tr>
<tr>
<td>Program Type</td>
<td>The Program Type is a name used to identify the basic functions of a program. Each program type is made up of several logic modules. Each logic module contains small sections of RPG code. The program type determines which particular logic modules go together to create the desired program.</td>
</tr>
<tr>
<td>Lockout Act (action)</td>
<td>Allows the user to specify which action codes they do not want included in the program. Any codes listed will not be allowed. That is, the program will not allow the indicator associated with the action code being locked out to ever be turned on. The source to process the Action Code will still be included but the associated indicator will never be allowed to be turned on. Utilizes array @NAC in the programs.</td>
</tr>
</tbody>
</table>

To identify program type
1. On Program Purpose and Type, choose Program Type Determination (F11).
2. On Dialogue Selection, answer the questions.
1. The following graphics illustrate the flow you use in selecting the proper program type.
Figure 7–4  Selecting the Proper Program Type

What is the general type of program?

*Interactive*

*Interactive form* $\rightarrow$ E0010

*Print a report* $\rightarrow$ B

*Conversion program* $\rightarrow$ C

*Batch update program* $\rightarrow$ D

Figure 7–5  A: Interactive Program flow

```
A Interactive Program

Does it contain a subfile?  YES | NO

Does it update the master file?  YES | NO

Does it update both the MASTER and TRANSACTION files?  YES | NO

Does the file have unique keys?  YES | NO

Does the entire subfile be balanced validated?  YES | NO

Are there selection exits?  YES | NO

Controlled by Action Code?  YES | NO

Does the file have unique keys?  YES | NO

How will the subfile be processed?  Process at | Process only changed

A0010  B0010  A0020
D0010  D0050  D0090
D0080  D0040  D0070
D0060  D0010  D0020
D0030

```
Function Exits

Figure 7–6  Three Program flow

7.2 Function Exits

Program Type Cross-Reference (F2)
Choose Program Type Cross-Reference (F2) to access Program Type Cross-Reference which allows you to view all the programs with the same program type.

Program Type Determination (F11)
Choose Program Type Determination (F11) to access the first dialogue screen if there is no program type.

When you copy a program with specifications, it is not necessary to complete the question and answer process, which the system uses to determine the program or logic type.
Work with File Specifications

This chapter contains these topics:

- Section 8.1, "What Are File Specifications?"
- Section 8.2, "Function Exits."

The program generator requires that you specify the files for your program. The program generator adds any necessary validation files and servers.

File specifications allow you to enter the database files your program uses. After you select your specific program type, continue by completing the File Specifications screen. The system stores information in F93102 and F93103 and creates the F93105 records.

A significant feature of the Program Generator is its interpretive ability to include secondary editing and referencing files.

This chapter includes the following tasks:

- To enter file specifications
- To generate source code from file specifications

### 8.1 What Are File Specifications?

A key step in generating source code is that you correctly specify the master files for a program. The database Input/Output operations depend on the files you specify.

<table>
<thead>
<tr>
<th>PROGRAM TYPE</th>
<th>DESCRIPTION</th>
<th>SPECIFICATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0010</td>
<td>SFL (IBM Subfile) Inquiry</td>
<td>Specify the master file with an M or 1 in the Input field.</td>
</tr>
<tr>
<td>A0020</td>
<td>Single Record Inquiry</td>
<td></td>
</tr>
<tr>
<td>C0010</td>
<td>Standard Report</td>
<td></td>
</tr>
<tr>
<td>C0020</td>
<td>Standard Report - Subheading</td>
<td></td>
</tr>
<tr>
<td>C0025</td>
<td>Subheading above Columns</td>
<td></td>
</tr>
<tr>
<td>E0010</td>
<td>Window</td>
<td></td>
</tr>
</tbody>
</table>
What Are File Specifications?

The Program Generator requires that you:

- Specify one master file with an M or a 1. Do not specify one file with an M and another file with a 1.
- Enter the correct function code on the Software Versions Repository for the screen or report. Otherwise, the Program Generator does not generate moves to the screen or report.

You can use non-JD Edwards World files with the Program Generator, but you must enter the file in the Software Versions Repository.

<table>
<thead>
<tr>
<th>PROGRAM TYPE</th>
<th>DESCRIPTION</th>
<th>SPECIFICATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>B0010</td>
<td>Single Record Maintenance</td>
<td>Specify the master file with an M or 1 in the Update field</td>
</tr>
<tr>
<td>D0040</td>
<td>SFL Maintenance - KEY</td>
<td></td>
</tr>
<tr>
<td>D0045</td>
<td>SFL Maintenance - KEY, No Action Code</td>
<td></td>
</tr>
<tr>
<td>X0010</td>
<td>SFL Maintenance - KEY</td>
<td></td>
</tr>
<tr>
<td>Y0020</td>
<td>Batch Update - 1 File</td>
<td></td>
</tr>
<tr>
<td>Y0030</td>
<td>File Conversion - 1 File</td>
<td></td>
</tr>
<tr>
<td>D0100</td>
<td>SFL Maintenance - KEY, 2 Update Files</td>
<td>Specify the master file, which the system maintains in the SFL Control format, with a 1 in the Update field. Specify the transaction file, which the system maintains in the SFL format, with a 2 in the Update field.</td>
</tr>
<tr>
<td>D0010</td>
<td>SFL Maintenance - RRN</td>
<td>Specify the logical file which the system uses to fill the subfile, with a 1 in the Input field. Also, enter a File Information Data Structure name for the logical file in the fold area.</td>
</tr>
<tr>
<td>D0020</td>
<td>SFL Maintenance - RRN</td>
<td>Specify the logical file which the system uses to fill the subfile, with a 1 in the Input field. Also, enter a File Information Data Structure name for the logical file in the fold area.</td>
</tr>
<tr>
<td>D0030</td>
<td>No Action code</td>
<td>Specify the physical file that the system updates with a 2 in the Update field. Also, enter N in the Key field for the physical file in the fold area.</td>
</tr>
<tr>
<td>D0070</td>
<td>SFL Maintenance - RRN</td>
<td>Specify the physical file that the system updates with a 2 in the Update field. Also, enter N in the Key field for the physical file in the fold area.</td>
</tr>
<tr>
<td>D0070</td>
<td>No Action code</td>
<td>Specify the physical file that the system updates with a 2 in the Update field. Also, enter N in the Key field for the physical file in the fold area.</td>
</tr>
<tr>
<td>D0080</td>
<td>SFL Maintenance - RRN</td>
<td>Specify the master file, which the system maintains in the SFL Control format with a 1 in the Update field. Specify the logical file that the system uses to fill the subfile with a 3 in the Input field. Also, enter a File Information Data Structure name for the logical file in the fold area. Specify the physical file that the system updates with a 2 in the Update field. Also, enter N in the Key field for the physical file in the fold area.</td>
</tr>
<tr>
<td>D0090</td>
<td>SFL Maintenance - RRN</td>
<td>Specify the master file, which the system maintains in the SFL Control format with a 1 in the Update field. Specify the logical file that the system uses to fill the subfile with a 3 in the Input field. Also, enter a File Information Data Structure name for the logical file in the fold area. Specify the physical file that the system updates with a 2 in the Update field. Also, enter N in the Key field for the physical file in the fold area.</td>
</tr>
<tr>
<td>D0050</td>
<td>SFL Maintenance - RRN, 2 Update Files</td>
<td>Specify the master file, which the system maintains in the SFL Control format with a 1 in the Update field. Specify the logical file that the system uses to fill the subfile with a 3 in the Input field. Also, enter a File Information Data Structure name for the logical file in the fold area. Specify the physical file that the system updates with a 2 in the Update field. Also, enter N in the Key field for the physical file in the fold area.</td>
</tr>
<tr>
<td>X0020</td>
<td>Batch Update, 2 Files</td>
<td>Specify the input file with a 1 in the Input field. Specify the output file with a 2 in the Update field.</td>
</tr>
<tr>
<td>X0030</td>
<td>Batch Update, 2 Files</td>
<td>Specify the input file with a 1 in the Input field. Specify the output file with a 2 in the Update field.</td>
</tr>
<tr>
<td>Y0010</td>
<td>File Conversion, 2 Files</td>
<td>Specify the input file with a 1 in the Input field. Specify the output file with a 2 in the Update field.</td>
</tr>
</tbody>
</table>
The Program Generator does not require that you enter values in the Input, Output, or Update fields for a screen or report.

After you complete the appropriate fields on the File Specifications screen, a job runs interactively that analyzes the file specifications and creates records for three Program Generator files. As the system processes this job, messages display at the bottom of the screen.

<table>
<thead>
<tr>
<th>File Description</th>
<th>File Specifications F93102</th>
</tr>
</thead>
<tbody>
<tr>
<td>The system updates this file with one record for each file in the File Specification.</td>
<td></td>
</tr>
<tr>
<td>■ If the master file includes a Business Unit field, then the system adds the Business Unit Security file (F0001) to the File Specifications.</td>
<td></td>
</tr>
<tr>
<td>■ If the master file is for an interactive program and contains a field that uses a validation file, then the system adds that file to the File Specifications. If you later decide that the file validation is not necessary, you can delete it on the File Specifications screen.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File Description</th>
<th>Data Base Format Parameters F93103</th>
</tr>
</thead>
<tbody>
<tr>
<td>The system updates this file with one record for each format in each file.</td>
<td></td>
</tr>
<tr>
<td>If the file is a database file, then the F93103 record contains the name of the Key List that the Program Generator uses, and the names of the key fields.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File Description</th>
<th>Detail Program Logic Parameters F93105</th>
</tr>
</thead>
<tbody>
<tr>
<td>The system updates this file with one record for each field in each file. The system uses the records in the Detailed Programming Facility.</td>
<td></td>
</tr>
<tr>
<td>■ If the file is a master file or device file, then the system includes all fields.</td>
<td></td>
</tr>
<tr>
<td>■ If the file is a database file that you use only for input purposes, then the system includes only the key fields.</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** If you make changes to the fields in any of the files you list in the File Specifications, you must run the File Specifications again. If you delete a field from a file, you must manually delete that field from the Detailed Programming Facility. Running the File Specifications again does not remove records from the F93105 file.

**To enter file specifications**

1. On Define Generator Specification, enter 1 in the following field next to File Specifications
   ■ Option
What Are File Specifications?

Figure 8–1  Define Generator Specification (Enter File) screen

2. On File Specifications, complete the following field:
   - File

3. Complete the appropriate field:
   - Input
   - Output
   - Update
   - Add
What Are File Specifications?

Work with File Specifications

Figure 8–2  File Specifications screen

If you specify Update for a file, the Program Generator examines all fields in that file and includes any other files necessary to edit those fields during an update.

4. Choose Extended Parameters (F4) to complete the fields in the fold area.

Figure 8–3  File Specifications (Fold) screen
## What Are File Specifications?

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>The member ID of the file used by the program.</td>
</tr>
<tr>
<td>Input</td>
<td>A code used to specify that a data file will be used as input only by the program being generated. The value entered in this field designates secondary meanings for the use of the file: &quot;M&quot; or &quot;1&quot; thru &quot;9&quot; = Master input file No. 1 thru No. 9, P = Primary input file. The &quot;P&quot; will generate the F specification as input primary S = Secondary input file. The &quot;S&quot; designates input secondary. X = Input file. Any master file designation or an &quot;X&quot; will generate the RPG file (F) specification as input full procedural. When working within a Report program type: Files are not treated as input/primary. The program forces a read of the file for control level processing. You can only define one master file read within a report.</td>
</tr>
<tr>
<td>Output</td>
<td>A code of X indicates this data file is used as output only by the program to be generated. The RPG file (F) specification will be generated with a file type of O in position 15.</td>
</tr>
<tr>
<td>Update</td>
<td>A code in this field designates that a file is to be updated within the program being generated. The value entered in this field designates secondary meanings for the use of the file: M or 1 thru 9 - Update master file P = Update primary file S or X - Update secondary file T = Update transaction file When defining a subfile transaction processor program type that updates the master file by relative record number, you must designate the keyed file as master file No. 1 and the file updated by relative record number as master file No. 2.</td>
</tr>
<tr>
<td>Add</td>
<td>A code of X specifies that a file will have records written to it in the program being generated. The data file designated as the master file in all file maintenance programs must be designated as allowing file additions. A code of X will generate an A in column 66 of the file (F) specification in RPG.</td>
</tr>
<tr>
<td>Src Lib/File</td>
<td>The Library Name field contains the name of a valid AS/400 library name. Defaults from SVR. In the Program Generator Data File parameters this library name is the library where the data file’s source file resides. For logical files it is necessary that the based on physical file’s source exist in the same source file.</td>
</tr>
<tr>
<td>Keyed (Y/N)</td>
<td>A code of Y indicates the data file being specified is keyed. A value of N indicates the file access will be by relative record number. The default value is Y. If processing by RRN, the physical file that is being updated must be specified as keyed = N.</td>
</tr>
</tbody>
</table>
8.2 Function Exits

Extended Parameters (F4)
Choose Extended Parameters (F4) to display the fold area with the library names of the source files. Default library names are in the Software Versions Repository and your library list.

Data Model (F5)
Choose Data Model (F5) to access the Work with File Relationships screen to build the data model. You must build or rebuild the Cross Reference Index before you can view the data model by choosing Cross-Ref Index from the Rebuilds and Global Updates menu (G9642).

Search (F9)
Choose Search (F9) to access the Software Search facility. On the Software Search facility, you enter a program name to view all programs that meet or are greater than the search criteria.

Notice that the system updates the fields on this screen from your File Specifications.

To generate source code from file specifications
When you generate the source, the system submits a batch job to process your program specifications. The system submits the job to the generation job queue in your CASE Profile. This naming convention for this job is your member ID with a prefix of G.

---

**Field** | **Explanation**
---|---
File Info DS | Name assigned to an RPG III file information data structure if needed for an associated data file.
  - If processing by RRN, the logical file that is used to retrieve database records must have a file information data structure name. In addition, you must define SH#RRN as a hidden field on the SFL format of the video, since the Program Generator uses this field to store the relative record number.
  - Suggested naming conventions are INFDS1, INFDS2, and so forth.
  - Used with a keyed data file that does not have UNIQUE keys.

If you use the POST operation code from IBM for a file information data structure, you must do it for every file information data structure in the program.

PF Src Lib/File | Library where the source resides for the physical file linked to the logical file.
External (Y/N) | The External File field designates to the program generator whether or not you wish the data file being specified to be an externally defined file or an internally defined file. The default is "Yes" if left blank.

Absolutely no data files specified in normal application software created by JD Edwards World and Company may be internally defined. This parameter is strictly reserved for files designated in multi-file access utility functions that deal with source files or other types of system software.
1. On Software Versions Repository, locate a member.

**Figure 8-4  Software Versions Repository (Source Code) screen**

2. Enter 15 in the following field to generate the source and help.
   - Option

3. Enter 14 in the following field to compile the program.
   - Option

4. Review the program compile and correct any errors.

5. Repeat the steps to generate and compile if necessary.
Define General Instructions

This chapter contains these topics:

- Section 9.1, "About Special Characters,"
- Section 9.2, "Special Characters within Help Instructions."

You use General Instructions to create or change program-specific help text for the program that you are creating. To work with Define General Instructions you should be familiar with:

- Entering and changing text on the Edit screen
- Using special characters
- Updating the help file

The system stores information in the Hxxxxx member of the Additional Help/Modifications Master file (F93002).

9.1 About Special Characters

Following are special characters for general instructions:

<table>
<thead>
<tr>
<th>Character</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>Must be in positions 1 &amp; 2. This causes a page skip when you print the text.</td>
</tr>
<tr>
<td>++</td>
<td>Must be in positions 1 &amp; 2 which you follow with a data item. This causes the system to enter the most current data dictionary information.</td>
</tr>
<tr>
<td>&gt;&gt;</td>
<td>Enters all help instruction records for the program after the &gt;&gt; character. This character displays only when you print the text.</td>
</tr>
<tr>
<td>//BYPASS</td>
<td>Marks the beginning of help information that the system ignores. Enter at the beginning of comment lines.</td>
</tr>
<tr>
<td>//END</td>
<td>Marks the end of help information that the system ignores. Enter at the end of comment lines.</td>
</tr>
<tr>
<td></td>
<td>Underlines text.</td>
</tr>
<tr>
<td>c</td>
<td>Underlines and highlights the text.</td>
</tr>
<tr>
<td>Character</td>
<td>Explanation</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| ~         | Highlights the text  
            Press Shift + Tilde, and then press the Space Bar.  
            Alternatively, press ALT + HEX + A1 if you do not have a Tilde (~) on your keyboard. |

## 9.2 Special Characters within Help Instructions

You can use the special characters to display fields with specific attributes. You begin and end the text selection as follows.

- To underline General Use, enter `General Use`
- To highlight Additional Features, enter `~Additional Features~`
- To underlines and highlight Special Considerations, enter `¢Special Considerations¢`

After you define the General Instructions, you must rebuild the help instructions in order to include them in the interactive Help Instructions Master file (F98HELP).

#### Figure 9–1 Help Instructions Modifications

When you choose Help Instruction from the Computer Assisted Design (CAD) menu (G92) to review the F00HELP1 file, notice that the system adds some directional statements to the general instructions you create. The format is as follows:

++ HELP100, ++ HELP200, and so on.

The rebuild automatically generates these statements in order to categorize the help instructions, and to include additional help. For example, it automatically creates field explanations and a list of functions and selections for the program. HELP100, HELP200, and so on, are entries in the Data Dictionary.

The results might display as follows:
Select General Instructions to view the program-specific help text for the program.

This chapter includes the following tasks:

- To define general instructions
- To update the help instructions

**To define general instructions**

1. On Define Generator Specification, enter 1 in the following field next to Define General Instructions:
   - Option
2. On the Edit screen, enter the program-specific help text.

   You should keep the text between columns 5 and 70 or the text will be truncated.

To update the help instructions
There are two methods to update the help instructions.

On Software Versions Repository, enter 15 in the following field to regenerate the program. Alternatively, enter 18 in the following field to rebuild the help instructions:
Figure 9–5  Columns (Inventory by Cost Center) screen
This chapter contains these topics:

- Section 10.1, "Defining Option and Function Exits,"
- Section 10.2, "Function Exit."

10.1 Defining Option and Function Exits

To add function exits (function key exits) and subfile selection options to your interactive program, use Define Option and Function Key Exits. The program generator automatically adds the standard function exits to your program, such as Prompt for valid field values (F1), Display Error Message (F7), and Exit (F3). You can highlight the function exits you want to display on line 24 of the program screen using Screen Design Aid (SDA) or Vocabulary Overrides. The program generator creates a list of function exits and selection options for the program. This allows the user to choose Display Options (F24) on any screen to display the list of all function exits and use Prompt for valid field values (F1) in the Option field to display the list of options.

The system stores information for CASE in the Program Exit Parameters file (F93104). The system creates or updates the Function Key Translation Detail (F9611) and Generic Function Key Master (F96012) tables for the specific screen.

Each set of fields on the Option & Function Key Exits screen pertains to the Function Key or Selection Option you are defining. There is a correlation between the values you enter in the Field field and the Key field. The value in the Field field is the internal data name for the program and the system compares this value to the value in the Key field. The system uses the value in the Program ID field for the name of the program or routine the system executes. The Parm fields are the parameters the system needs for the program or routine.

To define Option and Function Exits

1. On Define Generator Specification, enter 1 in the following field to define option and function exits:
   - Option
2. On Option & Function Key Exits, complete the following fields:
   - Field
   - Program ID
   - Key
   - Parm (Parameter) 1 through 8
   - Purpose of Exit
   - Returned Key Fld
   - Returned Desc Fld
### Field Explanation

**Field**

The RPG field name (6 bytes) to be passed as a parameter on function key exits or subfile options.

*Screen-specific information*

The internal field name the system assigns to each option and function exit in the program you are generating.

Correlation exists between this field and the Function exit Definitions repository.

Maintained in the soft coding server data structure (I00SC).

- This is a required field
- Use #S01 - #S16 for options
- Use #F01 - #F15 for function exits

**Program Id**

The identification, such as program number, table number, and report number, this is assigned to an element of software.

*Screen-specific information*

The name of the program that the system executes when you choose the function exit or enter a selection option value.

By prefixing the name with an asterisk (*) you may designate the name of a logic module. A logic module’s name that you use for this purpose must begin with an X followed by any eight characters. The name cannot be longer than nine characters in order to allow for entry of the asterisk prefix. This function allows the programmer to create logic other than the standard execution of an external program when a user chooses a function exit or enters a selection option.
### Field Explanation

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>This field is used to specify the function key or subfile option number for a particular exit in the program being generated. Along with function key or option number, you must enter the program ID and any parameters that should be passed to the external program.</td>
</tr>
<tr>
<td></td>
<td><strong>Screen-specific information</strong></td>
</tr>
<tr>
<td></td>
<td>You can only define function exits for #F01 through #F15 and subfile options for #S01 through #S16.</td>
</tr>
<tr>
<td>Parm 1</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 2</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 3</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 4</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 5</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 6</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 7</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Parm 8</td>
<td>The RPG field name (6 bytes) to be passed as a parameter on function exits or subfile options.</td>
</tr>
<tr>
<td>Purpose of Exit</td>
<td>A name or remark that describes an element in the JD Edwards World systems.</td>
</tr>
<tr>
<td></td>
<td><strong>Screen-specific information</strong></td>
</tr>
<tr>
<td></td>
<td>Special Use: For the CASE system, this field is used for program exit remarks and AAI remarks. It also allows you to enter a data dictionary key when prefixed with an * (asterisk). This lets you use standard explanations and provide more extensive explanations for each exit or AAI when viewed with the help instructions.</td>
</tr>
<tr>
<td>Returned Key Fld</td>
<td>This field is used to specify the name of the data field to be updated by a returned key when exiting to an inquiry program that allows passing the selected record’s key back to the initiating program.</td>
</tr>
<tr>
<td></td>
<td><strong>Screen-specific information</strong></td>
</tr>
<tr>
<td></td>
<td>Causes logic generation to let a returned key pass through the local data area and loads the value in the specified key field. Only valid with the CL program J98LDAKY</td>
</tr>
</tbody>
</table>
10.2 Function Exit

Search (F9)

Choose Search (F9) to access the Software Search facility. On Software Search, enter a program name to view all programs that meet or are greater than the search criteria. You can also enter Generic* to view the names of all program in the SVR.

10.2.1 What You Should Know About

<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
</table>
| Values in the Parameter fields | Use caution when using an internal program data name in the Parm fields. Using screen (VD prefix) or subfile (SF prefix) fields might cause issues because the program the system retrieves can change the data in the field. To avoid transferring screen or subfile fields values, alternative options for VDxxxx or SFxxxx include:  
  - Transfer PSxxxx This requires a manual source change to the program in order to properly load the PSxxxx field with the screen or subfile field, or load the field using Program Design Language.  
  - Transfer SHxxxx You can define the SHxxxx fields as hidden fields on their screen and then load them with the proper information using the Detailed Programming facility. |

Returned Desc Fld

This field is used to specify the name of the data field to be updated by a returned description when exiting to an inquiry program that allows passing the selected record's description back to the initiating program.

Screen-specific information

Causes logic generation to let a returned description pass through the local data area and loads the value to the designated description field.

- Only valid with the CL program J98LDAKY
- For more information on using the Returned Key and Returned Desc Fld, see the program level Helps for P93104
This chapter includes these topics:

- Section 11.1, "About the Detailed Programming Facility,"
- Section 11.2, "About Full Data Field Parameters,"
- Section 11.3, "Loading VC0 Description Fields,"
- Section 11.4, "Enabling the Database Update Function for Subfiles,"
- Section 11.5, "Creating *ENTRY PLIST Entries,"
- Section 11.6, "Protecting Fields from Being Cleared,"
- Section 11.7, "Disabling Data Dictionary Edits,"
- Section 11.8, "Creating a Partial KLIS for a File."

The Detailed Programming Facility allows you to specify data field definition parameters. The Detailed Programming Facility screen lists the files in order and then each field in order within the files for the shell program the Program Generator creates. It also provides access to Field Detail and Program Design Language. The system creates this specification after you enter the File Specifications. Additionally, the system stores the information in the Detail Program Logic Parameters file (F93105).

11.1 About the Detailed Programming Facility

After you enter the file specifications, you access the Detailed Programming Facility from the Define Generator Specification screen.

To access the Detailed Programming Facility, you enter 1 in the Option field on the Define Generator Specification screen.
The Detailed Programming Facility screen displays.

Field Explanation

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name</td>
<td>The member ID of the file used by the program.</td>
</tr>
<tr>
<td>O P</td>
<td>Allows for selection exits for each field.</td>
</tr>
</tbody>
</table>
### 11.1.1 Available Options

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 - Data Dictionary</td>
<td>Access the Data Dictionary Repository for the data item.</td>
</tr>
<tr>
<td>4 - Select/Work With</td>
<td>Access the Full Data Field Parameters screen for more detail on the field.</td>
</tr>
<tr>
<td>6 - Data Formula Entry</td>
<td>Access the Data Item Formula Revisions screen where you enter Program Design Language (PDL) code.</td>
</tr>
<tr>
<td>9 - Delete Record</td>
<td>Allows you to delete a field from the Detailed Programming Facility.</td>
</tr>
</tbody>
</table>

### 11.1.2 Function Exits

**Repository Services (F6)**

Choose Repository Services (F6) to access a screen of JD Edwards World technical functions or repositories.
11.2 About Full Data Field Parameters

Full Data Field Parameters allows you to create additional source code.

11.2.1 Primary Uses of Full Data Field Parameters

The primary uses of full data field parameters include:

- Loading VC0 description fields: The system can store VC0 (descriptive data) about either a screen (video) or report in another file. Use this screen to enter the file in which you want the system to store the data about the field, the file key, where you want the system to store the description and the field with which you want to associate the field.

- Enabling the database update function for subfiles: Entering N in the Entry Optional field enables the subfile field that controls data base updates.

- Creating the *ENTRY PLIST code for a program: Specifies which data fields you want to include in an *ENTRY PLIST statement and the sequence in which they will appear.

- Protecting a field from the system clearing it every time the system executes the S001 routine when you generate the code.
  - You specify N in the Clear After (Y/N) field.
  - The system requires this for output only fields that do not have a VC0 prefix.
  - The system requires this for key fields in RRN program types.

- Adding error message the user creates.

- Suppressing edits in S005 for audit fields.

- Creating a partial KLIST for an input file.

To access the Full Data Field Parameters screen, you enter 4 in the Option field next to the field for which you want to create additional source code on the Detailed Programming Facility screen.
The Full Data Field Parameters screen displays.

**Figure 11–4  Full Data Field Parameters screen**
<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source of Data</td>
<td>Indicates what information is to be loaded into the “Write To” field on the screen. This field is loaded automatically by CAP during the data field generation process initiated by adding files to the file specifications program. It is loaded based upon either a display file or report file data dictionary item name matching with the same data dictionary item name in the specified data base files. If no match occurs for the designated master file fields, this field is loaded with “<em>SKIP”. (An entry of “</em>” followed by an internal logic module name allows creating standard calculation routines for certain fields.) An entry of *PROC will replace standard code with that generated by PDL.</td>
</tr>
<tr>
<td>Dictionary Name</td>
<td>The 4 character data item name from the data dictionary. Used extensively for field editing within the program generator.</td>
</tr>
<tr>
<td>Field Type</td>
<td>Used to designate master file field names and display/report file field names within the data field parameter records.</td>
</tr>
<tr>
<td></td>
<td>M – indicates a master file field</td>
</tr>
<tr>
<td></td>
<td>P – indicates the field is in the control record portion of a video screen (so if not a subfile, all fields would be a P)</td>
</tr>
<tr>
<td></td>
<td>S – indicates the field is in the subfile portion of a video</td>
</tr>
<tr>
<td></td>
<td>D – indicates a field within a report detail format</td>
</tr>
<tr>
<td></td>
<td>H – indicates a field within a report heading format</td>
</tr>
<tr>
<td></td>
<td>T – indicates a field within a report total format</td>
</tr>
<tr>
<td>Data Field Use</td>
<td>To determine how a data item is used on a video screen or report as far as:</td>
</tr>
<tr>
<td></td>
<td>I – input only</td>
</tr>
<tr>
<td></td>
<td>O – output only</td>
</tr>
<tr>
<td></td>
<td>B – both input and output</td>
</tr>
<tr>
<td></td>
<td>H – hidden field</td>
</tr>
<tr>
<td>Key Position</td>
<td>Designates the relative position of the field in the key list. It is used in the program generator to generate key lists (KLIST). You may also define a partial key by blanking out the key position for a particular field. Just remember, partial keys should be defined from the bottom up; for example, don’t remove key position 01 if there are 4 keys in the key list.</td>
</tr>
<tr>
<td>PLIST Sequence</td>
<td>The PLIST Sequence field specifies to the Program Generator which data fields you wish to include as passed parameters on a *ENTRY PLIST statement and the sequence in which they will appear.</td>
</tr>
<tr>
<td></td>
<td>• 01 - 32 are valid</td>
</tr>
<tr>
<td></td>
<td>• Must enter as 01 and not 1</td>
</tr>
<tr>
<td></td>
<td>• If the first parameter is passed a non-blank value, an auto-inquiry will be performed</td>
</tr>
<tr>
<td>Entry Optional</td>
<td>Used with subfile maintenance programs to identify the field that controls database updates.</td>
</tr>
<tr>
<td></td>
<td>• One field needs to be designated as Entry Optional: N</td>
</tr>
<tr>
<td></td>
<td>• Defaults to a blank</td>
</tr>
</tbody>
</table>
### About Full Data Field Parameters

**Clear After (Y/N)**  
Designates to the Program Generator whether a field is always cleared at the end of each transaction entry or is only cleared when the user presses the specific function key to clear the screen.  
- **Y** – indicates the field will be cleared at the end of each transaction entry. The default is **Y**.  
- **N** – indicates the field will not be cleared unless specified by the user by pressing the appropriate function key.

**Right Adj (Y/N)**  
A code of:  
- **Y** – indicates the field should be right adjusted.  
- **N** – indicates the field should NOT be right adjusted.  
- **C** – indicates the field is a business unit and should be left filled with blanks instead of zeros.  
- **A** – indicates the field is an account number and the account number edit routine will be used for editing.  
Can only be used when the Read From field is a video field and the Write To field is a data base field.

**Center (Y/N)**  
A code of **Y** will center the data within the field when it is displayed.

**Description File**  
Used in conjunction with loading a VC0 description field.  
- Identifies the file that contains the description

**Descr. File Key**  
Specifies the key field name to use for retrieving the data description from the designated description file. Enter the field name used to chain to that file. If you need to use a KLIST, enter the KLIST name.  
If this description is coming from the User Defined Codes file, enter the field that contains the "code" portion of the User Defined Codes key. For example, to retrieve state description using the Address Book file, you would enter ABADDS not DRKY01. The system code and record type will be retrieved from the data dictionary item for state code (ADDS).  
The program generator will produce the code to chain to the appropriate file and move (left justified) the description to the specified output field (usually VC0xxx).

**Dictionary Edit**  
Controls the generation of data dictionary editing for fields in the master file.  
- **Defaults to Y**  
- Specifying **N** will result in no data dictionary editing for the value that is moved to a master file field  
- Is useful for audit fields such as User ID that can be loaded from the Program Status Data Structure and need no editing.

**Error Msg No**  
Identifies a custom error message to use when errors are detected on a screen field.  
- Loads the value in array EMK of subroutine S999

**Validation File**  
Specifies the file name to use for validating the current data field contents. This file name is automatically provided from the data dictionary if it exists.
11.2.2 Function Exits

File Field Description Screen (F16)
Choose File Field Description Screen (F16) to access the File Field Description Screen. This function exit is field sensitive.

- If the cursor is in the Description File Key field, and this field:
  - Contains a file name, the system enters the fields on the screen with the fields from the description file.
  - Is blank, the system displays a blank File Field screen for you to enter a file name and then displays the fields from that description file.
- If the cursor is not in the Description File field, the system displays a blank File Field screen for you to enter a file name and then displays the fields from that description file.
11.3 Loading VC0 Description Fields

The Detailed Programming Facility allows you to specify what file to use to access a description for a screen (video) or report description field whose prefix is VC0 (VC0xxx).

11.3.1 Example

In the following example, QXXCC is a field in the Business Unit Master table (F0006) that contains the business unit value. The F0006 table also contains descriptions of the business units. MCDL01 is the field in the F0006 table that contains the business unit description and the system enters this description into VC0001.

**Figure 11–5 Full Data Field Parameters (Example 1) screen**

![Full Data Field Parameters (Example 1) screen](image)

11.3.2 Example: User Defined Code

If you are accessing a description for a user defined code (UDC) field, you enter F0005 (User Defined Codes table) in the Description File and the value for the field for which you are accessing the description in the Description File Key field.

The Program Generator retrieves the Install System Code and User Defined Code Type from the Data Dictionary and builds the composite key to access the User Defined Code file.
In the example above, QXXTY is a UDC field in the User Defined Codes table. F0005 contains descriptions of UDCs. DRDL01 is the field in F0005 that contains the UDC description and the system enters this description into VC0002.

Because a server program accesses the F0005 table, it is not necessary to include it in the File Specifications.

Loading field descriptions using this approach only works if the system enters the field description into a VC0 field.

Specifying a file does not guarantee that the system enters the file you specify into the File Specifications. You must review the File Specifications to ensure the files from which you want to retrieve descriptions are present. The exception is for files that you access with a server program.

**To load the VCO Description fields**

Complete the following fields:

- Source of Data
- Description File
- Descr. File Key
### Enabling the Database Update Function for Subfiles

If you are designing a subfile maintenance program, you must define at least one field in your subfile as a required field.

#### Field Description Parameters (VCO) Screen

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source of Data</td>
<td>Indicates what information is to be loaded into the &quot;Write To&quot; field on the screen. This field is loaded automatically by CAP during the data field generation process initiated by adding files to the file specifications program. It is loaded based upon either a display file or report file data dictionary item name matching with the same data dictionary item name in the specified data base files. If no match occurs for the designated master file fields, this field is loaded with &quot;<em>SKIP&quot;. (An entry of &quot;</em>&quot; followed by an internal logic module name allows creating standard calculation routines for certain fields.) An entry of &quot;*PROC&quot; will replace standard code with that generated by PDL.</td>
</tr>
<tr>
<td>Description File</td>
<td>Used in conjunction with loading a VCO description field.</td>
</tr>
<tr>
<td>Descr. File Key</td>
<td>Identifies the file that contains the description</td>
</tr>
</tbody>
</table>

*If this description is coming from the User Defined Codes file, enter the field that contains the "code" portion of the User Defined Codes key. For example, to retrieve state description using the Address Book file, you would enter ABADD0S not DRKY01. The system code and record type will be retrieved from the data dictionary item for state code (ADDS). The program generator will produce the code to chain to the appropriate file and move (left justified) the description to the specified output field (usually VC0xxx).*

---

**11.4 Enabling the Database Update Function for Subfiles**

If you are designing a subfile maintenance program, you must define at least one field in your subfile as a required field.
In the following example, the Item ID (SFXIT) field is the field that controls database updates:

The system performs the following:

- If Item ID is blank, but there is a database record for the subfile record, then the system deletes the database record.
- If Item ID is not blank, then the system saves or updates the database depending on whether the database record exists in the subfile.

You must define one or more hidden fields in the subfile record if the program type uses:

- Key processing for the subfile. The hidden field must be SHxxxx, where xxxx is the data dictionary item. There must be a SHxxxx field for each key field that is in the subfile record.
- RRN processing for the subfile. The hidden field must be SH#RRN.

To enable the database update function for subfiles
On the Full Data Field Parameters screen, enter N in the Entry Optional field.

**Figure 11–8  Full Data Field Parameters (Update) screen**

11.5 Creating *ENTRY PLIST Entries

You use PLIST entries to define which data items to include in a parameter list. You can use a maximum of 32 parameters.

11.5.1 Example

The system uses data item VDXCC as the third parameter in the entry list of Subroutine S999. The program generator creates a field name, which is the same data dictionary item with a prefix of ##. The system moves this parameter field to VDXCC from the parameter field.
You must use the data item in the display file, not the database file, for creating PLIST parameters.

**Figure 11–9  Full Data Field Parameters (Example 3) screen**

If the parameter value is not blank, the system updates the variable $AUTO with a 1. When the system launches this program, this parameter informs the program to perform an automatic inquiry (S003).

**Figure 11–10  Required Program Parameters screen**

If the system retrieves this program directly, the CL program retrieving this RPG program must issue a blank parameter.
To create *ENTRY PLIST entries
On the Full Data Field Parameters screen, enter a two-digit number corresponding to the sequence of the parameter in the PLIST Sequence field.

Figure 11–11  Full Data Field Parameters (PLIST) screen

11.6 Protecting Fields from Being Cleared

This feature is useful when creating data entry programs with a repetitive data field. For example, when there are multiple occurrences of the date field on a screen, a user only needs to enter a date in the first occurrence of the date field.

- The system clears all fields except those with a prefix of VC0 each cycle in Subroutine S001.
- The default value for this field is Y.
- The function exit F22 clears all fields.

To protect fields from being cleared
On the Full Data Field Parameters screen, enter N in the Clear After field.
11.6.1 What You Should Know About

<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Error Messages</td>
<td>Updating the Error Message Number and Error Index fields adds errors to the EMK array in Subroutine S999.</td>
</tr>
<tr>
<td></td>
<td>For example, in Subroutine S999, the system applies error message 1684 to Error Index 21 of the EMK array. JD Edwards World reserves indexes 1 to 20.</td>
</tr>
<tr>
<td></td>
<td>You must modify your source to use this error message index.</td>
</tr>
</tbody>
</table>
Figure 11–13  Full Data Fields Parameters (Error Message) screen

11.7 Disabling Data Dictionary Edits

This feature is useful if you add custom validation through the Source Entry Utility (SEU).

To disable Data Dictionary edits
Enter N in the Dictionary Edit field.

Figure 11–14  Full Data Field Parameters (Disable Data Dictionary) screen
11.8 Creating a Partial KLST for a File

The Program Generator displays the full key list. You can change the key position to exclude subordinate elements.

11.8.1 Example

In the example that follows, the key list sequence for the Business Unit security file is:

- User ID
- File Name
- Through Business Unit

If you want to use a partial key to access this file, start with the highest number key field and clear the Key Position (KY PS) field. If you need both the full key list and a partial key list, enter this using the SEU.

To create a partial KLST for a file

On Detailed Programming Facility, clear the Key Position (KY PS) field, starting with the last element.

In the following example, clear the Key Position (KY PS) field for CC - Thru. The key for F0001 is the User ID and File Name, key positions 01 and 02.

*Figure 11–15  Detailed Programming Facility (KLST) screen*
This chapter contains these topics:

- Section 12.1, "Overview,"
- Section 12.2, "Example - Interactive Programs Using Processing Options,"
- Section 12.3, "Example - Report Program Using Processing Options,"
- Section 12.4, "Defining Processing Options."

12.1 Overview

Processing options allow individual programs to perform in many different ways. They are analogous to mechanical switches that you set before you run the program. Processing options allow users to enter parameters prior to running a program which cause varied outcomes of the program. Processing options:

- Control which fields appear on the data entry screen
- Control how the program processes data
- Set up certain default values for entry

The Define Processing Options function allows you to define processing options that the program uses. After you define the processing options, you can access them through the DREAM Writer versions list and change the processing values for a specific version. To define processing options you must be familiar with the Processing Options Setup screen.

The system stores information in the DREAM Writer Master Parameter file (F98301). The system retrieves the processing options in the housekeeping subroutine S999 by including a copy member which accesses a program to retrieve the values and enter them into array @OP.

When you define processing options, you must be aware of the following:

- The system makes the following changes to the RPG source code:
  - Creates an O record type in file F98301.
  - Brings in /COPY statement for E81DRPT.
  - Brings in /COPY statement for C81DRPT.
  - Brings in EXSR C81DRPT statement in the housekeeping subroutine S999.
  - Loads processing options to array @OP, which has 99 elements of 25A.
- You must add code manually via the Source Entry Utility (SEU) or Program Design Language (PDL) to use the processing options in a program.
You add source code in the housekeeping subroutine S999 to move the processing option into a program work field.

Example: MOVEL @OP,1 $PO1 2

The system uses the program work field with PDL, or you can manually add source code to the program via SEU. A PDL example follows:

```plaintext
\ If document type is blank, \n\ use Processing Option as default \nbegin
  If VDDCTO = ' ' Then
    VDDCTO := $PO1;
  end
```

The program needs to have values for Program ID (PSPID) and Version ID (PSVERS) to retrieve the processing option values from the DREAM Writer parameter file:

- If your program is a report program, the system generates PSPID and PSVERS automatically as PLIST parameters. Therefore, you do not need to do anything.
- If your program is an interactive program, you must add PSPID(10) and PSVERS(10) as the first two PLIST parameters. Remember to modify any programs that launch this program so that the system delivers these two parameters.

### 12.1.1 What You Should Know About

<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Generator</td>
<td>The program generator does not include code that controls interactive processing options. The program generator does control some of the tasks for batch programs. The text you enter on the Processing Options Setup screen does not convey instructions to the program generator.</td>
</tr>
</tbody>
</table>

### 12.2 Example - Interactive Programs Using Processing Options

Following is an example of an interactive program containing processing options.

When you search in SEU for string C9803, the following screen displays:
This code copies the E Specs that relate to the common subroutine. This code defines all necessary arrays and tables for the copy module C9803. Typically, you add copy statements such as this at the end of the E Specs.

The next section of code in this program that relates to processing options is as follows:

The system loads the @OP array for the processing options. @OP1 contains the first of 99 processing option values, which is the value the user enters in the processing option you assign to position 1. The system then moves it into another program field to use.

The next section of code that relates to processing options is as follows:
12.3 Example - Report Program Using Processing Options

The program generator builds segments of code that the system requires to processing the processing options. The code that relates to report processing options exists in two locations. The locations are where the program generator copies the:

- Extension Specifications that relate to the common subroutine for retrieving processing options.
- Calculation Specifications that relate to the common subroutine for retrieving processing options.

12.4 Defining Processing Options

This section includes the following tasks:

- To enter processing options
- To view code for the processing options

To enter processing options

1. On Define Generator Specifications, enter 1 in the Option field to access Define Processing Options:
Figure 12–4 Define Generator Specification (Processing Options) screen

2. Complete the following fields on Processing Options Setup:
   - Sequence
   - Text
   - Option Number
   - Date (0/1/)
   - RJ (Right Justify)
   - Text Only
   - DL (Display Level)
   - Field Name
### Field | Explanation
--- | ---
Seq | Specifies how the processing option text lines should be ordered on the screen. Not input capable.
Text | The descriptive text for the processing option.
Opt Nbr | The Processing Option Number field specifies for DREAM Writer processing options the array index position for each processing option. This number should never change once assigned. The sequence number of processing options may be changed to allow for better presentation on the Processing Options Entry program but the processing option number should never be changed. This field is not input capable for existing lines of text.

**Screen-specific information**
You can change the sequence number of processing options to allow for better presentation on the Processing Options Entry program, however, you should never change the processing option number because the program includes code specific to the array position for the Processing Option value.

Date (0/1) (0/1/2) | The Date Field specifies whether or not the processing option refers to a date.
--- | ---
Valid values are:
0 – Indicates that the information is not a date.
1 – Indicates that a date is to be stored in the processing option as a Gregorian date in month, day and year format.
2 – Indicates that a date is to be stored in the processing option as a Julian date in century, year and day format.
3 – Indicates the same as a “2” with the exception that the display AND entry format is “YYYY/MM/DD” (full four digit year).

**NOTE:** All data entry for date information is entered in SYSTEM FORMAT with the exception of the “3”. 

---

**Figure 12–5  Processing Options Setup screen**
To view code for the processing options

1. On Software Versions Repository, locate the program for which you are adding processing options.

2. Enter 1 in the Option field next to the line in the subfile for the program.

   The code for the program displays on the Browse screen.
3. Scan for the following instances within the code:

- Where you instruct the compiler to retrieve the requisite source for the Extension Specification that relate to the C9803 subroutines.
- Where you interpret and act upon the values in the processing options.
- Where you instruct the compiler to copy the source for the calculation specifications that relate to the C9803 subroutine.

12.4.1 Function Exits

**Repository Services (F6)**
Choose Repository Services (F6) to access a screen with a list of JD Edwards World repositories.

**Language Preference Text (F18)**
Choose Language Preference Text (F18) to access a screen that you use to enter language specific processing options.
This part contains these chapters:

- Section 13, "Overview to Program Design Language,"
- Section 14, "About PDL Statements and Syntax,"
- Section 15, "Understand Additional PDL Operations."
This chapter contains these topics:

- Section 13.1, "Objectives,"
- Section 13.2, "About PDL."

13.1 Objectives

- To work with Data Item Formula Revisions
- To understand Program Design Language (PDL) statements and syntax
- To understand PDL editing, parsing, and source generation

13.2 About PDL

Use PDL to create specifications within the Detailed Programming Facility that causes specialized source code to generate. Use PDL for calculations or comparisons. When the program generates, the program generator converts the code into RPG source code.

Enter PDL code prior to the standard code that the program generator creates. If you want the PDL code to follow the standard code for a field, enter the PDL code on the field immediately following the field with which it is associated. The program generator creates all source code for fields in alphabetical order.

CASE stores PDL in the User Defined Procedures file (F93109) with one record per formula. The User Defined Procedures Detail file (F93110) divides the F93109 file into statements. The F93110 file contains multiple records for each formula.

PDL checks variable definitions as follows:

- Checks the variable to see if it is a keyword
- Checks for the variable in the RPG program
  - If not in the program, checks to see if it exists in the Data Dictionary Repository
  - If not in the Data Dictionary Repository, the user must define the variable

The PDL uses:

- Data Item Formula Revisions screen
- PDL Statements
- Blocks of Statements
About PDL

- Comments
- Assignments
- Database Operations
- Calls
- Loops
- Conditions
- Miscellaneous Keywords and Syntax

Perform the following tasks:
- Work with Data Item Formula Revisions
- Understand PDL Statements and Syntax
- Understand Additional PDL Operations
14

About PDL Statements and Syntax

This chapter contains these topics:

- Section 14.1, "About PDL Statements,"
- Section 14.2, "About Blocks of Statements,"
- Section 14.3, "About Comments,"
- Section 14.4, "About Assignments,"
- Section 14.5, "About Database Operations,"
- Section 14.6, "About Program Calls,"
- Section 14.7, "About Loops,"
- Section 14.8, "About Conditions,"
- Section 14.9, "About Miscellaneous Keywords and Syntax."

A Data Item Formula consists of Program Design Language (PDL) statements. PDL statements form the following types of operations.

- Blocks of statements
- Comments
- Assignments
- Database operations
- Program calls
- Loops
- Conditions
- Miscellaneous keywords and syntax

You must use specific syntax when you work with PDL statements.

14.1 About PDL Statements

A PDL statement combines one or more of the following elements:

- Keywords
- Variables
- Database Files
- Operators
14.1.1 Keywords

Keywords are the vocabulary of PDL. They identify the type of operation the statement performs.

14.1.2 Variables

The following are valid variable names in PDL statements:

- Database field names
  Examples: ABAN8, MCDL01
- Screen and report field names
  Examples: VDDOCO, SFTRDJ, VC0001, RR#CLS
- Data Dictionary
  You can use Data Dictionary fields in PDL. The system uses the data type and size as they are defined in the Data Dictionary.
- Indicators
  You can use indicators by using the names IN01 to IN99. You can also use INLR. You can use both of these in PDL assignment statements to set on or off, and in conditional expressions to test for on or off.
  Example:
  in98 := '0'
  If in98 = '0' Then
  Note: In PDL, the system does not use * with indicators. That is, you specify indicator 01 as in01 and not *in01.
- Program Workfields
  Any name that PDL recognizes as a variable, but is not a database field name, Data Dictionary field, screen or report field, or indicator, the system considers as a program work field. PDL will prompt you to define its data type.
  Examples: $#am1, $#xtp, $po1
  Be aware that if you did not yet generate the source code, PDL is not able to search the source code to find a definition.

14.1.3 Database Files

You must first define a database file name in the File Specifications before you can use it in one of the database I/O statements. PDL does not add file names to the specifications.

14.1.4 Operators

You define the valid assignment and arithmetic operators.
14.1.5 Constants

You specify alpha constants by enclosing them in single quotes. You specify numeric constants without quotes.

Examples:

vc0001 := 'Proof Mode';
$#am1 := 0;

PDL does NOT recognize the RPG constants such as *BLANK or *ZERO.

14.1.6 Punctuation

The basic PDL punctuation is a semi-colon (;), which you must use to separate PDL statements.

14.2 About Blocks of Statements

14.2.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| Begin    | Initiates a block of statements.  
|          | The syntax is: Begin |
| End      | Terminates a block of statements initiated by the Begin statement.  
|          | The syntax is: End |

14.2.2 Rules

- You must enter all Data Item Formulas within a Begin...End block. A comment statement may precede the Begin statement. For example:

  \ Use system date as default. \  
  Begin  
  If vdtrdj = '' Then  
  vdtrdj := $$edt;  
  End

- You must separate all statements within a Begin...End block by a semicolon. For example:

  \ Load A/B name to vc0 field. \  
  Begin  
  aban8 := q3an8;  
  chain f0101la;  
  If in98 = '0' Then  
  vc0003 := abalph;  
  End
You can nest Begin...End up to a maximum of 50 levels. For example:

```
\ Computer counter. \nBegin
If zaclst = '900' Then
  Begin
  rr#nin := ' 1';
  $#nin := 1;
  End;
If zaclst < '900' Then
  Begin
  rr#nin := ' 0';
  $#nin := 0;
  End
End
```

### 14.3 About Comments

#### 14.3.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>\ (backslash)</td>
<td>Initiates and terminates a comment.</td>
</tr>
<tr>
<td></td>
<td>The syntax is: \text \</td>
</tr>
<tr>
<td></td>
<td>You must enclose all comments within a pair of backslashes.</td>
</tr>
</tbody>
</table>

#### 14.3.2 Rules

Comment lines must not exceed 50 characters.

For example: Initial Comment

```
\ Compute extended amount. \nBegin
$#xtp := q2xqt * q2uncs;
End
```

For example: Embedded Comment

```
Begin
$#am1 := 0; \ Order Total \n$#xtp := 0; \ Extended Amount \nEnd
```
14.4 About Assignments

14.4.1 Operator and Syntax

<table>
<thead>
<tr>
<th>Operators</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>:=</td>
<td>The assignment operator. The system assigns the first variable the value of the variable or expression following the operator. The syntax is: variable := expression;</td>
</tr>
<tr>
<td>+</td>
<td>Add</td>
</tr>
<tr>
<td>-</td>
<td>Subtract</td>
</tr>
<tr>
<td>*</td>
<td>Multiply</td>
</tr>
<tr>
<td>/</td>
<td>Divide</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;</td>
</tr>
<tr>
<td>SST</td>
<td>Substring</td>
</tr>
</tbody>
</table>

The syntax is: variable := SST (field, n1, n2)

n1 = start position
n2 = length of string

14.4.2 Rules

You can use standard notation using parentheses for arithmetic operations. For example:

```
in98 := '0';
vremk := 'NOT DEFINED';
sfrdj := $edt;
$am1 := $am1 + (qzqty * qzcst);
$wrk := 100;
abalph := vd#fnm |> vd#lnm;
$cc := SST (qxxcc,3,10)
```

14.5 About Database Operations

14.5.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain</td>
<td>Provides for random data base processing. The syntax is: CHAIN file;</td>
</tr>
</tbody>
</table>
14.5.2 Rules

You must first define the file in the program using the File Specifications before you enter it in the statement.

The Chain, Poseq, Posgt, and Reade statements use the default KLIST name that the system generates for the file you specify.

You should assign a value to each field of the KLIST prior to entering the statement.

Specify indicator 98 in the statements to signify that the system did not retrieve a record for the program.

Specify indicator 99 in the statements to signify that a database operation error took place.

\ Load A/B name to vc0 field. \n
Begin
aban8 := q3an8;
chain f0101la;
If in98 = '0' Then
   vc0003 := abalph;
End
14.6 About Program Calls

14.6.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call</td>
<td>Allows you to execute another program. The syntax is: CALL variable;</td>
</tr>
<tr>
<td>Parm</td>
<td>Allows you to deliver parameters to a program that the program call statement executes. The syntax is: PARM variable;</td>
</tr>
</tbody>
</table>

14.6.2 Rules

Neither the Call statement nor the Parm statement allows the use of constants.
Prior to the Call statement you must enter an assignment statement to load a variable with the name of the program to launch, and load one or more variables with the values of the parameters.
The Parm statements must immediately follow the Call statement.
For example:
Begin
##pid := 'P1540 ';
##vers := 'ZJDE001';
##doco := nrdoco;
Call ##pid;
Parm ##pid;
Parm ##vers;
Parm ##doco;
End

14.7 About Loops

14.7.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Until</td>
<td>Provides for loop processing where the system evaluates a condition at the bottom of the loop. Translates to DOU in the RPG code. The syntax is: UNTIL (condition) DO (Statement)</td>
</tr>
</tbody>
</table>
14.7.2 Rules

The Do keyword is an integral part of the loop statement.
The statement following Do can be a single statement, or a block of statements within a Begin...End block.
The action is simply two expressions that you separate.
For example:
Begin
While in98 = '0' Do
  Begin
  $#xtp := q2xqt * q2uncs;
  $#am1 := $#am1 + $#xtp;
  reade f59422;
  End
  End
End

14.8 About Conditions

14.8.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>If</td>
<td>Provides for conditional processing.</td>
</tr>
<tr>
<td></td>
<td>§ The condition is two expressions that you separate by a relationship.</td>
</tr>
<tr>
<td></td>
<td>§ The data types of the expressions have to match. For example, alpha to alpha, numeric to numeric.</td>
</tr>
<tr>
<td>Then</td>
<td>Specifies the starting point for all actions the system takes when the condition of the If statement is met.</td>
</tr>
<tr>
<td>Else</td>
<td>Enter these statements following the If and Then statements. The system executes these statements when the condition of the If statement is not met.</td>
</tr>
<tr>
<td></td>
<td>The Then keyword is an integral part of the If statement.</td>
</tr>
<tr>
<td></td>
<td>§ The statement following the Then keyword can be a Begin/End block to allow for a block of statements when the condition is met.</td>
</tr>
</tbody>
</table>
- The Else statement can follow the statements you enter with If (condition) and Then (statement).
- The syntax is: IF (condition) THEN (statement) ELSE (statement)

### 14.8.2 Symbols

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equal</td>
</tr>
<tr>
<td>≠</td>
<td>Not Equal</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater Than</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less Than</td>
</tr>
<tr>
<td>≥</td>
<td>Greater Than or Equal To</td>
</tr>
<tr>
<td>≤</td>
<td>Less Than or Equal To</td>
</tr>
</tbody>
</table>

### 14.8.3 Rules

You do not have to enter the semicolon (;) to end the statement following the Else, or the Then when there is no Else.

For example, a simple If...Then statement:

```
Begin
  If sftrdj = '' Then
  sftrdj := $$edt
End
```

For example, an If...Then...Else statement

```
Begin
  If in98 = '0' Then
    vc0003 := abalph
  Else
    vc0003 := 'NOT DEFINED'
End
```

If you nest Begin/End blocks between the Then and Else statements, you should use the semicolon after each individual statement but not following the End.

For example, an If...Then with a Begin...End statement

```
Begin
  If zaclst = '999' Then
    Begin
      rr#nin := '1';
      $#nim := 1;
    End
  End
End
```
For example, an If...Then...Else with Begin...End statement

Begin
If zaclst = '900' Then
    Begin
        rr#nin := '0';
        $#nin := 0;
    End;
Else
    If zaclst < '900' Then
        Begin
            rr#nin := '<0';
            $#nin := 1-;
        End;
    Else
        Begin
            rr#nin := '>0';
            $#nin := 1;
        End
    End

14.9 About Miscellaneous Keywords and Syntax

14.9.1 Keywords and Syntax

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| Include  | Provides the ability to include other PDL modules in the User Calculation.  
           | The syntax is: INCLUDE (module name) |
| Return   | Provides for user specification of the Source of Data alone or as the result of a series of procedures.  
           | The syntax is: RETURN variable; |

14.9.2 Rules for Include

You can keep PDL modules in the form of a copy book by designating *FORMULA in the Program ID field and *LIBRARY in the File ID field on the Data Item Formula Revisions screen.
The include module should have a unique name, for example @CONCAT.

It is JD Edwards World naming convention to begin module names with the @ symbol.

The keyword include causes the Program Generator to automatically generate the appropriate code for the include module. This prevents the need to reenter user calculations that are necessary in numerous programs.

Following is an example of an include module and the include statement that calls the module.
14.9.3 Rules for Return

Specifying the Return keyword is the same as entering *PROC in the Read From field in the Detail Programming Facility.

The system omits all standard processing for this data field. In other words, by specifying the Return keyword, the system uses the code the PDL generates instead of any standard logic.

For example:
begin
$#b1 := 0;
$#b1 := q2xqt * q2uncs;
return $#b1
end
Understand Additional PDL Operations

This chapter contains these topics:

- Section 15.1, "Editing,"
- Section 15.2, "Parsing,"
- Section 15.3, "Source Code Generation,"
- Section 15.4, "Add PDL to a Field,"
- Section 15.5, "Function Exits."

15.1 Editing

The Data Item Formula is one long continuous field. If there is an error, the entire field displays in reverse image, and the system places the cursor in the field following the error. You can display the error messages by choosing Display Error Message (F7).

15.2 Parsing

The system stores the Data Item Formula in the File Specifications database in two forms:

- The generator stores the unparsed form in the User Defined Procedures file (F93109), with one record for one formula.
- The generator stores the parsed form in the User Defined Procedures Detail file (F93110), with multiple records for each formula. Each record corresponds to an RPG operation code.

15.3 Source Code Generation

The generator merges the PDL code into the program based on the field you enter in the Detailed Programming Facility.

<table>
<thead>
<tr>
<th>Read From</th>
<th>Write To</th>
<th>Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) QXXIT</td>
<td>VDXIT</td>
<td>S004 (moves database fields to video fields)</td>
</tr>
<tr>
<td>2) VDXIT</td>
<td>QXXIT</td>
<td>S005 (moves video fields to data base fields)</td>
</tr>
</tbody>
</table>
The generator places the PDL code before the standard code Program Generator code for the field in the Write To field. If you want the code the PDL generates to replace the standard code, then enter *PROC in the Read From field.

**Note:** Use caution when performing this as the system performs no editing or formatting of the field, except what you enter in the Data Item Formula.

If you want the code the PDL generates to come after the standard code the Program Generator generates for the field in the Write To field, then you must place the Data Item Formula on the field immediately following the Write To field in the Detailed Programming Facility.

The generator lists the fields in the Detailed Programming Facility in alphabetical order, and you cannot change the order.

### 15.3.1 Data Item Formula Examples

Two examples illustrate the PDL statements and syntax. Both are from an inquiry program with a subfile.

#### 15.3.1.1 Example User Defined PDL

The example illustrates a data item formula for a user defined PDL entry point in subroutine S004. The purpose of the formula is to determine if the database record meets the search criteria for order number ($SEL = 0 to omit). The program work fields $doco1 and $doco2 contain the lower and upper values for the inquiry search fields with a subfile.

*Figure 15–1 Data Item Formula Revisions (User Defined PDL) screen*

This example also illustrates the following types of PDL statements:
Additionally, this example illustrates the nesting of conditions:

If vddoco = '' Then
If q1doco < $doco1 Then
???$sel := '0';

The statement separator, the semicolon, is not necessary until the outermost If...Then statement is complete.

15.3.1.2 Example Subfile Field

This example illustrates a data item formula for a subfile field that is a computed field. The program is locating sales order header records. The computed field is the order total and is based on the sales order detail records in F59422.

![Figure 15–2 Data Item Formula Revisions (Subfile Field) screen](image)

This example illustrates the following types of PDL statements:

<table>
<thead>
<tr>
<th>Type of PDL Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment</td>
<td>$sel := '0'</td>
</tr>
<tr>
<td>Blocks</td>
<td>begin...end</td>
</tr>
<tr>
<td>Comment</td>
<td>\ Test order number for inclusion. \</td>
</tr>
<tr>
<td>Condition</td>
<td>If q1doco &lt; $doco1 Then $sel := '0';</td>
</tr>
<tr>
<td>Assignment</td>
<td>$#ami := 0; $#xtp := 0;</td>
</tr>
<tr>
<td></td>
<td>q2doco := gldoco;</td>
</tr>
</tbody>
</table>
You can separate statement separators in the loop statements within the Begin...End block, and then follow the end statement with a separator.

### 15.4 Add PDL to a Field

**To work with Data Item Formula Revisions**

1. On Software Versions Repository, locate a program and then access Define General Specifications.
2. On Define General Specifications, enter 1 the Option field to access Detailed Programming Facility.
3. On Detailed Programming Facility, enter 6 in the Option field next to the field for which you want to add PDL.

*Figure 15–3 Detailed Programming Facility (Revisions) screen*

The Data Item Formula Revisions screen displays.
4. Enter the PDL statements for the field in the Data Item Formula area.

<table>
<thead>
<tr>
<th>Field</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program ID</td>
<td>The RPG program name defined in the Software Versions Repository Master table.</td>
</tr>
<tr>
<td>File ID</td>
<td>The member ID of the file used by the program.</td>
</tr>
<tr>
<td>Field Name</td>
<td>This specifies the field name as it is identified in the file.</td>
</tr>
<tr>
<td>Data Item Formula</td>
<td>A set of Program Design Language (PDL) statements, which are then translated into RPG code.</td>
</tr>
</tbody>
</table>

15.5 Function Exits

**Display Variable Definitions (F5)**
Choose Display Variable Definitions (F5) to access a screen with a list of variable definitions.

**Repository Services (F6)**
Choose Repository Services (F6) to access a screen with a list of JD Edwards World technical functions or repositories.
This part contains these chapters:

- Chapter 16, "Overview to Source Modifications,
- Chapter 17, "Change Generated Source Code,
- Chapter 18, "Regenerate Source Code,
- Chapter 19, "Work with Model Control Language Programs."
This chapter contains these topics:

- Section 16.1, "Objectives,"
- Section 16.2, "About Source Modifications."

16.1 Objectives

- To change source code
- To regenerate source code
- To copy and customize Model Control Language (CL) Programs

16.2 About Source Modifications

After you generate source code, you can make modifications to it by using the Program Generator steps or using Source Entry Utility. When you make changes to your CASE specifications, you need to regenerate the source. CASE allows for continual improvement and modification to your original specifications.

Perform the following tasks

- Change generated source code
- Regenerate source code
- Work with model central language programs
This chapter contains the topic:

- **Section 17.1, "Pre-SEU and Post-SEU Process."**

After you generate code for a specific program, you can enter any necessary modifications to the code by using the Source Entry Utility (SEU). The program generator maintains a copy of all modifications in a separate file. When you make changes to the files for the program, or the detailed field definition parameters, you can regenerate the program source to reflect the changes and maintain your modifications.

The Program Generator indexes the modification lines based on the data the system maintains in column positions 81 through 113. After the program generates, it merges the modification lines in to the generated code according to their index values.

There are two different methods to change generated source code.

### 17.1 Pre-SEU and Post-SEU Process

A front-end JD Edwards World program, MPxxxx, monitors the changes on the SEU. The MPxxxx job does not have to finish before you recompile. The system automatically merges the changes on the SEU when you generate the program, not when you compile. The system stores all changes on the SEU in the Pxxxxx member in the Additional Help/Modifications Master file (F93002).

For source code lines that you move or copy, you must clear the serial number from column 80 onward.

You can view all changes on the SEU by entering 30 in the Option field on the Software Versions Repository.

Columns 1 and 2 include a 21 for lines you add, 22 for changes, and 23 for lines you delete.
To change generated source code
From the Software Versions Repository, locate a program and perform one of the following:

1. Enter 2 in the following field to access the JD Edwards World SEU feature
   - Option
2. On SEU, make your changes.
   When you change your program using SEU, you do not have to regenerate the code. You only need to recompile the changes.

Alternatively, after you locate a program on Software Versions Repository you can:

2. On Define General Specifications, enter 1 the following field to access Detailed Programming Facility.
   - Option

3. On Detailed Programming Facility, enter 6 in the following field next to the field for which you want to change PDL.
   - Option

4. On Data Item Formula Revisions, make your changes.
Regenerate Source Code

This chapter contains these topics:

- Section 18.1, "When to Regenerate Source Code,"
- Section 18.2, "Changing CAP Status,"
- Section 18.3, "Resolving CAP Status Invalid Error."

When regenerating source code you should know:

- When to regenerate source code
- How to solve a generation error

18.1 When to Regenerate Source Code

You should regenerate a program whenever you modify a program specification. You should regenerate your programs when you:

- Change the program type
- Add or remove a file from the program file specifications
- Change the content of a file
  
  You must repeat the File Specifications step in this situation to enter the new information into the Detailed Programming Facility.
- Add or change a function exit or selection exit
- Change a data field definition parameter, for example:
  
  - Add, change, or delete a PDL
- Add, change, or delete the Full Data Field Parameters
- Add Processing Options to a program that previously did not have any, or delete all Processing Options
- Change the Lockout Action field values

When you add, change, or remove a file in the program or change the program type, you must access the File Specifications screen from the Define Generator Specification screen and click Enter to submit the Detailed Field Specifications interactive job.

You can view all changes on the SEU by entering 30 in the Option field on the Software Versions Repository.

Columns 1 and 2 include a 21 for lines you add, 22 for changes, and 23 for lines you delete.
18.2 Changing CAP Status

If you change the CAP Status field, the system deletes the changes in the Source Entry Utility that it stores in the P member of the Additional Help/Modifications Master file (F93002). JD Edwards World recommends that you do not change the CAP Status field unless the changes you make to your program become unmanageable. When the CAP Status field is set to Y, you can regenerate your program from one JD Edwards World release to the next.

Change the CAP Status to N when either of the following occurs:

■ You test the program and are ready to move it into production.
■ You must make large manual changes that the program generator cannot generate for you. For example, adding special subroutines or complicated calculation logic.

To change CAP status
1. On Define Generator Specification, enter 1 in the following field next to Program Purpose and Type.
   - Option
2. On Program Purpose and Type, enter N in the following field.
   - CAP Status

Figure 18–1  Program Purpose and Type screen

The Delete KBG Modifications screen displays.
3. To remove the modifications member, choose Delete (F6).

The Define Generator Specification screen displays.

### 18.3 Resolving CAP Status Invalid Error

The Program Generator verifies that the job completes normally before each source generation. When the program generator does not complete normally or if you delete the specifications for a program, the system sends an error message to your workstation that states:CAP Status Invalid for program Pxxxx ... generation terminated.

If you receive a message in the job log indicating that the buffer length is longer than the record, or field AGSRCS is not found, verify that the JDESRC file in your source library exists and has a length of 142 and 8 fields.

To resolve the CAP Status Invalid error, perform any of the following:

<table>
<thead>
<tr>
<th>Possible Resolution</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensure the CAP status is set to Y on the Program Purpose and Type screen.</td>
<td>Any job that prevents the MPxxxxx job from completing normally will change the CAP Status to N.</td>
</tr>
<tr>
<td>■ Allow the MPxxxxx job to complete.</td>
<td></td>
</tr>
<tr>
<td>■ Do not cancel it in the job queue.</td>
<td></td>
</tr>
<tr>
<td>If you change the CAP Status field to N, the system deletes the changes in the Source Entry Utility that it stores in the P member of the Additional Help/Modifications Master file (F93002).</td>
<td></td>
</tr>
<tr>
<td>If the File Specifications ends abnormally, the system changes the value in the CAP Status field to D. Change the value in the CAP Status field to Y and process the file specifications.</td>
<td></td>
</tr>
<tr>
<td>Possible Resolution</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Ensure the Pxxxxx member exists in the Additional Help/Modifications Master file (F93002).</td>
<td>The Pxxxxx member must exist in order to generate a program. The system initially creates the Pxxxxx member during the Program Purpose and Type definition step.</td>
</tr>
<tr>
<td>Ensure the Mxxxxx member does not exist in F93002.</td>
<td>The Mxxxxx member must not exist in order to generate a program. Use the RMVM command to remove this member.</td>
</tr>
<tr>
<td>Ensure that one step of the generation process completes before you start the batch job of another step.</td>
<td>NA</td>
</tr>
</tbody>
</table>
This chapter contains these topics:

- Section 19.1, "Working with CL Models,"
- Section 19.2, "JD Edwards World Model CL Programs."

Control language (CL) is the primary interface between the system programmer and the AS/400 operating system. A command is a single control language statement. A series of commands can serve as source statements you can use to create a CL program. You compile the commands into a program the system calls whenever it needs the functions the program provides.

JD Edwards World provides you with a series of model CL programs that you can copy and change to call the CASE programs that you develop. You must use a CL program to add an RPG program as an option on a JD Edwards World menu.

You use the Source Entry Utility (SEU) to change the source code for your CL programs. When adding a new CL program using SEU, you can copy the JD Edwards World model CLs and tailor their specifications to fit your needs.

---

**Note:** You can view only the source code if the source code resides on your machine.

---

### 19.1 Working with CL Models

**To copy a model CL**

1. On Software Versions Repository, locate a model.
2. Enter 3 in the Option field next to the program.
3. Click Enter in the Copy Source Prompt window.
4. On Software Versions Repository, enter 2 in the Option field next to the program.
   The source code displays.
5. On the Source Entry Utility, enter the new program name.

The following illustrates the source code for J98MODEL1 using the program Help. You can also view the code using the Software Versions Repository.
To customize a CL model
1. On the Source Entry Utility, locate the lines that contain lower-case xx and make changes.
   The lines in the model that require changing contain lower-case xx. This design allows you to easily scan the code for the xx and enter your changes.
2. Exit and save the CL program.
3. Compile the program.

### 19.2 JD Edwards World Model CL Programs

JD Edwards World includes a series of model CL programs that you can copy and customize to meet your programming needs. The following table describes each model CL program.

<table>
<thead>
<tr>
<th>Model CL Programs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>J98MODEL1</td>
<td>Serves as a template for all interactive programs that do not retrieve processing options in the CL code.</td>
</tr>
<tr>
<td>J98MODEL2</td>
<td>Serves as a template for batch programs that need the DREAM Writer but have no printer file.</td>
</tr>
<tr>
<td>J98MODEL3</td>
<td>Serves as a template for interactive programs that need a prompt for parameters.</td>
</tr>
</tbody>
</table>
You can create certain model CL programs using the Quick Start CL Generator. See Chapter 25, "Work with Quick Start CL Generator" for more information.

<table>
<thead>
<tr>
<th>Model CL Programs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>J98MODEL4</td>
<td>Serves as a template for either batch or interactive programs that require the retrieval of processing options in the CL code, but do not require DREAM Writer selection or sequencing.</td>
</tr>
<tr>
<td>J98MODEL5</td>
<td>Serves as a template for batch CL programs that call report programs with fixed selection and sequencing while still passing all printer file overrides, processing options, and page-heading functions to the RPG report program.</td>
</tr>
<tr>
<td>J98MODEL6</td>
<td>Serves as a template for batch CL programs that require all DREAM Writer functions and call multiple print programs over the same OPNQRYF access path.</td>
</tr>
<tr>
<td>J98MODEL7</td>
<td>Serves as a template for batch CL programs that require all DREAM Writer functions and call multiple print programs over the same OPNQRYF access path.</td>
</tr>
<tr>
<td>J98MODEL8</td>
<td>Serves as a template for batch programs that have a control file.</td>
</tr>
</tbody>
</table>
Part V
CASE Programs

This part contains these chapters:

- Chapter 20, "Overview to CASE Programs,"
- Chapter 21, "Overview to Subfile Inquiry Programs,"
- Chapter 22, "Overview to Subfile Maintenance Programs,"
- Chapter 23, "Create Report Programs."
20

Overview to CASE Programs

This chapter contains these topics:

■ Section 20.1, "Objectives,"
■ Section 20.2, "About CASE Programs."

20.1 Objectives

■ To create CASE programs

20.2 About CASE Programs

Perform the following tasks:

■ Create Subfile Inquiry Programs
■ Create Subfile Maintenance Programs
■ Create Report Programs

Note: The Report Design Aid is currently not available in the Java platform for JD Edwards World software. You must create reports using the green screen platform of the software.
This chapter contains these topics:

- Section 21.1, "Program Type Description,"
- Section 21.2, "Display File Definition,"
- Section 21.3, "CL Program Definition,"
- Section 21.4, "File Specifications,"
- Section 21.5, "Detailed Programming Facility,"
- Section 21.6, "Special Considerations,"
- Section 21.7, "Quick Start Generation."

You can create subfile inquiry programs that allow a user to process data and run programs using an inquiry screen that you create. A subfile inquiry screen presents a subfile with information, allowing a user to view several records at one time.

The use and values you enter to create a typical Interactive subfile inquiry program follow.

### 21.1 Program Type Description

Use the Program Type Description to create an interactive subfile program that is for inquiry purposes only. This program type processes a single master file by key. You do not use Lockout Action Codes fields. Create a display file prior to generating this program type.

### 21.2 Display File Definition

The Display File Definition program type validates and changes, where necessary, the data a user enters (scrubs) the key fields in the control format of the display file prior to processing the master file. You denote the key fields by enter K in the Edited Field in the Field Definition screen of Screen Design Aid (SDA). If you are using the Data Base Field Selection feature in SDA, the system updates the key fields.

You do not need to define the Action Code, it is an optional field. Define a default cursor location if there is no action code.

### 21.3 CL Program Definition

Using the CL Program Definition, you can copy and revise the J98MODEL1 model CL Program and create a CL program for use with program type A0010. You can also use the Quick Start CL Generator to create your CL programs.
21.4 File Specifications

The File Specifications program type requires that you define a single master file and a display file. The master file contains a value of M or 1 in the Input column. The display file begins with V and the selection columns are blank. You can add files to retrieve descriptions, if necessary.

21.5 Detailed Programming Facility

The Detailed Programming Facility allows you to use a key list for record retrieval from the master file. If you are not using the complete key list, update the Key Sequence field on the Detailed Programming Facility to include only those data items which are necessary. This key list should match your key field definition from the control format of the display file.

21.6 Special Considerations

Add special logic if you want to process the master file by using the key as a restrictive key. The default logic performs a SETLL, which positions the records from the file by using the key and then reading without a key until the subfile loads.

21.7 Quick Start Generation

You can generate this program type using Quick Start.
Overview to Subfile Maintenance Programs

This chapter contains these topics:

- Section 22.1, "Program Type Description,"
- Section 22.2, "Display File Definition,"
- Section 22.3, "CL Program Definition,"
- Section 22.4, "File Specifications,"
- Section 22.5, "Detailed Programming Facility,"
- Section 22.6, "Special Considerations,"
- Section 22.7, "Quick Start Generation."

You can create subfile maintenance programs that allow a user to process data and run programs using an interactive screen you create.

The use and values you enter to create a typical Interactive Subfile Maintenance Program follow.

22.1 Program Type Description

Use the Program Type Description program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes a single master file by key. User defined selection exits and function exits are optional.

22.2 Display File Definition

The Display File Definition program type validates and changes, where necessary, the data a user enters (scrubs) the key fields in the control format of the display file prior to processing the master file. You denote the key fields by enter K in the Edited Field in the Field Definition screen of Screen Design Aid (SDA). If you are using the Data Base Field Selection feature in SDA, the system updates the key fields.

You must define the Action Code and the Lockout Action Codes are optional.

This subfile maintenance program type includes special logic which permits the system to delete individual subfile records. This logic performs when you enter a C in the Action Code. The system compares the previous value with the current value and deletes the record if the current value is blank. The system stores the previous value in a hidden field at the subfile record level using the Display All Defined Fields in SDA.
22.3 CL Program Definition

Using the CL Program Definition, you can copy and revise the J98MODEL1 model CL program to create a CL program for use with program type D0040. Use the Quick Start CL Generator to create your CL program.

22.4 File Specifications

The File Specifications program type requires that you define a single master file and a display file. The master file contains a value of M or 1 in the Update column. The display file begins with V and the selection columns are blank. Add files to retrieve descriptions, if necessary.

22.5 Detailed Programming Facility

You use the Detailed Programming Facility to access the Full Data Field Parameters screen, which contains details for the subfile field controlling the database update. By entering N in the Update the Entry Optional Y/N field, this informs the generator that the user must enter a value in this field before the system updates the database.

22.6 Special Considerations

The Special Considerations program type uses the key information in the display file for chaining to the master file. This type must also have a hidden field and an optional entry field.

22.7 Quick Start Generation

You can generate this program type using Quick Start.
This chapter contains these topics:

- Section 23.1, "Understanding RDA Special Use Fields,"
- Section 23.2, "Creating a Total Format,"
- Section 23.3, "Defining a Subheading,"
- Section 23.4, "Understanding DREAM Writer Considerations."

When using Report Design Aid (RDA) in conjunction with the JD Edwards World CASE tools, you can have the program generator assist you with totals and subheadings.

You should be familiar with the definition and use of report totals and subheadings when using the CASE tools. You should also be aware of some DREAM Writer considerations. See Work with DREAM Writer in the JD Edwards World Technical Foundation Guide for more information.

**Note:** The Report Design Aid is currently not available in the Java platform for JD Edwards World software. You must create reports using the green screen platform of the software.

### 23.1 Understanding RDA Special Use Fields

The system uses certain fields in RDA when generating reports that contain dynamic (hierarchical) totaling and subheadings. The following figure and tables illustrate how the system uses these fields within a report.
The system uses the following fields in the TOTAL1 format:

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC1ROW</td>
<td>Prints the data dictionary row description of the level break field. Default length is 30.</td>
</tr>
<tr>
<td>VC1KEY</td>
<td>Prints the value of the level break field. Default length is 12.</td>
</tr>
</tbody>
</table>
| VC1DSC  | Prints the description of the value of the break field. Default length is 30. Only works with the following fields:  
- User defined codes  
- Company Number  
- Address Book Number  
- Business Unit |

The system uses the following fields in only the HEADING2 format and therefore it uses them in only a C0020 or C0025 program type - Report w/Subheadings.

When you use subheadings, the system automatically underlines them.

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC0ROW</td>
<td>Prints the data dictionary row description of the level break field. Default length is 30.</td>
</tr>
<tr>
<td>VC0KEY</td>
<td>Prints the value of the level break field. Default length is 12.</td>
</tr>
</tbody>
</table>
| VC0DSC  | Prints the description of the value of the break field. Default length is 30. Only works with the following fields:  
- User defined codes  
- Company Number  
- Address Book Number  
- Business Unit |
In programs you generate using CASE, the level breaks are soft coded. The DREAM Writer setup determines this.

### 23.2 Creating a Total Format

When you define a total format, you can define the area of the report where the description of the level break occurs. You can display up to three pieces of information for each total level break:

- The field description of the level break
- The value of the field at the level break
- The description of that value

For example, if you choose to total your report at the business unit level, the report might read:

```
Business Unit      5      San Francisco
```

#### To create a total format

1. On Software Versions Repository, locate the report for which you want to add a total format.
2. On Software Versions Repository, enter 10 in the Option field to access the Report Design Aid.
3. On Report Design Aid, choose Record Formats List (F10).

Figure 23–2 Record Formats List report

![Record Formats List report](image)

4. On the Record Formats List screen, complete the following fields to enter the TOTAL1 format:
   - Format Name
   - Type
   - Start/End Lines
   - Fld Pfx (Field Prefix)
5. Press enter to return to Report Design Aid.
6. On Report Design Aid, enter an asterisk (*) in the column and row position to begin the total description.
The Field Definition screen displays.

**Figure 23–3 Field Definition screen**

7. On Field Definition, enter VC1ROW in the Field Name field.
8. Click Enter twice.
   
   The description for the total field replaces the asterisk (*).
   
   In the sample report that follows, when you print the report, the field contains the descriptive text Business Unit.

9. On Report Design Aid, enter an asterisk (*) in the column and row position to display the key value.
   
   The Field Definition screen displays.

10. On Field Definition, enter VC1KEY in the Field Name field.

**Figure 23–4 Field Definition (VC1KEY) screen**

11. Click Enter.
The description for the key value replaces the asterisk (*)

In the sample report that follows, when you print the report, the field contains the value of 5.

12. On Report Design Aid, enter an asterisk (*) in the column and row position to begin the key value description.

The Field Definition screen displays.

13. On Field Definition, enter VC1DSC in the Field Name field:

![Figure 23–5 Field Definition (VC1DSC) screen]

14. Click Enter.

The description for the key value replaces the asterisk (*)

In the sample report that follows, when you print the report, the field contains the value San Francisco.

15. On Report Design Aid, add the field to be accumulated to the report.

The field that contains data for the Quantity on Hand column is RRXQTY. The system places the total amount of Quantity On Hand in field $$XQTY as part of the TOTAL1 format. The program generator creates the total amount field by adding a $$ prefix to the data item name.

The following illustrates the report.

![Figure 23–6 Inventory by Business Unit Report]
23.3 Defining a Subheading

You can define a subheading before you choose the type of detail you want in the report.

Subheading field descriptions are similar to those for totals. You can display up to three pieces of information at each subhead:

- The field description
- The value
- The description of the value of the level break fields

For example, if you choose to add a subheading to your report using business unit as the level break field, the report can read:

Business Unit 5 San Francisco

When adding the field description for the subhead, use field VC0ROW. When adding the value of the subhead, use field VC0KEY. When adding the description of the value of the subhead, use field VC0DSC.

Add these fields on the Field Definition screen in the same manner as the VC1 fields for the TOTAL1 format. On the design area, enter an asterisk (*) where the subheading field should begin. The Field Definition screen opens. Enter the field name and any other appropriate information.

To define a subheading

1. On Software Versions Repository, locate the report for which you want to define subheadings.
2. On Software Versions Repository, enter 10 in the Option field to access the Report Design Aid.
3. On Report Design Aid, choose Record Formats List (F10).
4. On the Record Formats List, enter HEADING2 on the first blank line in the Format Name field:

Figure 23–7 Record Formats List (Define a Subheading) report

5. Complete the following fields:
Understanding DREAM Writer Considerations

- Type
- Start/End Lines
- Fld Pfx (Field Prefix)

The system accommodates the placement of the fields on the report.

The following is an example of a report using a HEADING2 format. The system creates this report using a C0020 program type.

**Figure 23–8  Inventory by Business Unit Report (HEADING2)**

Program type C0025 report prints the subheadings above the column titles as follows. You use the same steps to define this report as you use for the C0020 program type.

**Figure 23–9  Inventory by Business Unit Report (Type C0025)**

The report program adds the grand totals automatically because it utilizes the total format. When you use DREAM Writer and create a version, you define the fields to use as total levels. For further information regarding DREAM Writer, see Understand DREAM Writer in the *JD Edwards World Technical Foundation Guide*.

### 23.4 Understanding DREAM Writer Considerations

When compiling your report, use the PRTF command to print a cover page. PRTS does not print a cover page when the system finishes compiling the report.

When the program generator creates the report program, it includes a cover page. Using the DREAM Writer, you can choose to print the cover page.

The title fields the system included on the cover page are in the following example:

**Figure 23–10  Cover Page With Title Fields in DREAM Writer**

The VC0CO field is the name of the company. The TTL@ field is line 1 of the DREAM Writer version. TXT2 and TXT3 are lines 2 and 3 of the DREAM Writer version.

The field names for report headings are similar to those of the cover page. For report headings, the VC0CO field contains the name of the company. The DREAM Writer fields have an RR prefix. For example, RRTTL@ field contains line 1 of the DREAM Writer version ID description. The RRTXT2 and RRTXT3 are lines 2 and 3 of the DREAM Writer description, respectively.
On the Additional Parameters screen in the DREAM Writer version, you must enter 2 in the Type Report Totaling field. This enables you to specify your total level fields on the Data Sequence screen.
Part VI
Additional Tools

This part contains these chapters:

- Chapter 24, "Overview to Additional Tools,"
- Chapter 25, "Work with Quick Start CL Generator,"
- Chapter 26, "Work with the Quick Start Application Tool,"
- Chapter 27, "Work with Action Diagramming,"
This chapter contains these topics:

- Chapter 24.1, "Objectives,"
- Chapter 24.2, "About Additional Tools."

### 24.1 Objectives

- To use the Quick Start CL Generator
- To use the Quick Start Application Tool
- To use Action Diagramming

### 24.2 About Additional Tools

You can quickly create programs, forms, and reports using:

- Quick Start CL Generator
- Quick Start Application Tool

Produce a diagram to illustrate the different groupings of logic and the interrelationships of code using the Action Diagramming feature.

Perform the following tasks:

- Work with Quick Start CL Generator
- Work with the Quick Start Application Tool
- Work with Action Diagramming
This chapter contains the topic:

- Section 25.1, "Working with Quick Start CL Generator."

### 25.1 Working with Quick Start CL Generator

The Quick Start CL Generator provides a quick and easy way for you to create a Control Language (CL) program for any of the following four types of programs.

- Standard interactive program
- Standard report program with DREAM Writer
- File processor
- File processor with DREAM Writer

Quick Start CL Generator does not add the new CL program to a menu. You must perform that task manually.

Perform the following tasks:

- To create a program using the Quick Start CL Generator
- To compile a CL program

**Navigation**

From Computer Assisted Programming (CAP) (G93), choose Quick Start CL Generator

**To create a program using the Quick Start CL Generator**

1. On Quick Start CL Generator, complete the following fields and click Enter:
   - Description
   - Program Name
   - Screen or Report Name
   - Select a Program Type(1-4)
   - Master File
   - Source File Name
   - Source Library Name
   - Object Library Name

2. Perform one of the following:
■ Click Exit (F3) to return to the menu.
■ Click Enter to compile the program.

Figure 25–1 Quick Start CL Generator screen

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Use this field to enter a short one-line description of the program you are creating.</td>
</tr>
<tr>
<td>Program Name</td>
<td>Type the name of the RPG program that the CL program will call. This is a required field. Do not leave it blank. The name of the CL program generated will be the same as the RPG program name, but prefaced with a J instead of a P.</td>
</tr>
<tr>
<td>Screen or Report Name</td>
<td>Type the screen or report file name associated with the program. This field is only required for program type 2.</td>
</tr>
</tbody>
</table>
| Select a Program Type(1–4)| Type one of the following in this field to indicate the type of program you are creating:  
1 – Standard Interactive Program  
2 – Standard Report Program with DREAM Writer  
3 – File Processor  
4 – File Processor with DREAM Writer |
| Master File               | Type the name of the data file to use for the program you are creating. This field is required for program types 2, 3, and 4. |
| Source File Name          | The member ID of the file used by the program. |
| Source Library Name       | File and library that contains the file source. |
| Object Library Name       | Enter the name of the object library where the program you are creating will reside. This is defaulted from the CASE Profiles. |
To compile a CL program
On Quick Start CL Generator, enter 1 to choose Compile CL Program.

Figure 25–2  Quick Start CL Generator (Compile) screen
This chapter contains these topics:

- Section 26.1, "Quick Start Process,"
- Section 26.2, "Defining the Application,"
- Section 26.3, "Selecting Data Fields,"
- Section 26.4, "Browsing or Updating the Screens or Reports (Optional),"
- Section 26.5, "Compiling the Screens or Report (Optional),"
- Section 26.6, "Modifying Specifications (Optional),"
- Section 26.7, "Submitting the Program to Compile (Optional),"
- Section 26.8, "Updating the Data Dictionary and Glossary."

The Quick Start Application Tool allows you to quickly create initial versions of programs, screens, and reports. After you create a new version, you can access the Screen or Report Design Aid or the Program Generator and make adjustments to the version.

The tool provides an easy way for you to create a prototype. This program allows you to:

- Create the program that you associate with the screen or report.
- Select fields dynamically from the master and detail files, as well as other database files.
- Compile your screen or report.
- Create specifications for the Program Generator.
- Create and compile your source code, optionally.
- Create a Control Language (CL) program to launch your new screen or report program.

The Quick Start Application Tool recognizes whether the program is a subfile. Additionally, this tool:

- Adds a hidden field to the screen for subfile maintenance.
- Sets the Entry Optional field to N for subfile maintenance.

Quick Start cannot:

- Define which VC0 fields to use as defaults in the version.
- Add the CL program to a menu.
26.1 Quick Start Process

The Quick Start Application Tool is a set of steps that allow you to:

1. Define the Application
   - Define the type of program you want to create
   - Define the screen options
   - Define the report options
   - Define the files and libraries
   - Define the source file to use to create the application

2. Select Data Fields
   - Select the individual data fields to display on the screen or report using JD Edwards World Screen/Report Design Aid
   - Sequence the fields any way you choose.

3. Browse or update the screens or report you are creating (optional).

4. Compile screens or the report (optional).

5. Modify Specifications (optional).
   - Using the File Specifications, the Detailed Programming Facility, and the Help Instructions based on the program type you select.
   - Compile the program. Even if you compile the screen or report in a previous step, the system prompts you to do so again.

6. Compile the program (optional).


---

**Note:** As you create a version, you should continue through the steps and complete the entire process. The optional steps allow you to perform additional functions that relate to the process. If you exit the process and access the tool at a later time, the system enters all of the information on the Quick Start Application Tool screen from the previous version you created if you did not sign off the system.

---

**Navigation**

From Computer Assisted Programming (CAP) (G93), choose Quick Start Application Tool

---

26.2 Defining the Application

You create versions of programs, screens, and reports by defining the initial criteria. The system enters all of the information on this screen from the previous version you created if you did not sign off the system.

**To define the application**

On Quick Start Application Tool, complete the following fields:
Defining the Application

- Description
- Program Name
- Create Program (Y/N)?
- Screen or Report Name
- Select a Program Type (1-4)
- Action Code
- Selection Option
- Report Detail Subheadings
- Report Total Subheadings
- Report Totals
- Master File
- Library Name
- Detail File (optional)
- Library Name
- Source File Name
- Source Library Name
- Object Library Name

**Figure 26–1 Quick Start Application Tool screen**

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Use this field to enter a short one-line description of the program you are creating.</td>
</tr>
</tbody>
</table>
Selecting Data Fields

The system displays key fields from each data file. You can select, deselect or resequence the fields that you want to use in your program.

The fields from the primary data file display first in the list, followed by the fields from the secondary data file, if you entered one.

To select data fields

1. On Quick Start Field Selection, perform any of the following:
   - To select a field, enter 1 in the Option field to the left of the field name and click Enter.
   - To specify heading or subfile fields enter 1 or 2, respectively, in the column to the right of the selection and sequencing column.

---

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Name</td>
<td>Type the name of the RPG program that the CL program will call. This is a required field. Do not leave it blank. The name of the CL program generated will be the same as the RPG program name, but prefaced with a J instead of a P.</td>
</tr>
<tr>
<td>Create Program (Y/N)?</td>
<td>Indicate in this field whether you want to create the RPG program.</td>
</tr>
<tr>
<td>Screen or Report Name</td>
<td>Type the screen or report file name associated with the program. This field is only required for program type 2.</td>
</tr>
<tr>
<td>Select a Program Type (1-4)</td>
<td>Type one of the following in this field to indicate the type of program you are creating: 1 – Standard Interactive Program 2 – Standard Report Program with DREAM Writer 3 – File Processor 4 – File Processor with DREAM Writer</td>
</tr>
<tr>
<td>Action Code</td>
<td>Enter Y if you are creating a screen with an Action Code field. Enter N if you are not creating a screen with an Action Code field.</td>
</tr>
<tr>
<td>Selection Option</td>
<td>Enter Y if you are creating a screen with a selection option.</td>
</tr>
<tr>
<td>Report Detail Subheadings</td>
<td>Enter Y if you are creating a report with detail subheadings.</td>
</tr>
<tr>
<td>Report Total Subheadings</td>
<td>Enter Y if you are creating a report with total subheadings.</td>
</tr>
<tr>
<td>Report Totals</td>
<td>Enter Y if you are creating a report with totals.</td>
</tr>
<tr>
<td>Master File</td>
<td>Type the name of the data file to use for the program you are creating. This field is required for program types 2, 3, and 4.</td>
</tr>
<tr>
<td>Library Name</td>
<td>Type in the name of the library your master file is in.</td>
</tr>
<tr>
<td>Detail File (optional)</td>
<td>Type in the name of an optional secondary file from which you want to select data.</td>
</tr>
<tr>
<td>Library Name</td>
<td>Type in the name of the library your secondary file is in.</td>
</tr>
<tr>
<td>Source File Name</td>
<td>The member ID of the file used by the program.</td>
</tr>
<tr>
<td>Source Library Name</td>
<td>File and library that contains the file source.</td>
</tr>
<tr>
<td>Object Library Name</td>
<td>Enter the name of the object library where the program you are creating will reside. This is defaulted from the CASE Profiles.</td>
</tr>
</tbody>
</table>
This field only displays if the program you are creating is a transaction processor.

- To sequence a field, enter the sequence number in the Option field to the left of the fields you want to use in your program and click Enter.
- To add fields, enter the names of additional fields on the screen.

2. Press (F3) to continue.

3. The system prompts you perform one of the following:
   - Exit (F3) to exit the program.
   - Replace (F6) to continue to the next step in the process.

Figure 26–2  Quick Start Field Selection screen

26.4 Browsing or Updating the Screens or Reports (Optional)

At this point in the process, you can view or update the screens or reports you are create using the browse or update mode.

To browse or update screens or reports
On Quick Start Application Tool, enter 1 to browse or 2 to update.

To continue the steps to create a version, enter 8.

To exit the program, enter 9 and return to the Computer Assisted Programming (CAP) menu.
26.5 Compiling the Screens or Report (Optional)

At this point in the process, you can compile the screens or report. Before you compile the program, you must complete this step.

To compile the screens or report
On Quick Start Application Tool, enter 3 to compile the screens or reports.
The system retrieves the object library for the compile from the CASE Profiles.
To continue the steps to create a version, enter 8.
26.6 Modifying Specifications (Optional)

At this point in the process, you can access the Program Generator Specifications screen to modify specifications.

To modify specifications
On Quick Start Application Tool, enter 2 to modify program specifications.
The Program Generator Specifications screen displays.
To continue the steps to create a version, enter 8.
26.7 Submitting the Program to Compile (Optional)

Do not submit the program to compile until the screens or report successfully compile.

To submit the program to compile
On Quick Start Application Tool, enter 1 to compile the screens or reports.
To continue the steps to create a version, enter 8.
26.8 Updating the Data Dictionary and Glossary

You must enter the purpose of the program on the Data Item Glossary Revisions screen.

To update the data dictionary and glossary
1. On Quick Start Application Tool, click Enter to continue.
2. On Data Item Glossary Revisions, enter the description of the program's purpose that displays in the online help instructions.

Figure 26–8 Data Item Glossary Revisions screen

3. Click Add to add the program purpose statement.

4. Click Exit (F3).

5. On Quick Start Application Tool, perform one of the following:
- Click Enter to exit the program and return to the menu.
- Enter one of the following:
  - 1 to Return to the Data Field Pick List
  - 7 to Return to Quick Start Definition

*Figure 26–9 Quick Start Application Tool (Exit) screen*
This chapter contains these topics:

- Section 27.1, "Building an Action Diagram,"
- Section 27.2, "Viewing an Action Diagram,"
- Section 27.3, "Accessing the Logic Translation Feature."

The Action Diagramming functionality allows you to produce a diagram which illustrates the different groupings of logic and the interrelationships of code within a program. The system generates the diagrams from the program source code. They provide easy access to more detailed information about the files, fields and programs in the code.

### 27.1 Building an Action Diagram

The Build Action Diagram program allows you to build the necessary cross reference items to produce the action diagram. Using DREAM Writer as the initial screen to the batch job, you can specify the programs for which you want to build an action diagram.

JD Edwards World includes sample Action Diagrams with the software but you must build the Action Diagram for all other programs. This is not an automatic function.

**To build an action diagram**

**Navigation**

From Action Diagramming (G9363), choose Build Action Diagrams

On Build Action Diagram, choose a version.
27.2 Viewing an Action Diagram

When you view an action diagram, you are viewing a graphical representation of the code's hierarchy within the program and how different subsets of code relate to other subsets of code. You can view the code for a subroutine from the program or exit to facilities that show more detail for fields, files, and programs.

To view an action diagram

**Navigation**

*From Action Diagramming (G9363), choose Display Action Diagram*

On Display Action Diagram, enter a program ID in the Program ID field to view an action diagram.

For example, enter P92801.
The logic groups for the program display.

<table>
<thead>
<tr>
<th>Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lvl/Sbr</td>
<td>Specifies the logic level and subroutine.</td>
</tr>
<tr>
<td>Program ID</td>
<td>The program name for the action diagram.</td>
</tr>
<tr>
<td>Scan</td>
<td>Allows the user to search for specific information.</td>
</tr>
</tbody>
</table>

The use of colors, arrows, indentation, and connecting vertical lines indicates the hierarchy and relationships of the code within the program.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>===&gt;</td>
<td>Signals the beginning or ending of a loop.</td>
</tr>
<tr>
<td>---&gt;</td>
<td>Signals an IF or WHEN statement or their associated end statement.</td>
</tr>
<tr>
<td>Blank</td>
<td>Labels are in reverse image.</td>
</tr>
</tbody>
</table>

### 27.2.1 Function Exits

**Display File Usage (F10)**
Choose Display File Usage (F10) to view the files in the file specifications of the program.

**Return to Previous Logic Level (F12)**
Choose Return to Previous Logic Level (F12) to return to the logic level immediately prior to the one that currently displays.
Scan Text Forward (F16)
Choose Scan Text Forward (F16) to enter a value in the Scan field and then scan forward through the code to locate the value.

Scan Text Backward (F17)
Choose Scan Text Backward (F17) to enter a value in the Scan field and then scan backward through the code to locate the value.

Skip to Start Group (F19)
Choose Skip to Start Group (F19) to skip to the beginning (start) of a section of code. The user places the cursor within the section of code and then chooses Skip to Start Group to go to the beginning of that section of code.

Skip to End Group (F20)
Choose Skip to End Group (F20) to skip to the end of a section of code. The user places the cursor within the section of code and then chooses Skip to End Group to go to the end of that section of code.

Print Action Diagram (F21)
Choose Print Action Diagram (F21) to obtain a printout of the action diagram.

Program Flowchart (F23)
Choose Program Flowchart (F23) to view and print, or view, or print a flowchart which illustrates the interaction of files and processes that relate to a single program. You can continue to view lower levels of detail as well.

27.2.2 Cursor Sensitive Function Exits
To access information that relates to fields, files, and programs appearing in the program code, you can use cursor sensitive function exits to access this information by placing the cursor at the beginning of the field, file, or program.

Software Versions Repository (F13)
Choose Software Versions Repository (F13) to access the Software Versions Repository.

File Field Description (F14)
Choose File Field Description (F14) to display the File Field Description screen.

Data Cross Reference (F15)
Choose Data Cross Reference (F15) to access the cross reference program.

Data Dictionary (F18)
Choose Data Dictionary (F18) to access the Data Dictionary program.

The following chart indicates which function exits access relevant information for the different elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Function Exit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fields</td>
<td>Data Cross Reference (F15)</td>
<td>Displays all the programs that use the data item.</td>
</tr>
</tbody>
</table>

27-4 JD Edwards World Computer Aided Software Engineering Guide
### 27.2.3 Option Field Values

**View (5)**

Allows the user to view subroutine code whenever it indicates that the program is to execute a subroutine.

### 27.3 Accessing the Logic Translation Feature

The Logic Translation feature allows you to view how the Action Diagram feature translates the RPG code of a program into an Action Diagram.

**To access the logic translation feature**

**Navigation**

*From Action Diagramming (G9363), choose Translation Table*

The system displays the RPG operation in the first column and how it translates that operation within an action diagram in the second column.
Figure 27–3  Translation Table screen
Part VII
Source Code Inventory and Database

This part contains these chapters:

- Chapter 28, "Overview to Source Code Inventory and Database,"
- Chapter 29, "Understand Source Sequence,"
- Chapter 30, "Working with Program Types,"
- Chapter 31, "Work with Logic Modules,"
- Chapter 32, "Understand Directives,"
- Chapter 33, "Work with the Question and Answer System,"
- Chapter 34, "Create User Defined PDL."
This chapter contains these topics:

- Section 28.1, "Objectives,"
- Section 28.2, "About the Source Code Inventory and Database."

28.1 Objectives

- To understand the Source Sequence Line Number
- To create or modify program types
- To create or modify logic modules
- To understand directives
- To understand the Question and Answer system
- To create user defined PDL

28.2 About the Source Code Inventory and Database

Perform the following tasks:

- Understand the Source Sequence Line Number
- Create or modify program types
- Create or modify logic modules
- Understand directives
- Work with the Question and Answer system
- Create user defined PDL
This chapter contains these topics:
- Section 29.1, "Source Serial Numbers,"
- Section 29.2, "Source Sequence Line Structure,"
- Section 29.3, "Structure of the Serial Number."

When you use the program generator, it is important that you understand how the system manages the source code in the program. The topics in this chapter include the key elements that the system assigns.

### 29.1 Source Serial Numbers

When the program generator creates a new program, it assigns each line of source code within the program a twelve-digit serial number. If you regenerate a program after making changes, the program generator uses the serial numbers to integrate your changes, and then renumbers the entire source.

### 29.2 Source Sequence Line Structure

The source sequence line structure includes six elements:

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Key</td>
<td>The primary key represents source code lines that come from a Primary Logic Module. The primary key begins in column 80.</td>
</tr>
<tr>
<td>Secondary Key</td>
<td>The secondary key represents the source code lines that come from a Detail Logic module. The secondary key begins in column 90.</td>
</tr>
<tr>
<td>Serial Number</td>
<td>The serial number is a 12-digit number the program generator assigns to each line of source code in a program. The serial number begins in column 100.</td>
</tr>
<tr>
<td>User ID</td>
<td>When the program generator creates a program, it places the User ID of the program’s creator within the source sequence line.</td>
</tr>
<tr>
<td>SAR Number</td>
<td>When the program generator creates a program, it places the SAR Number, if available, within the source sequence line.</td>
</tr>
<tr>
<td>Date Last Change</td>
<td>When the program generator creates a program, it places the date you add or change the code within the source sequence line.</td>
</tr>
</tbody>
</table>
The following illustrates the parts of the source sequence line.

**Figure 29–1  Parts of the Source Sequence Line**

<table>
<thead>
<tr>
<th>Primary Key</th>
<th>Secondary Key</th>
<th>Serial Number</th>
<th>User ID</th>
<th>SAR Number</th>
<th>Date Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700070000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700080000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700090000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700100000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700110000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700120000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
<tr>
<td>S999-4</td>
<td>RR#BEN</td>
<td>00700130000000QUARLES</td>
<td>721561</td>
<td>000000</td>
<td></td>
</tr>
</tbody>
</table>

The Primary and Secondary keys and serial number make up a unique key for each line of code.

### 29.3 Structure of the Serial Number

**Figure 29–2  Structure of the Serial Number**

- **29.3.1 Source Inventory Master File (F93001) - XXXX**
  - Assigns numbers to the first sequence of the serial number.
  - Increments by 10 to allow you to insert lines as the Program Generator Source Inventory Master file changes.
  - Allows a maximum of 9999 lines.

- **29.3.2 Generation Execution - YYYY**
  - Assigns numbers when the system generates the program.
  - Represents lines that are part of a detail logic module.
  - Increments by 10 to allow you to insert lines.
  - Allows a maximum of 9999 lines.

- **29.3.3 User Change in SEU - ZZZZ**
  - Represent lines of code that the user inserts via SEU.
  - Allows a maximum of 9999 lines.
This chapter contains these topics:

- Section 30.1, "Reviewing Abbreviations for Program Types,"
- Section 30.2, "Reviewing Program Types Index,"
- Section 30.3, "Reviewing Program Types Cross Reference,"
- Section 30.4, "Creating or Modifying Program Types."

Within the program generator, the program type specifies the basic function or type of program that you create. For example, there are separate program types for basic interactive maintenance programs, programs which use subfiles, conversion programs, report programs, and batch update programs.

The system ties each program type to the question and answer process with the program generator.

- After answering a series of questions about the program to generate, the system determines the program type and assigns it to your program specifications.
- The program generator constructs the program using primary and detail logic modules within the program type.

JD Edwards World provides you with program types for the most common programs. You can create your own program types for your organization's needs.

### 30.1 Reviewing Abbreviations for Program Types

You can use the Index to review abbreviations for data that displays on the Create Modify screen.

**Navigation**

From Model Program Design Menu (G9361), under PROGRAM TYPES, choose Index
30.2 Reviewing Program Types Index

You can use the Program Type Index program (P93900) to locate program types. You can access this from the Program Types Cross Reference screen, Index screen, as well as the Create/Modify screen.

30.2.1 Available Options

You can right click on any program type and choose one of the following options:

- Display Full Logic - Access the Data Item Glossary Revisions screen
- Print Logic Class Source
  - Prints the generic source of the shell program without any of the specifics (detail logic modules).
  - Use this if you are creating your own program types and you want to review them.
- Display Logic Class Source
  - Displays the generic source of the shell program without any of the specifics (detail logic modules).
  - Use this if you are creating your own program types and you want to review them.
- Return Selected Program Type - Retrieves the program type when the system accesses it from another program.
- Display Logic Cross Reference - Displays the programs you create using this logic type.
- Revise Logic Class - Displays a bill of materials list for the program type.
30.3 Reviewing Program Types Cross Reference

The Program Types Cross Reference screen allows you to review additional information about program types.

Navigation
From Model Program Design Menu (G9361), under PROGRAM TYPES, choose Cross Reference
30.4 Creating or Modifying Program Types

As you create or modify program types, you should be aware of the following:

30.4.1 Program Type

It is the list of the segments of code the system requires to build this type of program. You might consider this a bill of materials list.

30.4.2 Primary Module

This is the main sections of code that the system uses to create the first level of program source.

30.4.3 Glossary K

The system uses these to document logic modules within a program type. When the system generates a program, it validates the field against the Data Dictionary, and adds the glossary for the key as documentation for the logic module.

To create or modify program types

Ensure you are of the alpha order requirement when creating new program types. The following is an example using program type D0040.
Navigation
From Model Program Design Menu (G9361), under PROGRAM TYPES, choose Create/Modify

On Create/Modify, locate an existing program type.

Figure 30–4 Create/Modify screen
This chapter contains these topics:

- Section 31.1, "Primary Logic Modules,"
- Section 31.2, "Detail Logic Modules,"
- Section 31.3, "Generation Options,"
- Section 31.4, "Viewing the Logic Module Index,"
- Section 31.5, "Viewing Logic Module Cross Reference,"
- Section 31.6, "Viewing Logic Module Op Codes,"
- Section 31.7, "Maintaining the Logic Module File,"
- Section 31.8, "Creating or Modifying Logic Modules,"
- Section 31.9, "Creating or Modifying Formula Library Entry,"
- Section 31.10, "Copying or Moving Program Specifications,"
- Section 31.11, "Printing Program Generator Specifications,"
- Section 31.12, "Reviewing Source Modifications,"
- Section 31.13, "Using Program Generator Updates,"
- Section 31.14, "Using CASE Specifications Inquiry,"

There are two types of logic modules:

- Primary
- Detail

### 31.1 Primary Logic Modules

Primary logic modules include:

- Main segments of code in the definition of a program type.
- Full sections of a program or subroutines within the program, normally.
- Functional directives to the generation program.

Each primary logic module includes code with a five character directive code. See Columns 1 through 5 in the Source Code Inventory Master file (F93001).

The JD Edwards World CASE software provides approximately 100 different primary logic modules. This includes many variations on mainline logic, field initialization,
update logic, housekeeping, and so forth. Use the Index for logic modules to become familiar with the various types of primary logic modules.

Primary logic modules contain the following:

- Program identification specifications
- Extension specifications
- Data structures
- Mainline calculations
- Default logic from data dictionary
- Subroutine calculations
- Update subroutine
- Housekeeping subroutine

### 31.2 Detail Logic Modules

Detail logic modules direct the final integration of the database, screen, or report specifications into the primary logic modules that make up the final program type.

Detail logic modules are usually functional or data field-related segments of code. Functional directives reference the detail logic modules which contain substitution directives to the generation program. A prefix of X indicates the system does not use the detail logic module in conjunction with a conditional directive. A prefix of Z indicates the system uses the detail logic module in conjunction with a conditional directive. See Chapter 32, "Understand Directives" for more information about directives.

### 31.3 Generation Options

Following are additional programs you can use on the Model Program Design Menu.

#### 31.3.1 Help Instructions Edit/Build

You use this to access the Software Versions Repository to rebuild the Helps for a single program.

#### 31.3.2 All Help Instructions

You use this to submit a job to regenerate the helps for all programs.

#### 31.3.3 Global Program Regeneration

You use this to regenerate all programs that have a CAP Status of Y.

---

**Caution:** Use caution when you use this program.

---

Working with logic modules includes the following tasks:

- View the Logic Module Index
- View the Logic Module Cross Reference
- View Logic Module Op Codes
31.4 Viewing the Logic Module Index

The system allows multiple logic modules for each subroutine. Depending on the type of program in which you use the subroutine, the same subroutine can appear differently.

You can review the logic modules on the User Defined Codes window.

To view the logic module index

Navigation
From Model Program Design Menu (G9361), under LOGIC MODULES, choose Index

Figure 31–1 User Defined Codes Window screen

31.5 Viewing Logic Module Cross Reference

The Logic Module Cross Reference allows you to determine which program types use a particular logic module.

To view the logic module cross reference

Navigation
From Model Program Design Menu (G9361), under LOGIC MODULES, choose Cross Reference
Enter a primary logic module name.

**Figure 31–2  Cross Reference (View Logic) screen**

![Cross Reference (View Logic) screen](image)

### 31.6 Viewing Logic Module Op Codes

On the Op Codes screen, the:

- Left column lists the PDL op codes.
- Right column displays the x-module that the system launches to generate the source code.

If PDL does not generate source code, the Operation Code to Logic Module X-Ref file (F93108) might have been accidently cleared.

**To view the logic module op codes**

**Navigation**

From Model Program Design Menu (G9361), under LOGIC MODULES, choose Op Codes
31.7 Maintaining the Logic Module File

The following programs do not appear on a menu and you must access these programs manually.

**Caution:** Use extreme caution when using these programs.

31.7.1 Resequence Logic Module

Use this program when you need to add several lines to a logic module and resequence the line numbers. If you add or change lines in a logic module, you must manually change or add the serial numbers for the logic module or run this program. The Resequence Master Source program (93998) launches a program to resequence an existing logic module.

Normally, you create and incorporate a new logic module into a new program type. You use the new program type and delete the old program type when there are no longer programs with that program type with a CAP status of Y.

CALL P93998 PARM (logic module name).

31.7.2 Remove Logic Module

You use this program when you no longer use a logic module and want to reduce the amount of source code in the F93001 file. The Remove Logic Module program (P93999) removes lines from F93001 and launches a program to remove an existing logic module.

You must ensure that there are no programs with a CAP status of Y that use a program type with this logic module.

CALL P93999 PARM (logic module name).
Creating or Modifying Logic Modules

The Create/Modify screen allows you to review only the logic module you want, otherwise all 12,000 lines of code display because the F93001 is a single member file.

You can choose Field Sensitive Help to access a list of logic modules, UDC 93/LM.

When the system accesses the code, it performs three steps:

- Creates a work file in QTEMP/F93001WRK.
- Adds a member to F93001WRK.
- Clears the member in F93001WRK.

You can exit the code without saving your changes.

To create or modify logic modules

Navigation
From Model Program Design Menu (G9361), under LOGIC MODULES, choose Create/Modify

1. Enter a logic module name.

2. Create or change the appropriate lines of code
Creating or Modifying Formula Library Entry

When you choose this menu selection, the system preloads the screen with the keys for entering a formula.

Alternatively, you also access this screen through the Detailed Programming Facility to enter PDL.

To create or modify the formula library entry

Navigation
From Model Program Design Menu (G9361), under LOGIC MODULES, choose Formula Library Entry
31.10 Copying or Moving Program Specifications

The Parameter Copy/Move program allows you to copy the following from one library to another or from one program ID to another:

- Program Generator specifications
- Data dictionary glossary (program purpose)
- DREAM Writer processing options

Alternatively, you can enter 3 in the Option field on Software Versions Repository to copy Program Generator specifications within a library.

To copy or move program specifications

Navigation

From Model Program Design Menu (G9361), under OTHER TOOLS, choose Parameter Copy/Move

On Parameter Copy/Move complete the following fields:

- Program Generator Specs
- Data Dictionary
- Processing Option
- From Program ID
- To Program ID
31.11 Printing Program Generator Specifications

Use the Print Program Specifications program to print the program specifications. You must use a logical file.

If the print job ends abnormally, review the Additional Parameters screen of the DREAM Writer and ensure that the File Output Type field is set to a Logical File and not Open Query.

To print program generator specifications

**Navigation**

From Model Program Design Menu (G9361), under OTHER TOOLS, choose Print Program Specifications

Copy the appropriate version and change it to print the version of the specifications you want.
31.12 Reviewing Source Modifications

The Review Source Modifications program displays the source code that a user adds manually through the Source Entry Utility. You view the Pxxxxx member in the Additional Help/Modifications Master file (F93002).

The lines of code are the result of the MPxxxxx job that runs and compares the before image of the source code with the source code after the user makes changes and stores the code in the Pxxxxx member in the F93002.

To review source modifications

Navigation

From Model Program Design Menu (G9361), under OTHER TOOLS, choose Review Source Modifications

Alternatively, you can enter 30 in the Option field on Software Versions Repository to access Review Source Modifications screen.

1. On the Review Source Modifications screen, locate the program.
2. Enter 30 in the Option field to view source code modifications.
31.13 Using Program Generator Updates

The both of the Generator Updates merge JD Edwards World updates for the Program Generator.

The system uses these programs during a PTF install.

To use program generator updates

Navigation
From Model Program Design Menu (G9361), choose Generator Updates From Generator Updates (G9366), choose the appropriate Compare/Update

31.14 Using CASE Specifications Inquiry

The CASE Specifications Inquiry allows you to view the programs you design using the JD Edwards World CASE Tools. You can modify and delete CASE Specifications using this utility as well as access the source code in the Software Versions Repository.

To use CASE specifications inquiry

Navigation
From Model Program Design Menu (G9361), under OTHER TOOLS, choose Case Specifications Inquiry

1. On CASE Specifications Inquiry, complete any of the following fields:
   - Program ID
   - System Code
   - CAP Status
- **Program Type**
  The system displays the records that meet your search criteria.

2. Complete the following field:
   - **Option**

*Figure 31–10 CASE Specifications Inquiry screen*
This chapter contains these topics:

- **Section 32.1, "Functional Directives,"
- **Section 32.2, "Substitution Directives,"
- **Section 32.3, "Exception Directives,"
- **Section 32.4, "Conditional Directives."

Directives are in the logic modules and instruct the program generator what type of action to take when constructing source code. They use the first five columns of the RPG statement.

There are several types of directives, including:

- Functional Directives
- Substitution Directives
- Exception Directives
- Conditional Directives

JD Edwards World supplies all directives and you cannot create your own directives.

### 32.1 Functional Directives

Functional directives:

- Control major functions within a program.
- Provide the initiation point for creating database specific logic and screen or report file control logic.
- Initiate the inclusion of copy modules into the source code.
- Seize detail logic modules for inclusion.

Functional directives are only found within primary logic modules and cannot reside in a detail logic module.

Following are the functional directives JD Edwards World includes with the software:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*ACTN</td>
<td>None</td>
<td>S999</td>
<td>Load action code lock out array</td>
</tr>
<tr>
<td>*ATOT</td>
<td>XADDTOT1</td>
<td>S010</td>
<td>Accumulate report total logic</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>AUTHR</td>
<td>None</td>
<td>F spec</td>
<td>Program author</td>
</tr>
<tr>
<td>*AUTOI</td>
<td>X*ENTRYI</td>
<td>S999</td>
<td>Automatic inquiry at execution test logic</td>
</tr>
<tr>
<td>CLRN</td>
<td>None</td>
<td>S001</td>
<td>Clear user requested fields</td>
</tr>
<tr>
<td>CLRY</td>
<td>None</td>
<td>S001</td>
<td>Clear all data fields for next transaction</td>
</tr>
<tr>
<td>CLSYL</td>
<td>None</td>
<td>S001</td>
<td>Clear all subfile fields</td>
</tr>
<tr>
<td>COPY</td>
<td>XCOPY-SUB</td>
<td>Various</td>
<td>RPGIII copy function for common subroutines</td>
</tr>
<tr>
<td>CTOT</td>
<td>XCLRTOT1</td>
<td>S010</td>
<td>Clear report totals</td>
</tr>
<tr>
<td>*DATES</td>
<td>XDSDATE</td>
<td>I Spec</td>
<td>Data structures for Gregorian dates (not using record buffer)</td>
</tr>
<tr>
<td>*DATER</td>
<td>None</td>
<td>I Spec</td>
<td>Data structures for Gregorian dates in the record buffer #BUFIN</td>
</tr>
<tr>
<td>DESC</td>
<td>None</td>
<td>F Spec</td>
<td>File or program description</td>
</tr>
<tr>
<td>*DPARM</td>
<td>XFIELDVAL</td>
<td>S998</td>
<td>Retrieve all Data Dictionary values for videos</td>
</tr>
<tr>
<td>*DPRMS</td>
<td>XFIELDVL2</td>
<td>S998</td>
<td>Retrieve Data Dictionary values for detail subheading reports</td>
</tr>
<tr>
<td>*DPRMR</td>
<td>XFIELDVAL</td>
<td>S998</td>
<td>Retrieve Data Dictionary values for total subheading reports</td>
</tr>
<tr>
<td>DSPF</td>
<td>None</td>
<td>Various</td>
<td>Variable name substitution for display file(s) fields</td>
</tr>
<tr>
<td>DSP1</td>
<td>S</td>
<td>004</td>
<td>Display logic for primary video fields</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD1</td>
<td>S004</td>
<td>Format Alpha field for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD2</td>
<td>S004</td>
<td>Format Gregorian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD3</td>
<td>S004</td>
<td>Format Julian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD4</td>
<td>S004</td>
<td>Format VC0 field from VTX</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD5</td>
<td>S004</td>
<td>Format VC0 field from designated description file (field details)</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD6</td>
<td>S004</td>
<td>Format VC0 field from F0005</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD7</td>
<td>S004</td>
<td>Format Alpha 3 or 28</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD8</td>
<td>S004</td>
<td>Repeat of XDSPFLD1</td>
</tr>
<tr>
<td>DSP2</td>
<td></td>
<td></td>
<td>Display logic for primary video fields</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD1</td>
<td>S004</td>
<td>Format Alpha field for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD2</td>
<td>S004</td>
<td>Format Gregorian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD3</td>
<td>S004</td>
<td>Format Julian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD4</td>
<td>S004</td>
<td>Format VC0 field from VTX</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD5</td>
<td>S004</td>
<td>Format VC0 field from designated description file (field details)</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD6</td>
<td>S004</td>
<td>Format VC0 field from F0005</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD7</td>
<td>S004</td>
<td>Format Alpha 3 or 28</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD8</td>
<td>S004</td>
<td>Repeat of XDSPFLD1</td>
</tr>
<tr>
<td>*EMK</td>
<td>XLOADEMK</td>
<td>S999</td>
<td>Load user defined error messages</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>ENTRY</td>
<td>X*ENTRYYP</td>
<td>Various</td>
<td>Load program execution passed parameters</td>
</tr>
<tr>
<td></td>
<td>X*ENTRYYM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*EXITC</td>
<td>XEXIT-CMD0</td>
<td>S00EX</td>
<td>Function key exit execution logic</td>
</tr>
<tr>
<td></td>
<td>XEXIT-CMD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*EXITS</td>
<td>XEXIT-SEL0</td>
<td>S00OP</td>
<td>Selection exit execution logic</td>
</tr>
<tr>
<td>*EXITW</td>
<td>XEXIT-SEL0</td>
<td>S00P</td>
<td>Selection exit execution logic</td>
</tr>
<tr>
<td>*FIELD</td>
<td>XFIELDEDT1</td>
<td>S005</td>
<td>Active Data Dictionary field validation logic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Data Dictionary alpha edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT2</td>
<td>S005</td>
<td>Validation n=Master - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT3</td>
<td>S005</td>
<td>Gregorian edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT4</td>
<td>S005</td>
<td>Julian edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT5</td>
<td>S005</td>
<td>Data Dictionary numeric edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT6</td>
<td>S005</td>
<td>Alpha field size 10</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT7</td>
<td>S005</td>
<td>User defined code edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT8</td>
<td>S005</td>
<td>No dictionary</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT9</td>
<td>S005</td>
<td>Validation n = Master - Numeric</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTA</td>
<td>S005</td>
<td>Account ID</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTC</td>
<td>S005</td>
<td>Cost center edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTE</td>
<td>S005</td>
<td>Numeric field size 7</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTR</td>
<td>S005</td>
<td>Right adjust</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTS</td>
<td>S005</td>
<td>Validation = Master - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTT</td>
<td>S005</td>
<td>Validation = Master - Alpha Rt Adj</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTU</td>
<td>S005</td>
<td>Validation = Master - Numeric</td>
</tr>
<tr>
<td>FILES</td>
<td>None</td>
<td>F spec</td>
<td>Program file descriptions</td>
</tr>
<tr>
<td>*FLDxx</td>
<td>XFIELDEDT1</td>
<td>S005</td>
<td>Active Data Dictionary field validation for primary data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Data Dictionary alpha edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT2</td>
<td>S005</td>
<td>Validation n=Master - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT3</td>
<td>S005</td>
<td>Gregorian edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT4</td>
<td>S005</td>
<td>Julian edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT5</td>
<td>S005</td>
<td>Data Dictionary numeric edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT6</td>
<td>S005</td>
<td>Alpha field size 10</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT7</td>
<td>S005</td>
<td>User defined code edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT8</td>
<td>S005</td>
<td>No dictionary</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDT9</td>
<td>S005</td>
<td>Validation n = Master - Numeric</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTA</td>
<td>S005</td>
<td>Account ID</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTC</td>
<td>S005</td>
<td>Cost center edit</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTE</td>
<td>S005</td>
<td>Numeric field size 7</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTR</td>
<td>S005</td>
<td>Right adjust</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTS</td>
<td>S005</td>
<td>Validation = Master - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTT</td>
<td>S005</td>
<td>Validation = Master - Alpha Rt Adj</td>
</tr>
<tr>
<td></td>
<td>XFIELDEDTU</td>
<td>S005</td>
<td>Validation = Master - Numeric</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>INFDS</td>
<td>XINFDS1</td>
<td>Ispec</td>
<td>File information data structures, if specified</td>
</tr>
<tr>
<td></td>
<td>XINFDS2</td>
<td>Ispec</td>
<td>OBSOLETE. Use SRVFDS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Standard database file information data structure. The field prefix is incremented from $1 to $x where x = number of files</td>
</tr>
<tr>
<td>KEY1</td>
<td>XFIELDLD1</td>
<td>S003</td>
<td>Load master file key fields for inquiry programs.</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD2</td>
<td>S003</td>
<td>Load video input - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD3</td>
<td>S003</td>
<td>Load video input - Numeric</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD4</td>
<td>S003</td>
<td>Load video input - Cost Center</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD5</td>
<td>S003</td>
<td>Load video input - Julian Date</td>
</tr>
<tr>
<td></td>
<td>XNEXT-NBR</td>
<td>S003</td>
<td>Load video input - Gregorian Date</td>
</tr>
<tr>
<td>KEYS</td>
<td>XFIELDLD1</td>
<td>S003</td>
<td>Load master file key fields in subfile format.</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD2</td>
<td>S003</td>
<td>Load video input - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD3</td>
<td>S003</td>
<td>Load video input - Numeric</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD4</td>
<td>S003</td>
<td>Load video input - Cost Center</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD5</td>
<td>S003</td>
<td>Load video input - Julian Date</td>
</tr>
<tr>
<td></td>
<td>XNEXT-NBR</td>
<td>S003</td>
<td>Load video input - Gregorian Date</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Load video input - Next Numbering</td>
</tr>
<tr>
<td>KEYS2</td>
<td>XFIELDLD1</td>
<td>S005</td>
<td>Load master file key fields in primary video format</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD2</td>
<td>S005</td>
<td>Load video input - Alpha</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD3</td>
<td>S005</td>
<td>Load video input - Numeric</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD4</td>
<td>S005</td>
<td>Load video input - Cost Center</td>
</tr>
<tr>
<td></td>
<td>XFIELDLD5</td>
<td>S005</td>
<td>Load video input - Julian Date</td>
</tr>
<tr>
<td></td>
<td>XNEXT-NBR</td>
<td>S005</td>
<td>Load video input - Gregorian Date</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Load video input - Next Numbering</td>
</tr>
<tr>
<td>KLIST</td>
<td>XKEYLIST</td>
<td>S999</td>
<td>Create data file key list</td>
</tr>
<tr>
<td>*LVLS</td>
<td>XSAVVAL1</td>
<td></td>
<td>Save report level break data</td>
</tr>
<tr>
<td>MF</td>
<td>None</td>
<td>Various</td>
<td>Variable name substitution for master database files</td>
</tr>
<tr>
<td>*MCUxx</td>
<td>None</td>
<td>S003</td>
<td>Business Unit security logic where xx = master field designation 1 thru 9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S004</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>S00EX</td>
<td></td>
</tr>
<tr>
<td>*OPEN</td>
<td>XFILEOPN1</td>
<td>S999</td>
<td>Open report program data files</td>
</tr>
<tr>
<td>OPTE</td>
<td>None</td>
<td>S005</td>
<td>Subfile processing condition test based on mandatory entry fields in subfile format</td>
</tr>
<tr>
<td>*OTOT</td>
<td>XPRTTOT1</td>
<td>S010</td>
<td>Print all report level totals</td>
</tr>
<tr>
<td>PDL</td>
<td>None</td>
<td>Various</td>
<td>User defined entry point</td>
</tr>
<tr>
<td>*RKYxx</td>
<td>None</td>
<td>S999</td>
<td>Load softcoding record key for reports where xx = master file designation 1 thru 9</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>----------------------------------------------------------</td>
</tr>
<tr>
<td>RPTD</td>
<td>XDSPFLD1</td>
<td>S004</td>
<td>Format data for report detail format</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD2</td>
<td>S004</td>
<td>Format Alpha field for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD3</td>
<td>S004</td>
<td>Format Gregorian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD4</td>
<td>S004</td>
<td>Format Julian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD5</td>
<td>S004</td>
<td>Format VC0 field from VTX</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD6</td>
<td>S004</td>
<td>Format VC0 field from description file (field details)</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD7</td>
<td>S004</td>
<td>Format VC0 field from F0005</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD8</td>
<td>S004</td>
<td>Repeat of XDSPFLD1</td>
</tr>
<tr>
<td>RPTH</td>
<td>XDSPFLD1</td>
<td>S004</td>
<td>Format data for report heading format</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD2</td>
<td>S004</td>
<td>Format Alpha field for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD3</td>
<td>S004</td>
<td>Format Gregorian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD4</td>
<td>S004</td>
<td>Format Julian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD5</td>
<td>S004</td>
<td>Format VC0 field from VTX</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD6</td>
<td>S004</td>
<td>Format VC0 field from description file (field details)</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD7</td>
<td>S004</td>
<td>Format VC0 field from F0005</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD8</td>
<td>S004</td>
<td>Repeat of XDSPFLD1</td>
</tr>
<tr>
<td>RPTT</td>
<td>XDSPFLD1</td>
<td>S004</td>
<td>Format data for report total format</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD2</td>
<td>S004</td>
<td>Format Alpha field for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD3</td>
<td>S004</td>
<td>Format Gregorian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD4</td>
<td>S004</td>
<td>Format Julian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD5</td>
<td>S004</td>
<td>Format VC0 field from VTX</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD6</td>
<td>S004</td>
<td>Format VC0 field from description file (field details)</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD7</td>
<td>S004</td>
<td>Format VC0 field from F0005</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD8</td>
<td>S004</td>
<td>Repeat of XDSPFLD1</td>
</tr>
<tr>
<td>*RPTT</td>
<td>XDSPFLD1</td>
<td>S004</td>
<td>Format data for report total format</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD2</td>
<td>S004</td>
<td>Format Alpha field for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD3</td>
<td>S004</td>
<td>Format Gregorian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD4</td>
<td>S004</td>
<td>Format Julian Date for output</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD5</td>
<td>S004</td>
<td>Format VC0 field from VTX</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD6</td>
<td>S004</td>
<td>Format VC0 field from description file (field details)</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD7</td>
<td>S004</td>
<td>Format VC0 field from F0005</td>
</tr>
<tr>
<td></td>
<td>XDSPFLD8</td>
<td>S004</td>
<td>Repeat of XDSPFLD1</td>
</tr>
<tr>
<td>*RTA</td>
<td>XTOTARRY</td>
<td>E spec</td>
<td>Load totaling arrays</td>
</tr>
<tr>
<td>*RTS</td>
<td>None</td>
<td>I spec</td>
<td>Report softcoding array</td>
</tr>
<tr>
<td>*RTX</td>
<td>None</td>
<td>I spec</td>
<td>Report softcoding text fields</td>
</tr>
<tr>
<td>*RTXI</td>
<td>XVTIDX</td>
<td>S999</td>
<td>Set maximum VTX index to use</td>
</tr>
</tbody>
</table>
### Active Data Dictionary data field validation for subfile fields.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*SFFLD</td>
<td></td>
<td></td>
<td>Active Data Dictionary data field validation for subfile fields.</td>
</tr>
<tr>
<td>XFIELDDEDT1</td>
<td>S005</td>
<td></td>
<td>Data Dictionary alpha edit</td>
</tr>
<tr>
<td>XFIELDDEDT2</td>
<td>S005</td>
<td></td>
<td>Validation n = Master - Alpha</td>
</tr>
<tr>
<td>XFIELDDEDT3</td>
<td>S005</td>
<td></td>
<td>Gregorian edit</td>
</tr>
<tr>
<td>XFIELDDEDT4</td>
<td>S005</td>
<td></td>
<td>Julian edit</td>
</tr>
<tr>
<td>XFIELDDEDT5</td>
<td>S005</td>
<td></td>
<td>Data Dictionary numeric edit</td>
</tr>
<tr>
<td>XFIELDDEDT6</td>
<td>S005</td>
<td></td>
<td>Alpha field size 10</td>
</tr>
<tr>
<td>XFIELDDEDT7</td>
<td>S005</td>
<td></td>
<td>User defined code edit</td>
</tr>
<tr>
<td>XFIELDDEDT8</td>
<td>S005</td>
<td></td>
<td>No dictionary</td>
</tr>
<tr>
<td>XFIELDDEDT9</td>
<td>S005</td>
<td></td>
<td>Validation n = Master - Numeric</td>
</tr>
<tr>
<td>XFIELDDEDTA</td>
<td>S005</td>
<td></td>
<td>Account ID</td>
</tr>
<tr>
<td>XFIELDDETC</td>
<td>S005</td>
<td></td>
<td>Cost center edit</td>
</tr>
<tr>
<td>XFIELDDETE</td>
<td>S005</td>
<td></td>
<td>Numeric field size 7</td>
</tr>
<tr>
<td>XFIELDDETR</td>
<td>S005</td>
<td></td>
<td>Right adjust</td>
</tr>
<tr>
<td>XFIELDDETS</td>
<td>S005</td>
<td></td>
<td>Validation = Master - Alpha</td>
</tr>
<tr>
<td>XFIELDDETT</td>
<td>S005</td>
<td></td>
<td>Validation = Master - Alpha Rt Adj</td>
</tr>
<tr>
<td>XFIELDDETTU</td>
<td>S005</td>
<td></td>
<td>Validation = Master- Numeric</td>
</tr>
</tbody>
</table>

### Active Data Dictionary data field validation for subfile data fields. Where xx = specified master file 1 thru 9.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SLDxx</td>
<td></td>
<td></td>
<td>Active Data Dictionary data field validation for subfile data fields. Where xx = specified master file 1 thru 9.</td>
</tr>
<tr>
<td>XFIELDDEDT1</td>
<td>S005</td>
<td></td>
<td>Data Dictionary alpha edit</td>
</tr>
<tr>
<td>XFIELDDEDT2</td>
<td>S005</td>
<td></td>
<td>Validation n = Master - Alpha</td>
</tr>
<tr>
<td>XFIELDDEDT3</td>
<td>S005</td>
<td></td>
<td>Gregorian edit</td>
</tr>
<tr>
<td>XFIELDDEDT4</td>
<td>S005</td>
<td></td>
<td>Julian edit</td>
</tr>
<tr>
<td>XFIELDDEDT5</td>
<td>S005</td>
<td></td>
<td>Data Dictionary numeric edit</td>
</tr>
<tr>
<td>XFIELDDEDT6</td>
<td>S005</td>
<td></td>
<td>Alpha field size 10</td>
</tr>
<tr>
<td>XFIELDDEDT7</td>
<td>S005</td>
<td></td>
<td>User defined code edit</td>
</tr>
<tr>
<td>XFIELDDEDT8</td>
<td>S005</td>
<td></td>
<td>No dictionary</td>
</tr>
<tr>
<td>XFIELDDEDT9</td>
<td>S005</td>
<td></td>
<td>Validation n = Master - Numeric</td>
</tr>
<tr>
<td>XFIELDDEDTA</td>
<td>S005</td>
<td></td>
<td>Account ID</td>
</tr>
<tr>
<td>XFIELDDETC</td>
<td>S005</td>
<td></td>
<td>Cost center edit</td>
</tr>
<tr>
<td>XFIELDDETC</td>
<td>S005</td>
<td></td>
<td>Numeric field size 7</td>
</tr>
<tr>
<td>XFIELDDETR</td>
<td>S005</td>
<td></td>
<td>Right adjust</td>
</tr>
<tr>
<td>XFIELDDETS</td>
<td>S005</td>
<td></td>
<td>Validation = Master - Alpha</td>
</tr>
<tr>
<td>XFIELDDETT</td>
<td>S005</td>
<td></td>
<td>Validation = Master - Alpha Rt Adj</td>
</tr>
<tr>
<td>XFIELDDETTU</td>
<td>S005</td>
<td></td>
<td>Validation = Master- Numeric</td>
</tr>
</tbody>
</table>

### Cursor Control, F1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*S00VL</td>
<td>None</td>
<td>I spec</td>
<td>Cursor Control, F1</td>
</tr>
</tbody>
</table>

### Program title

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>None</td>
<td>H spec</td>
<td>Program title</td>
</tr>
</tbody>
</table>

### Load softcoding record key for display files where xx=display file designation 1 - 9.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*VKYxx</td>
<td>None</td>
<td>S999</td>
<td>Load softcoding record key for display files where xx=display file designation 1 - 9.</td>
</tr>
</tbody>
</table>

### Display file softcoding array

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*VTS</td>
<td>None</td>
<td>I spec</td>
<td>Display file softcoding array</td>
</tr>
</tbody>
</table>
Substitution Directives

Understand Directives

32-7

* Automatically include JD Edwards World standards, which are beyond normal requirements.

32.2 Substitution Directives

Substitution directives:

- Control the translation of symbolic names to the actual data field names the system requires for an individual line of source code.
- Substitute information within a line of code.
  - If the system replaces a field, the field it replaces begins with an & (ampersand).
  - If the substitution is positional, this directive informs the program generator where to position the substitution on a line of code.

Figure 32–1 Substitution Directives, Columns 1 to 5

Following are the substitution directives JD Edwards World includes with the software:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*VTX</td>
<td>None</td>
<td>I spec</td>
<td>Update softcoding text field ending positions based upon size definition in display file</td>
</tr>
<tr>
<td>*VTXI</td>
<td>XVTXIDX</td>
<td>S999</td>
<td>Set maximum VTX index used</td>
</tr>
</tbody>
</table>

*VTX None I spec Update softcoding text field ending positions based upon size definition in display file

*VTXI XVTXIDX S999 Set maximum VTX index used

* Automatically include JD Edwards World standards, which are beyond normal requirements.

32.2 Substitution Directives

Substitution directives:

- Control the translation of symbolic names to the actual data field names the system requires for an individual line of source code.
- Substitute information within a line of code.
  - If the system replaces a field, the field it replaces begins with an & (ampersand).
  - If the substitution is positional, this directive informs the program generator where to position the substitution on a line of code.

Figure 32–1 Substitution Directives, Columns 1 to 5

Following are the substitution directives JD Edwards World includes with the software:

<table>
<thead>
<tr>
<th>Directive</th>
<th>Column Allowed</th>
<th>Column Allowed</th>
<th>Column Allowed</th>
<th>Column Allowed</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>@</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Four character Data Dictionary name</td>
</tr>
<tr>
<td>#</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>Primary parameter that passes for *ENTRY</td>
</tr>
<tr>
<td>A</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Highest VTX field.</td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unused at this time.</td>
</tr>
<tr>
<td>Directive</td>
<td>Column Allowed</td>
<td>Column Allowed</td>
<td>Column Allowed</td>
<td>Column Allowed</td>
<td>Function</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------</td>
</tr>
<tr>
<td>C</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>Function key exit indicator test</td>
</tr>
<tr>
<td>D</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td>Descriptions for fields, files, and copy modules</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td>Error message key</td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Validation file name</td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>User defined calculation logic result field name</td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Descriptive display file name</td>
</tr>
<tr>
<td>I</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Display field error condition attribute indicator</td>
</tr>
<tr>
<td>J</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Data file names</td>
</tr>
<tr>
<td>K</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>Descriptive display file key field name</td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Data file key list name and optional file/format name</td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>File information data structure name</td>
</tr>
<tr>
<td>N</td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
<td>Full data field name (write to)</td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Common subroutine name</td>
</tr>
<tr>
<td>P</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Function key/selection exit program to execute</td>
</tr>
<tr>
<td>Q</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Field name to receive description value</td>
</tr>
<tr>
<td>Directive</td>
<td>Column Allowed</td>
<td>Column Allowed</td>
<td>Column Allowed</td>
<td>Column Allowed</td>
<td>Function</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------</td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Field name to receive key value</td>
</tr>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Selection exit value test</td>
</tr>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Function key/selection exit</td>
</tr>
<tr>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>File information data structure subfield prefix</td>
</tr>
<tr>
<td>V</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>Source of data (Read From) field name</td>
</tr>
<tr>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Data file key list key field name</td>
</tr>
<tr>
<td>X</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>Error message array index</td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Function key/selection exit parameter field name</td>
</tr>
<tr>
<td>Z</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Numeric field size definition (right adj alpha)</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Gregorian date Data Structure numeric 6 byte date</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Gregorian date Data Structure numeric 2 byte month</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Gregorian date Data Structure numeric 2 byte day</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>Gregorian date Data Structure numeric 2 byte year</td>
</tr>
<tr>
<td>4</td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
<td>Parameter 1 from *PROC calculations</td>
</tr>
</tbody>
</table>
32.3 Exception Directives

Exception directives:

- Provide unusual option definition to the program generation process.
- Combine two other types of directives. For example:
  
  DSPF &01FILE
  
  Combines a functional directive (DSPF) with a substitution directive (&01FILE), so it is an exception directive.

Example:

You create a line of code for the READ Master file and then substitute the Master file name.

Most exception directives are substitution directives but are out of the normal syntax substitution directives use.

Following are the exception directives JD Edwards World includes with the software:

<table>
<thead>
<tr>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Result</th>
<th>Keyword</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td></td>
<td>&amp;xxFILE</td>
<td>Master/video/report file name</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td></td>
<td>&amp;xx(FILE)</td>
<td>File name in single quote marks</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td></td>
<td>&amp;xxFORMAT</td>
<td>Master/video format name</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td></td>
<td>&amp;xxFORMAT1</td>
<td>Subfile line 24 format name</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td></td>
<td>&amp;xxFORMATC</td>
<td>Subfile control record format name</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td></td>
<td>&amp;xxFORMATS</td>
<td>Subfile record format name</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td></td>
<td>&amp;xxKEYFLD</td>
<td>Master file primary key field name</td>
</tr>
</tbody>
</table>
Conditional Directives

Conditional directives:

- Are the most flexible and most powerful directives.
- Verify specific conditions exist before determining if the system must perform any action.
- Use positions 1 to 5 to provide directive initiation and use Factor 1, Factor 2, and the Result field to complete the directive definition.

Subroutine S010-11 includes examples of conditional directives. For example:

If SFSELC exists, include code for selection exits.

You can combine conditional directives.

Following are the conditional directives JD Edwards World includes with the software:

<table>
<thead>
<tr>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Result</th>
<th>Keyword</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>&amp;xPGCTL</td>
<td></td>
<td>Number of subfile records in 1 page</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>%</td>
<td></td>
<td>Factor 1 intentionally left blank</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>=</td>
<td>User defined calculation logic result</td>
</tr>
</tbody>
</table>

32.4 Conditional Directives

Conditional directives:

- Are the most flexible and most powerful directives.
- Verify specific conditions exist before determining if the system must perform any action.
- Use positions 1 to 5 to provide directive initiation and use Factor 1, Factor 2, and the Result field to complete the directive definition.

Subroutine S010-11 includes examples of conditional directives. For example:

If SFSELC exists, include code for selection exits.

You can combine conditional directives.

Following are the conditional directives JD Edwards World includes with the software:

<table>
<thead>
<tr>
<th>Position/Factor/Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pos 1</td>
<td>+ Include detail logic module if true</td>
</tr>
<tr>
<td></td>
<td>- Include detail logic module if false</td>
</tr>
<tr>
<td>Pos 2-5</td>
<td>FLDN Test existence of data field</td>
</tr>
<tr>
<td></td>
<td>DTAI Test existence of data item</td>
</tr>
<tr>
<td></td>
<td>FILE Test existence of file</td>
</tr>
<tr>
<td></td>
<td>FMT Test existence of file</td>
</tr>
<tr>
<td>Factor 1</td>
<td>Name of field, item, file or format to test. Can also contain *ANYx for file test which you can use to test for types of files in a program where x might optionally designate number of files.</td>
</tr>
<tr>
<td>Oper (file test only)</td>
<td>DSPF Display file</td>
</tr>
<tr>
<td></td>
<td>PF Physical file only</td>
</tr>
<tr>
<td></td>
<td>LF Logical file only</td>
</tr>
<tr>
<td></td>
<td>PRTF Printer file only</td>
</tr>
<tr>
<td></td>
<td>DB Database file</td>
</tr>
<tr>
<td>Factor 2</td>
<td>Name of detail logic module to include into source code. Might also use *AND to produce compound conditions</td>
</tr>
<tr>
<td>Result Field Pos 1</td>
<td>@ Any input file</td>
</tr>
<tr>
<td></td>
<td>M Master input file with M in file specifications</td>
</tr>
<tr>
<td></td>
<td>1-9 Master input file with 1 - 9 in field specifications</td>
</tr>
<tr>
<td>Position/Factor/Result</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Result Field Pos 2</td>
<td>@ Any output file</td>
</tr>
<tr>
<td>Result Field Pos 3</td>
<td>@ Any update file</td>
</tr>
<tr>
<td></td>
<td>M Master update file with M in file specifications</td>
</tr>
<tr>
<td></td>
<td>1-9 Master update file with 1 - 9 in field specifications</td>
</tr>
<tr>
<td>Result Field Pos 4</td>
<td>@ Any add file</td>
</tr>
</tbody>
</table>
This chapter contains these topics:

- Section 33.1, "About Simple Question & Answer,"
- Section 33.2, "Reviewing Questions in a Master Dialogue,"
- Section 33.3, "Adding New Q & A Dialogue,"
- Section 33.4, "Working with an Existing Dialogue."

The program generator uses JD Edwards World Question and Answer system as a method of determining the appropriate program type. Based on the answers to certain questions, the system selects a program type for you.

You can create your own questions and answers to produce your own custom program type. You can also modify the questions, known as a dialogue, the program generator uses through this feature.

### Navigation
From Model Program Design Menu (G9361), choose Maintain Q/A
From World CASE Q & A Menu (G9364), choose Simple Question and Answer

### 33.1 About Simple Question & Answer
The Simple Question & Answer screen, from which you begin all Question & Answer tasks, includes the following three fields.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Question and Answer</td>
<td>Access the Question Entry screen.</td>
</tr>
<tr>
<td>Add New Q &amp; A Dialogue</td>
<td>Access the Dialogue Descriptions screen.</td>
</tr>
<tr>
<td>Update Existing Q &amp; A Dialogue</td>
<td>Access the Dialogue Lists screen.</td>
</tr>
</tbody>
</table>

### 33.2 Reviewing Questions in a Master Dialogue
You choose Display Next Question (F19) and Display Previous Question (F20) to review all of the master questions. Additionally, you can make changes to the master question that displays on the Question Entry screen.

**To review questions in a master dialogue**
1. On Simple Question and Answer, enter Y in the following field:
2. On Question Entry, complete the following field:
   - Question Number
   The question detail displays.

3. To review the answers to the master question Click Change.
33.3 Adding New Q & A Dialogue

You can create your own questions and answers by completing the Add New Q & A Dialogue field.

The dialogue the CASE tool uses to determine the program type is Primary Key
*Default, Data Item LC.

To add new Q&A dialogue
1. On Simple Question & Answer, enter Y in the following field:
   - Add New Q & A Dialogue
   The Dialogue Descriptions screen displays.
2. Complete the following fields:
   - Dialogue Keys: Primary
   - Dialogue Keys: Secondary
   - Dialogue Type
   - Summary description
   - Beginning Question Number
   - Dialogue description

Figure 33–3  Dialogue Descriptions screen

The first Question Entry screen displays.
3. Complete the following fields:
   - Question Number
   - System Code
   - Category
3. Release
4. Subject
5. Show
6. Tickler
7. Question Description

4. Complete the following field to assist in future searches for this question:
   - Additional Keywords

**Figure 33–4  Question Entry (New) screen**

The Answer Entry screen displays.
5. Complete the following fields:
   - Question Number
   - Answer Number
   - Next Question
   - Answer

6. Optionally, complete the Return Value field.

   The Return Value field can contain a member name, or *PROMPT. *PROMPT lets the user manually complete the Return Value field.

   The following screen illustrates the use of the Return Value field. The return value is the program type for an interactive window program. In this case, there is no next question. The dialogue ends after the system enters the value E0010.

   When you click Add the value in the Next Question field changes. There is no value in the Return Value field. For any question, there is either a value in the Return Value field or the Next Question field.
Figure 33–6  Answer Entry (Next Question) screen

For an existing question and answer, you choose Display Next Question (F19) and Display Previous Question (F20) to review all other possible answers for this question.

7. To create a second answer to the question, click Enter.
   The screen clears.

8. Enter the number of the next answer in the following field:
   - Answer Number

9. Repeat the previous steps as necessary to enter the next question and answer.

10. To define the next question, click Exit to return to the Question Entry screen.

11. Repeat the previous steps as necessary to enter questions and answers.

12. Click Exit until the Simple Question & Answer screen displays.

### 33.4 Working with an Existing Dialogue

You can perform the following tasks using the Update Existing Q & A Dialogue feature:

- To review a dialogue flow
- To change a dialogue
- To copy a dialogue
- To rename a dialogue
- To run a dialogue
- To delete a dialogue
- To run a quiz
To perform these tasks, choose Update Existing Q & A Dialogue on the Simple Question and Answer screen to access the Dialogue Lists screen.

On Dialogue Lists, you can use the following values in the Option field:

2 - Change the questions and answers for the Dialogue
3 - Copy one dialogue to another dialogue
5 - Run the Q & A. You can specify the number of responses to allow.
6 - Flow displays the flow of the Q&A and how one question leads to another. You can access Q&A Revisions from the flow.
7 - Rename
9 - Delete
11 - Take a quiz from this screen

**To review a dialogue flow**

1. On Dialogue Lists, enter 6 in the Option field.
   
The Dialogue Flow Revisions screen displays.

   **Figure 33–7  Dialogue Flow Revision screen**

2. Choose Alternate Format (F11) to view the Alternate Format.
To change a dialogue
1. On Dialogue Lists, enter 2 in the Option field next to the dialogue you want to change.
   The Question Entry screen displays.
2. Click Change.
3. Make the changes to the questions and answers.

To copy a dialogue
1. On Dialogue Lists, enter 3 in the Option field next to the dialogue you want to copy.
   The Dialogue Copy screen displays.
2. Enter the name of the new dialogue list keys (primary key) in the following field:
   - Member
3. Enter the name of the new dialogue list keys (secondary key) in the following field:
   - Data item
4. Enter the name in which the new dialogue resides in the following field:
   - To Library
   The system copies the dialogue.

**To rename a dialogue**
1. On Dialogue Lists, enter 3 in the Option field next to the dialogue you want to rename.
   The Dialogue Copy screen displays.
Figure 33–10  Dialogue Copy (Rename) screen

2. Enter the name of the primary key in the following field:
   - Member

3. Enter the name of the secondary key in the following field:
   - Data item

4. Enter the name in which the new dialogue resides in the following field:
   - To Library

   The system renames the dialogue.

To run a dialogue

1. On Dialogue Lists, enter 5 in the Option field next to the dialogue you want to run.

   The Dialogue Test screen displays.
2. Enter a number for the maximum number of times you want to run this dialogue in the following field:
   - Maximum responses
   The questions of the dialogue display in sequence. When you reach the last question, a message displays at the bottom of the Dialogue Selection screen.

3. Choose Review Selections (F5) when the last question displays, to access the Quiz Answer Review screen.
4. The screen displays with the information about your answers. To review the remaining questions and answers, choose Display Next Question (F19) and Display Previous Question (F20).

5. Enter 4 to return to a specific question.

6. Click Exit on the last question screen to display the Dialogue Test screen.

**Figure 33–13  Dialogue Test (Exit) screen**

The Dialogue Test screen displays the values for number of times the dialogue was run, the Responses Returned at the end of the dialogue, and the Answer Number of the last question.

7. Click Exit (F3) to return to the Dialogue Lists screen.

**To delete a dialogue**

On Dialogue Lists, enter 9 in the Option field next to the dialogue you want to delete.

The system deletes the dialogue.

**To run a quiz**

To run a quiz, the dialogue type must be QUIZ.

1. On Dialogue Lists, enter 11 in the Option field next to the quiz you want to run.

   The first question of the quiz displays.

2. Answer the questions.

   When you finish answering the questions, a message displays at the bottom of the last screen, Question and Answer complete.

3. Perform one of the following:

   - To review choose Review Incorrect Answers (F5)
   - Click Enter
When you choose Review Incorrect Answers, the questions and answers display on the screen. When you click Enter, the system calculates the number of errors and displays your score.

Choose Review Incorrect Answers (F5) from this Dialogue Test screen to review your errors.

Figure 33–14  Dialogue Test (Review Incorrect Answers) screen
This chapter contains the topic:

- **Section 34.1, "Creating User Defined PDL."**

### 34.1 Creating User Defined PDL

You attach *PROCs to either a master file field or to a device file field (screen or report). If you attach them to a master file field, then the system places the code it generates in S005. If you attach them to a device file field, then the system places the code it generates in S004.

The purpose of User Defined PDL Entry Points is to allow you to create *PROCs in any subroutine and to allow them to exist without attaching them to a master file field or device file field. You define the entry points within subroutines where you enter PDL code using the Detailed Programming Facility. User Defined PDL Entry Points are a functional directive that you can enter into a primary logic module.

The User Defined PDL Entry Points cause the system to create RPG code in the same manner as users entering the PDL using the Detailed Programming Facility; however, you connect it to logic modules instead of fields.

In any primary logic module you can insert up to 99 PDL directives. Ordinarily, you number the first one PDL01, the second one PDL02, and so forth. PDL directives do not have to be in sequential order; however, each PDL directive must have a unique number within that logic module.

For all single record maintenance forms, you create a user defined PDL entry point in the mainline subroutine. Enter PDL to bring in a default value for a constant field. For example, change the logic module MAINLINE because this module creates the mainline code for all single record maintenance forms.

**Before You Begin**

You must be able to locate program types and logic modules. See Section 30.4, "Creating or Modifying Program Types" and Chapter 31, "Work with Logic Modules."

**To create user defined PDL**

**Navigation**

From Model Program Design Menu (G9361), under LOGIC MODULES, choose Create/Modify

1. On Create/Modify (Logic Module), access the Edit screen.
2. On the Edit screen, create the user defined PDL entry points within the affected primary logic modules.

   The naming convention for user defined PDL entry points is PDLxx, where xx is a two digit number between 01 and 99.

   You can either add the PDL directive to an existing line of code that does not contain a directive, or insert a new line and enter the directive on this line. The directive goes in positions 1 to 5 of the source line. If you insert a new line, remember to add the source sequence and serial number in the appropriate columns. (Move to column 80.)

   In this example, you enter PDL01 on line 9. Any PDL code that you enter for this entry point will come immediately after the statement EXSR S999 and before the test for *INLR.
3. Access the Software Versions Repository and locate the member.

4. Enter 10 (Design) next to the environment in the Option field:
   The Program Generator Specification screen displays.
5. On Define Generator Specification, enter 1 in the Option field to access the Detailed Programming Facility:

The Detailed Programming Facility screen displays.

6. Enter the PDL code through the Detailed Programming Facility.

All user defined PDL entry points appear after the form or report file fields in the Detailed Programming Facility.
7. On Detailed Programming Facility, enter 6 in the Option field to access Data Item Formula Revisions.

8. Enter the PDL code.

*Figure 34–6 Data Item Formula Revisions (User Defined PDL) screen*
This appendix contains these topics:

- **Section A.1, "Data File Design Aid,"
- **Section A.2, "Screen Design Aid,"
- **Section A.3, "Report Design Aid,"
- **Section A.4, "Program Generator."

You can use the following to aid you as you use the Program Generator. It includes topics that you should consider as you use the Program Generator.

### A.1 Data File Design Aid

Attempt to create files with keys to avoid having to process by relative record numbers. Processing by relative record numbers is more complex than processing by keys.

### A.2 Screen Design Aid

For Subfile Programs, you should define:

- A hidden field for the parts of the file key that the subfile video uses if you create a maintenance subfile.
- The hidden field, SH#RRN if processing by relative record numbers.
- A hidden field for the data structure if processing by relative record numbers.

You should:

- Note the video fields that relate to VC0 fields. You need this information in the Detailed Programming Facility in order to load the VC0 fields.
- Note the error indicators the system assigns to screen fields.
- Prepare final checks:
  - Did you assign the Default Cursor keyword to a video field?
  - Did you allow for upper and lower case on description fields?
  - Did you specify a K in the Edited field for the key fields only?
Note: The Screen Design Aid is currently not available in the Java platform for JD Edwards World software. You must perform these tasks using the green screen platform of the software.

A.3 Report Design Aid

- Change the Start/End lines for format HEADING1 from 1-4 to the length you need. Usually 1-8 will suffice.
- Add DETAIL1 format.
- Add TOTAL1 format if using hierarchical (dynamic) totaling. You must include one or all of the following fields which enable dynamic totaling:
  VC1ROW, VC1KEY, VC1DSC, VC0TO2
- Add HEADING2 format if using subheadings. You must include one or all of the following fields that enable subheadings to work properly:
  VC0ROW, VC0KEY, VC0DSC

A.4 Program Generator

- All Programs:
  - Ensure the CAP Status is set to Y. If CAP Status is not set to Y, then something could have ended abnormally.
  - Verify the program type.
- File Specifications:
  - You must specify one file with an M. Do not specify one file with an M and another with a 1 as the Program Generator views both files on an equal basis. Specify the main file as an M and subsequent files starting with 2.
  - You must specify a video or report file.
  - You cannot include description files if a field is in the HEADING2 format for a report.
- Option and Function Exits:
  - Ensure the program the system is launching is setup to accept parameters sent by the function key or selection exit.
  - Modify any CL programs that also launch a program to send blank parameters. For example, CALL Pxxxxx PARM( )
  - Ensure the program to retrieve exists before using it on this screen.
  - Attempt to send PSxxxx fields instead of VDxxxx or SFxxxx fields.
    - May inadvertently change in the program the system launches.
    - You might have to define and load the PSxxxx fields manually.
- Detailed Programming Facility:
  - Enter N in the Entry Optional field for key fields in a subfile. You specify this for the subfile fields, not the hidden fields as it enables the delete function.
  - Link VC0 fields to description files.
– Use PDL in the Detailed Programming Facility to:
– Affect subroutine S005 when you enter it with a data base field.
– Affect subroutine S004 when you enter it with a video field.
– Use the Return keyword to omit the standard code the Program Generator creates.

■ Specify a PLIST sequence if the program is going to receive parameters from another program. Use the video fields for this instead of the file fields.

■ Use *OUTPUT to retrieve the row description from the Data Dictionary for fields that you only use in the HEADING2 format and not the DETAIL1 format.
This appendix contains these topics:

- Section B.1, "Error Handling,"
- Section B.2, "Indicator Usage,"
- Section B.3, "Naming Conventions,"
- Section B.4, "Key List (KLIST),"
- Section B.5, "Work Fields,"
- Section B.6, "Current Date and Time."

**B.1 Error Handling**

JD Edwards World includes an efficient means of handling errors using arrays. The following illustrate the error handling arrays within the Single Record Maintenance Program.

- The EMK array contains the four byte data dictionary name of every error that can occur in this program.
- The @MK array maintains an indicator for each error in EMK. If one of the errors occurs, the system activates the indicator.
- The @ER array loads the error messages when you choose Display Error Message (F7) to view the errors.

The code to launch the error message handling program follows.
If an error indicator is set to one, the program moves the corresponding data item from the array of all possible errors (EMK) into the array of the errors (@ER).

The system loads the array which contains every possible error for this program only once (in S999).

## B.2 Indicator Usage

There are 99 indicators available for use. JD Edwards World groups them by purpose and includes standards for the use of the indicators that are in the following table. JD Edwards World does not include standards for indicators that are not in the following table.

<table>
<thead>
<tr>
<th>INDICATOR</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Causes the Invalid Function Key Pressed message to display</td>
</tr>
<tr>
<td>02</td>
<td>Dictates the color palette to use</td>
</tr>
<tr>
<td>04</td>
<td>Controls subfile keywords SFLDROP and SFLFOLD for fold areas</td>
</tr>
<tr>
<td>20</td>
<td>Controls the clear screen action code</td>
</tr>
<tr>
<td>21</td>
<td>Controls the add action code</td>
</tr>
<tr>
<td>22</td>
<td>Controls the change action code</td>
</tr>
<tr>
<td>23</td>
<td>Controls the delete action code</td>
</tr>
<tr>
<td>24</td>
<td>Controls the inquire action code</td>
</tr>
<tr>
<td>31</td>
<td>The system uses this in conjunction with subfile processing to initiate the INVITE or SFLCLR keyword</td>
</tr>
<tr>
<td>32</td>
<td>The system uses this in conjunction with subfile processing initiating the keyword SFLNXTCHG</td>
</tr>
<tr>
<td>37</td>
<td>The system uses this in conjunction with subfile processing to highlight the last record in the display (used only with inquiry subfiles)</td>
</tr>
</tbody>
</table>
**B.3 Naming Conventions**

Use the following characters in the first place of the naming convention to distinguish different item names:

- `@` - Array names
- `$` - Field names the program creates for flags and work fields
- `#` - Fields you define in common subroutines

**B.4 Key List (KLIST)**

Define key lists in the housekeeping subroutine.

Begin the key list name with the data file prefix. For example, the Address Book Master file prefix is AB, so the key list would be ABKY01.

The program generator creates key lists using the following naming conventions:

- `XXKY01` for physical files where `XX` is the file prefix. For example: ABKY01
- When a physical file needs to have more than one key list in a program, you denote the successive files by the last character. For example, for three key lists for the physical F0101, the key lists are: ABKY01, ABKY02, and ABKY03.
- `XXKY0x` for logical files where `XX` is equal to the file prefix and `X` is equal to the last letter of the logical file name. For example: ABKY0A for F0101LA, ABKY0B for F0101LB
- When a logical file needs to have more than one key list in a program, you denote the successive files by the penultimate character. For example, the three key lists for the logical F0101LA are: ABKY0A, ABKY1A, and ABKY2A.

### INDICATOR DESCRIPTION

<table>
<thead>
<tr>
<th>INDICATOR</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>The system uses this in conjunction with subfile processing to control the display keyword SFLDSP</td>
</tr>
<tr>
<td>42-79</td>
<td>The system uses this for error processing to indicate which fields are in error</td>
</tr>
<tr>
<td>40</td>
<td>The system reserves this for errors in the Action Code field</td>
</tr>
<tr>
<td>41</td>
<td>The system reserves this for errors in the key fields</td>
</tr>
<tr>
<td>80-89</td>
<td>Indicates a general reusable one-time indicators</td>
</tr>
<tr>
<td>93</td>
<td>Indicates global error indicator that highlights line 24</td>
</tr>
<tr>
<td>98</td>
<td>Indicates a chain or read failure</td>
</tr>
<tr>
<td>99</td>
<td>Indicates a record is in use</td>
</tr>
<tr>
<td>OF</td>
<td>Indicates overflow for report processing</td>
</tr>
<tr>
<td>LR</td>
<td>Indicates that the last record has been read and the program should end normally</td>
</tr>
<tr>
<td>RT</td>
<td>Indicates that a temporary or final halt in the program should take place and returns to the calling program leaving files open</td>
</tr>
</tbody>
</table>
B.5 Work Fields

Define work fields only once within a program. JD Edwards World recommends that you use the LIKE DEFN command to define work fields when their attributes directly tie to those of database fields.

For example, if the work field needs to have the same attributes as a field that exists in a file:

MOVE ABANS $ANS,

You then define $ANS as follows:

*LIKE DEFN ABANS $ANS

The advantage of this method is that the work field and database field retain the same attributes even if the database field changes.

When using work fields as an indicator, you should assign the prefix $ and enter descriptive text for the remainder of the name. For example, a work field name such as $GLOBL is more descriptive than a field name such as $G.

For numeric indices, use the fields in the data structure I00DSINX.

B.6 Current Date and Time

When retrieving the current date and time, use the TIME operation code instead of UDATE. UDATE obtains the date format of the system on which you compiled the program. You cannot change the date format without recompiling the program. TIME uses the system's date format when it executes the operations code.

Note: The TIME operation requires significant system resources. If possible, use it only once for a program. Typically, this would occur in the Housekeeping Subroutine (S999).

Always use program X0028 to edit dates and format them for output.
This appendix contains these topics:

- Section C.1, "Guidelines,"
- Section C.2, "A0010 - Interactive Subfile Inquiry,"
- Section C.3, "A0020 - Interactive Single Record Inquiry,"
- Section C.4, "B0010 - Interactive Single Record Maintenance,"
- Section C.5, "C0010 - Batch Report with Totals,"
- Section C.6, "C0020 - Batch Report with Totals and Subheadings,"
- Section C.7, "C0025 - Batch Report with Totals and Subheadings,"
- Section C.8, "D0010 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number,"
- Section C.9, "D0020 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number,"
- Section C.10, "D0030 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number with Read Next Modified Record,"
- Section C.11, "D0040 - Interactive Subfile Maintenance with Action Code, with Options, by Key,"
- Section C.12, "D0050 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Relative Record Number,"
- Section C.13, "D0060 - Interactive Subfile Maintenance with Action Code, without Options, by Key,"
- Section C.14, "D0070 - Interactive Subfile Maintenance with Action Code, with Options, by Relative Record Number,"
- Section C.15, "D0080 - Interactive Subfile Maintenance without Action Code, with Options, by Relative Record Number,"
- Section C.16, "D0090 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number, Balance,"
- Section C.17, "D0100 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Key,"
- Section C.18, "E0010 - Interactive Window,"
- Section C.19, "X0010 - Batch Update with Report,"
- Section C.20, "X0020 - Batch Update,"
This appendix includes each program type, the requirements for each program type, and its use. Use this as a quick reference for all program types.

C.1 Guidelines

The following are optional:

- General help instructions, however JD Edwards World highly recommends you include these.
- Detail (fold) areas and AAIs within program types.
- Processing Options. Define processing options for batch processing. The step in the program generator which automatically includes the logic to retrieve this information is subroutine S999. Define the special calculations to use the processing options.
- User defined options and function exits for all program types.
- Calculations you can create using Program Design Language in the Detailed Programming Facility.
- VC0 description fields updates.

C.2 A0010 - Interactive Subfile Inquiry

C.2.1 Description

Use this program type to create an interactive subfile program. This subfile program is for inquiry purposes only. This program type processes a single master file by key. You cannot set the Lockout Action Codes fields. Create a display file prior to generating this program type.

C.2.2 Display File Definition

This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of the Screen Design Aid (SDA) with the value K. If you are using the Data Base Field Selection feature in SDA, the known key field updates automatically.

The use of an Action Code is optional. Enter a default cursor location if there is no action code.

C.2.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program to use with this program type. You can use the Quick Start CL Generator to automatically create your CL program.
C.2.4 File Specifications

This program type requires that you define a single master file and a display file. The master file contains M or 1 in the Input column. The display file begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.2.5 Detailed Programming Facility

Use a key list for record retrieval from the master file. If you are not using the complete key list, update the Key Sequence Field in the Detailed Programming Facility to include only those data items which are necessary. This key list should match your key field definition from the control format of the display file.

C.2.6 Special Considerations

Add special logic if you want to process the master file using the key as a restrictive key. The default logic performs a SETLL which positions the records from the file using the key and then scans without a key until the subfile fills.

C.2.7 Quick Start Generation

You can generate this program type using the Quick Start CL Generator.

C.3 A0020 - Interactive Single Record Inquiry

C.3.1 Description

Use this program type to create an interactive single record program. This program is for inquiry purposes only. Create a display file prior to generating this program type. This program type processes a single master file by key.

C.3.2 Display File Definition

This program type scrubs the key field in the display file prior to processing the master file. The key field is noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you are use the Data Base Field Selection feature in SDA, the known key field updates automatically.

The use of an Action Code is optional. Enter a default cursor location if there is no action code. This program type does not use Lockout Action Codes.

C.3.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.3.4 File Specifications

This program type requires that you define a single master file and a display file. The master file contains M or 1 in the Input column. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.
C.3.5 Special Considerations
This program type uses a key list for record retrieval from the master file. This key list should match your key field definition from the control format of the display file. One record displays per inquiry.

C.3.6 Quick Start Generation
You cannot generate this program type using the Quick Start CL Generator.

C.4 B0010 - Interactive Single Record Maintenance

C.4.1 Description
Use this program type to create an interactive single record maintenance program. Create a display file prior to generating this program type. This program type processes a single master file by key.

C.4.2 Display File Definition
This program type scrubs the key field in the display file prior to processing the master file. The key field is noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you are using the Data Base Field Selection feature in SDA, the known key field updates automatically.

You must use an Action Code. Lockout Action Codes are optional.

C.4.3 CL Program Definition
Copy and revise the model CL program J98MODEL1 to create a CL program to use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.4.4 File Specifications
This program type requires that you define a single master file and a display file. The master file contains M or 1 in the Update column. The display file begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.4.5 Detailed Programming Facility
Use a key list for record retrieval from the master file. If you are not using the complete key list, update the Key Sequence Field in the Detailed Programming Facility to include only those data items which are necessary. This key list should match your key field definition from the control format of the display file.

C.4.6 Quick Start Generation
Generate this program type using the Quick Start CL Generator.

C.5 C0010 - Batch Report with Totals
C.5.1 Description
Use this program type to create a batch report program that you manipulate using DREAM Writer. Create a printer file prior to generating this program type. This program type processes a single master file. You choose the data for the batch report using DREAM Writer Data Selection and Data Sequence parameters. The Batch Report with Totals program type does not use Lockout Action Codes and user defined options and function exits.

C.5.2 Printer File Definition
This program type requires that formats HEADING1 and DETAIL1 exist in the printer file. Format TOTAL1 is optional for totals.

C.5.3 CL Program Definition
Copy and revise the model CL program J98MODEL6 to create a CL program to use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.5.4 File Specifications
This program type requires that you define a single master file and a printer file. The master file contains M or 1 in the Input column. The printer file begins with R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.5.5 Special Considerations
If you are printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).

When creating your DREAM Writer Version, ensure that the value of 2 is in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchal totaling and page breaks in the Data Sequence.

C.5.6 Quick Start Generation
Generate this program type using the Quick Start CL Generator.

C.6 C0020 - Batch Report with Totals and Subheadings

C.6.1 Description
Use this program type to create a batch report program that you manipulate using DREAM Writer. Create a printer file prior to generating this program type. This program type processes a single master file. You choose the data for the print program using DREAM Writer Data Selection and Data Sequence parameters. The Batch Report with Totals and Subheadings program type does not use Lockout Action Codes and user defined options and function exits.
C.6.2 Printer File Definition

This program type requires that formats HEADING1, HEADING2 and DETAIL1 exist in the printer file. Format TOTAL1 is optional for totals.

C.6.3 CL Program Definition

Copy and revise the model CL program J98MODEL6 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.6.4 File Specifications

This program type requires that you define a single master file and a printer file. The master file contains M or 1 in the Input column. The printer file begins with an R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.6.5 Special Considerations

If printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).

If printing subheadings using format HEADING2, use the special keywords for Data Dictionary description (VC0ROW), data key field (VC0KEY), and data key description (VC0DSC).

When creating your DREAM Writer Version, ensure that the value of 2 is in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchical totaling and page breaks in the Data Sequence.

C.6.6 Quick Start Generation

Generate this program type using the Quick Start CL Generator.

C.7 C0025 - Batch Report with Totals and Subheadings

C.7.1 Description

Use this program type to create a batch report program that you manipulate using DREAM Writer. Create a printer file prior to generating this program type. This program type processes a single master file. You choose the data for the print program using DREAM Writer Data Selection and Data Sequence parameters. The Batch Report with Totals and Subheadings program type does not use Lockout Action Codes and user defined options and function exits.

C.7.2 Printer File Definition

This program type requires that formats HEADING1, HEADING2 and DETAIL1 exist in the printer file. Format HEADING2 is the format that prints subheadings. Format TOTAL1 is optional for totals.
C.7.3 CL Program Definition

Copy and revise the model CL program J98MODEL6 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.7.4 File Specifications

This program type requires that you define a single master file and a printer file. The master file contains M or 1 in the Input column. The printer file begins with R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.7.5 Special Considerations

This program type is not a Q&A response in the Program Purpose and Type step. To use this program type, enter an X in the first Q&A question to provide this program type name.

This program type is identical to C0020 except that the subheadings headings print above the column headings. When you use this program type, control the page breaks to match the subheadings.

If you are printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).

If printing subheadings using format HEADING2, use the special keywords for Data Dictionary description (VC0ROW), data key field (VC0KEY), and data key description (VC0DSC).

When creating your DREAM Writer Version, ensure that the value of 2 is in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchal totaling and page breaks in the Data Sequence.

C.7.6 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.

C.8 D0010 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number

C.8.1 Description

Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and controls the sequence in which the records display. The secondary master file processes by relative record number and controls the database updates.

C.8.2 Display File Definition

This program type scrubs the key field in the control format of the display file prior to processing the master file. The key field is noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key field update automatically.
You must use an Action Code. Lockout Action Codes are optional.

The system stores the record number of each subfile record in a hidden relative record number field. Add the field SH#RRN to the subfile format with S in the Type field and 9.0 in the Size field, using the Display All Defined Fields in the SDA.

C.8.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.8.4 File Specifications

This program type requires that you define a keyed master file, a secondary master file which is not keyed, and a display file. The master file contains 1 in the Input column. You enter a value in the File Information Data Structure field in the fold area of the primary master file. The secondary master file contains 2 in the Update column and N in the Keyed (Y/N) field in the fold area. The display file begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.8.5 Detailed Programming Facility

Use a selection exit 4 to exit to the Detailed Programming Facility for the subfile field controlling the update to the database. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database. Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page down processing, locate the field within the second master file that is the key to the primary keyed master file. Use selection exit 4 to display the Detailed Programming Facility and enter N in the Clear After (Y/N) field. This prevents the key field for page up and page down key processing from clearing.

If you are creating an inquiry which uses a partial key list from the master file, you must change the Key Sequence field (KY PS) in the Detailed Programming Facility. The KY column is to the right of the master file field names and contains the sequence number for the key fields. Clear all sequence numbers that are not in the key search that you define in the control format of the display file. The key sequence you define in the Detailed Programming Facility should match the key fields in the control format.

C.8.6 Special Considerations

This program type uses the key information in the display file for positioning within the master file. This program type must also have a hidden relative record number field and an entry optional field.

C.8.7 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.
C.9 D0020 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number

C.9.1 Description
Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and controls the sequence in which the records display. The secondary master file processes by relative record number and controls the database updates.

C.9.2 Display File Definition
This program type scrubs the key field in the control format of the display file prior to processing the master file. The key field is noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key field update automatically.

This program type does not use an Action Code. Enter a default cursor location.

The system stores the record number of each subfile record in a hidden relative record number field. Add the field SH#RRN to the subfile format with a type of S and a size of 9.0 by using Display All Defined Fields in SDA.

C.9.3 CL Program Definition
Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.9.4 File Specifications
This program type requires that you define a keyed master file, a secondary master file which is not keyed and a display file. The master file contains 1 in the Input column. You enter a value in the File Information Data Structure field in the fold area of the primary master file. The secondary master file contains a 2 in the Update column and N in the Keyed (Y/N) field in the fold area. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.9.5 Detailed Programming Facility
Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the database update. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page down processing, locate the field within the second master file that is the key to the primary keyed master file. Use selection exit 4 to display the Detailed Programming Facility and enter N in the Clear After field. This prevents the key field for page up and page down processing from clearing.
If you are creating an inquiry which uses a partial key list from the master file, you must change the Key Sequence field (KY PS) in the Detailed Programming Facility. The KY column is to the right of the master file field names and contains the sequence number for the key fields. Clear all sequence numbers that are not in the key search that you define in the control format of the display file. The key sequence you define in the Detailed Programming Facility should match the key fields in the control format.

C.9.6 Special Considerations
This program type uses the key information in the display file for positioning within the master file. This type must also have a hidden relative record number field and an entry optional field.

C.9.7 Quick Start Generation
You cannot generate this program type using the Quick Start CL Generator.

C.10 D0030 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number with Read Next Modified Record

C.10.1 Description
Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and controls the sequence in which the records are display. The secondary master file processes by relative record number and controls the database updates. The system bases updates to the subfile on read next change (READC) logic.

C.10.2 Display File Definition
This program type scrubs the key field in the control format of the display file prior to processing the master file. The key field is noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key fields update automatically.

This program type does not use an Action Code. Enter a default cursor location. The system stores the record number of each subfile record in a hidden relative record number field. Add the field SH#RRN to the subfile format with a type of S and a size of 9.0 by using Display All Defined Fields in SDA.

C.10.3 CL Program Definition
Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.10.4 File Specifications
This program type requires that you define a keyed master file, a secondary master file which is not keyed and a display file. The master file contains 1 in the Input column. You enter a value in the File Information Data Structure field in the fold area of the
primary master file. The secondary master file contains a 2 in the Update column and N in the Keyed (Y/N) field in the fold area. The display begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.10.5 Detailed Programming Facility

Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the update to the database. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page down processing, locate the field within the second master file that is the key to the primary keyed master file. Use selection exit 4 to display the Detailed Programming Facility and enter N in the Clear After field. This prevents the key field for page up and page down processing from clearing.

If you are creating an inquiry which uses a partial key list from the master file, you must change the Key Sequence field (KY PS) in the Detailed Programming Facility. The KY column is to the right of the master file field names and contains the sequence number for the key fields. Clear all sequence numbers that are not in the key search that you define in the control format of the display file. The key sequence you define in the Detailed Programming Facility should match the key fields in the control format.

C.10.6 Special Considerations

This program type uses the key information in the display file for positioning within the master file. This type must also have a hidden relative record number field and an entry optional field.

C.10.7 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.

C.11 D0040 - Interactive Subfile Maintenance with Action Code, with Options, by Key

C.11.1 Description

Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes a single master file by key.

C.11.2 Display File Definition

This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you are using the Data Base Field Selection feature in SDA, the known key fields update automatically.
You must define Action Codes. Lockout Action Codes are optional.

This subfile maintenance program type allows you to delete individual subfile records using special logic. You perform this by entering C in the Action Code field, comparing the previous value with the current value and then deleting the record if the current value is blank. The system stores the previous value in a hidden field at the subfile record level using the Display All Defined Fields in the SDA.

C.11.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.11.4 File Specifications

This program type requires that you define a single master file and a display file. The master file contains M or 1 in the Update column. The display file begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.11.5 Detailed Programming Facility

Use a selection 4 to exit to the field details for the subfile field controlling the database update. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

C.11.6 Special Considerations

This program type uses the key information in the display file for chaining to the master file. This type must also have a hidden field and an entry optional field.

C.11.7 Quick Start Generation

Generate this program type using the Quick Start CL Generator.

C.12 D0050 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Relative Record Number

C.12.1 Description

Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and the system updates it from the fields in the control format of the display file. The secondary master file processes by relative record number and the system updates it from the fields in the subfile format of the display file.

C.12.2 Display File Definition

This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key fields update automatically.
You must define Action Codes. Lockout Action Codes are optional.

The system stores the record number of each subfile record in a hidden relative record number field. Add the field SH#RRN to the subfile format with a type of S and a size of 9.0 by using Display All Defined Fields in SDA.

C.12.3 CL Program Definition
Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.12.4 File Specifications
This program type requires that you define a two keyed master files, a secondary master file which is not keyed and a display file. The first master file contains 1 in the Update column. The system updates this file from the control format of the display file. The second master file is a non-keyed file which the subfile format of the display file updates. The second master file contains 2 in the Update column and X under the Add column. Enter N in the Keyed (Y/N) field in the fold area. The third master file is the logical file that the system uses for sequencing records in the subfile. This file contains 3 in the Input column. You enter a value in the File Information Data Structure field in the fold area of this master file. The keyed master files have a similar key list sequence. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.12.5 Detailed Programming Facility
Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the database update function. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page down processing, locate the field within the second master file that is the key to the primary keyed master file. Use selection exit 4 to display the Detailed Programming Facility and enter N in the Clear After field. This prevents the key field for page up and page down processing from clearing.

If you are creating an inquiry which uses a partial key list from the master file, you must change the Key Sequence field (KY PS) in the Detailed Programming Facility. The KY column is to the right of the master file field names and contains the sequence number for the key fields. Clear all sequence numbers that are not in the key search that you define in the control format of the display file. The key sequence you define in the Detailed Programming Facility should match the key fields in the control format.

C.12.6 Special Considerations
This program type uses the key information in the subfile control format of the display file for retrieving one record from the first master file and multiple records from the second master file. This type must also have a hidden field and an entry optional field.
C.13 D0060 - Interactive Subfile Maintenance with Action Code, without Options, by Key

C.13.1 Description
Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes a single master file by key.

C.13.2 Display File Definition
This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key fields update automatically.

You must define Action Codes. Lockout Action Codes are optional.

This subfile maintenance program type allows you to delete individual subfile records using special logic. You perform this logic by entering C in the Action Code, comparing the previous value with the current value and then deleting the record if the current value is blank. The system stores the previous value in a hidden field at the subfile record level. Define this field in the display file prior to generating this program type by using Display All Defined Fields in SDA.

C.13.3 CL Program Definition
Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.13.4 File Specifications
This program type requires that you define a single master file and a display file. The master file contains an M or 1 in the Update column. The display file begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.13.5 Detailed Programming Facility
Use a selection 4 to exit to the field details for the SF field controlling the update to the database. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

C.13.6 Special Considerations
This program type uses the key information in the display file for chaining to the master file. This type must also have a hidden field and an entry optional field.
C.13.7 Quick Start Generation

Generate this program type using the Quick Start CL Generator.

C.14  D0070 - Interactive Subfile Maintenance with Action Code, with Options, by Relative Record Number

C.14.1 Description

Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and controls the sequence in which the records display. The secondary master file processes by relative record number and controls the database updates.

C.14.2 Display File Definition

This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you are using the Data Base Field Selection feature in SDA, the known key fields update automatically.

You must define the Action Codes. Lockout Action Codes are optional.

The system stores the record number of each subfile record in a hidden relative record number field. Add the field SH#RRN to the subfile format with a type of S and a size of 9.0 by using Display All Defined Fields in SDA.

C.14.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.14.4 File Specifications

This program type requires that you define a keyed master file, a secondary master file which is not keyed, and a display file. The master file contains 1 in the Input column. You enter a value in the File Information Data Structure field in the fold area of the primary master file. The secondary master file contains a 2 in the Update column and N in the Keyed (Y/N) field in the fold area. The display file begins with a V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.14.5 Detailed Programming Facility

Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the database update. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page
down processing, locate the field within the second master file that is the key to the primary key.

If you are creating an inquiry which uses a partial key list from the master file, you must change the Key Sequence field (KY PS) in the Detailed Programming Facility. The KY column is to the right of the master file field names and contains the sequence number for the key fields. Clear all sequence numbers that are not in the key search that you define in the control format of the display file. The key sequence you define in the Detailed Programming Facility should match the key fields in the control format.

C.14.6 Special Considerations

This program type uses the key information in the display file for positioning within the master file. This type must also have a hidden relative record number field and an entry optional field.

C.14.7 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.

C.15 D0080 - Interactive Subfile Maintenance without Action Code, with Options, by Relative Record Number

C.15.1 Description

Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and controls the sequence in which the records display. The secondary master file processes by relative record number and controls the database updates.

C.15.2 Display File Definition

This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key fields update automatically.

Do not use Action Codes. Enter a default cursor location.

This program type processes the secondary master file by relative record number. The record number of each subfile record is stored in a hidden relative record number field.

Add the field SH#RRN to the subfile format with a type of S and a size of 9.0 by using Display All Defined Fields in SDA.

C.15.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.
C.15.4 File Specifications

This program type requires that you define a keyed master file, a secondary master file which is not keyed and a display file. The master file contains 1 in the Input column. You enter a value in the File Information Data Structure field in the fold area of the primary master file. The secondary master file contains a 2 in the Update column and N in the Keyed (Y/N) field in the fold area. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.15.5 Detailed Programming Facility

Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the database update. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page down processing, locate the field within the second master file that is the key to the primary keyed master file. Use selection exit 4 to display the Detailed Programming Facility and enter N in the Clear After field. This prevents the key field for page up and page down processing from clearing.

If you are creating an inquiry which uses a partial key list from the master file, you must change the Key Sequence field (KY PS) in the Detailed Programming Facility. The KY column is to the right of the master file field names and contains the sequence number for the key fields. Clear all sequence numbers that are not in the key search that you define in the control format of the display file. The key sequence you define in the Detailed Programming Facility should match the key fields in the control format.

C.15.6 Special Considerations

This program type uses the key information in the display file for positioning within the master file. This type must also have a hidden relative record number field and an entry optional field.

C.15.7 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.

C.16 D0090 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number, Balance

C.16.1 Description

Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and controls the sequence in which the records display. The secondary master file processes by relative record number and controls the database updates. The system verifies all records before it performs any database updates.
C.16.2 Display File Definition

This program type scrubs the key fields in the control format of the display file for positioning within the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key fields update automatically.

Do not use Action Codes. Enter a default cursor location.

This program type processes the secondary master file by relative record number. The record number of each subfile record is stored in a hidden relative record number field. Add the field SH#RRN to the subfile format with a type of S and a size of 9.0 by using Display All Defined Fields in SDA.

C.16.3 CL Program Definition

Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator for creation of your CL program.

C.16.4 File Specifications

This program type requires that you define a keyed master file, a secondary master file which is not keyed, and a display file. The master file contains 1 in the Input column. You enter a value in the File Information Data Structure field in the fold area of the primary master file. The secondary master file contains a 2 in the Update column and N in the Keyed (Y/N) field in the fold area. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.16.5 Detailed Programming Facility

Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the database update. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

Because there are two master files for this program type, you must add special logic to control the page up and page down keys. Subroutine S001 contains logic to clear all non-key fields for each of the master files. Since the second master file has no keys, all fields clear. This causes the page up and page down processing for the keyed master file to work incorrectly after the first subfile page fills. To rectify the page up and page down processing, locate the field within the second master file that is the key to the primary keyed master file. Use selection exit 4 to display the Detailed Programming Facility and enter N in the Clear After field. This prevents the key field for page up and page down processing from clearing.

C.16.6 Special Considerations

This program type uses the key information in the display file for positioning within the master file. This type must also have a hidden field and an entry optional field. The update logic in this program type processes all subfile transactions prior to performing the database updates. This allows the system to balance transactions or verify all records before it updates the database.

C.16.7 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.
C.17 D0100 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Key

C.17.1 Description
Use this program type to create an interactive subfile maintenance program. Create a display file prior to generating this program type. This program type processes two master files. The primary master file is keyed and the system updates it from the fields in the control format of the display file. The secondary master file processes by key and the system updates it from the fields in the subfile format of the display file.

C.17.2 Display File Definition
This program type scrubs the key fields in the control format of the display file prior to processing the master file. The key fields are noted by updating the Edited Field in the Field Definition screen of SDA with the value K. If you use the Data Base Field Selection feature in SDA, the known key fields update automatically.

You must define the Action Codes. Lockout Action Codes are optional.

This program type requires that you define one or more hidden fields in the subfile record. The fields in the subfile that are keys to the second master file must also have hidden fields. Add the hidden fields by using Display All Defined Fields in SDA.

C.17.3 CL Program Definition
Copy and revise the model CL program J98MODEL1 to create a CL program for use with this program type. Use the Quick Start CL Generator for creation of your CL program.

C.17.4 File Specifications
The first master file contains 1 in the Update column. The system updates this file from the control format of the display file. The second master file contains 2 in the Update column and X in the Add column. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.17.5 Detailed Programming Facility
Use a selection exit 4 to display the Detailed Programming Facility for the subfile field controlling the database update. Enter N in the Entry Optional field. Entering N informs the program generator that the user must complete this field before the system updates the database.

C.17.6 Special Considerations
This program type uses the key information in the display file for chaining to the master file. This type must also have a hidden field and an entry optional field.

C.17.7 Quick Start Generation
Generate this program type using the Quick Start CL Generator.
C.18  E0010 - Interactive Window

C.18.1  Description
Use this program type to create an interactive window program. Create a display file prior to generating this program type. This program type processes a single master file by key.

C.18.2  Display File Definition
SDA builds the DDS for a window program when you select Fast Path Create for Window, Y. Update the predefined VTX field from Row Desc to a meaningful Skip To description. SDA defines a key field. Delete this field and add a VD field which is the same as the key to the master file. If the key field is greater than 10 in length, you must also shorten the literal field that follows that key and precedes the window border.
Do not use Action Codes.

C.18.3  CL Program Definition
A CL program is option for this model.
If you want to create a CL program, copy and revise the model CL program J98MODEL1 to create a CL program to use with this program type. Use the Quick Start CL Generator to automatically create your CL program. The interactive window program type assumes three parameters. Add these to the call statement for your program.

C.18.4  File Specifications
This program type requires the use of a single master file and a display file. The master file contains M or 1 in the Input column. The display file begins with V and has blank selection columns. Add files to retrieve descriptions if necessary.

C.18.5  Define Option and Function Key Exits
If you use this window to retrieve values to the calling program, add #SSELC to the Function Exit definitions.

C.18.6  Detailed Programming Facility
If used, update all VC0 description fields in the Detailed Programming Facility.
The system uses a key list for record retrieval from the master file. If you are not using the complete key list, update the Key Sequence field (KY PS) in the Detailed Programming Facility to include only those data items which are necessary. This key list should match your key field definition from the control format of the display file.
Update the fields MNMNI and MNMTTL with the key and the key description fields.
Subroutine S004 assumes that only two fields display per master file record. If you plan to display more than two fields, modify this subroutine.
C.18.7 Special Considerations

This program type uses a key list for record retrieval from the master file. This key list should match your key field definition from the control format of the display file. One record displays per inquiry.

Subroutine S004 assumes that only two fields display per master file record. If you plan to display more than two fields, modify Subroutine S004 through JD Edwards World SEU or *PROC. JD Edwards World includes two entry points into this subroutine.

The system updates the window key literal in the upper left hand corner of the display file at run time. Modify subroutine S999 through *PROC prior to compiling the RPG program. Assign the video screen name to the work field VC01 using the entry point in subroutine S999.

C.18.8 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.

C.19 X0010 - Batch Update with Report

C.19.1 Description

Use this program type to create a batch update program that you manipulate using DREAM Writer. Create a printer file prior to generating this program type. This program type processes a single master file. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters.

C.19.2 Printer File Definition

This program type requires that formats HEADING1 and DETAIL1 exist in the printer file. Format TOTAL1 is optional exist for totals.

C.19.3 CL Program Definition

Copy and revise the model CL program J98MODEL6 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.19.4 File Specifications

The master file contains M or 1 in the Update column. Remove default value of X in the Add column or add special logic to your program for writing to the master file. The printer file begins with R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.19.5 Special Considerations

If you are printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).
When creating your DREAM Writer Version, ensure that the value of 2 is in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchical totaling and page breaks in the Data Sequence.

This program updates the master file in subroutine S010. You might want to add special logic to control when updates occur.

C.19.6 Quick Start Generation
You cannot generate this program type using the Quick Start CL Generator.

C.20 X0020 - Batch Update

C.20.1 Description
Use this program type to create a batch update program that you manipulate using DREAM Writer. This program type processes two master files. The system uses the primary master file to retrieve data from the secondary master file. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters. Do not create user defined options and function exits.

C.20.2 Printer File Definition
You do not use a printer file with this program type.

C.20.3 CL Program Definition
Copy and revise the model CL program J98MODEL2 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.20.4 File Specifications
This program type requires that you define a key for both the master file and secondary file. The master file contains 1 in the Input column. The secondary master file contains 2 in the Update column. Add files to retrieve descriptions if necessary.

C.20.5 Special Considerations
This program type chains to the secondary master file in subroutine S003. Update the key field or key list prior to this chain. The system does not automatically update key fields in this program type.

This program type contains subroutine S005 for all calculations. Add all special logic code between the read of the primary master file and the update or write of the secondary master file.

This program updates the master file records in subroutine S010.

C.20.6 Quick Start Generation
You cannot generate this program type using the Quick Start CL Generator.
C.21 X0030 - Batch Update with Subroutine S001

C.21.1 Description
Use this program type to create a batch update program that you manipulate using DREAM Writer. This program type processes two master files. The system uses the primary master file to retrieve data from the secondary master file. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters. Do not create user defined options and function exits.

C.21.2 Printer File Definition
You do not use a printer file with this program type.

C.21.3 CL Program Definition
Copy and revise the model CL program J98MODEL2 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.21.4 File Specifications
This program type requires that you define a key for both the master file and secondary file. The master file contains 1 in the Input column. The secondary master file has 2 in the Update column. Add files to retrieve descriptions if necessary.

C.21.5 Special Considerations
This program type clears the non-key fields from the primary master file between each record the system processes.

This program type chains to the secondary master file in subroutine S003. Update the key field or key list prior to the chain. Key fields do not automatically update in this program type.

This program type includes subroutine S005 for all calculations. Add all special logic necessary between the read of the primary master file and the update of the secondary master file.

This program updates the master file records in subroutine S010.

C.21.6 Quick Start Generation
You cannot generate this program type using the Quick Start CL Generator.

C.22 X0040 - Batch Update with Report

C.22.1 Description
Use this program type to create a batch update program that you manipulate using DREAM Writer. Create a printer file prior to generating this program type. You should design the printer file to print an audit trail of each record that the system updates. This program type processes two master files. The system interprets the primary
master file and updates the second master file. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters. Do not create user defined options and function exits.

C.22.2 Printer File Definition

This program type prints an audit trail for each record that the system saves in the second master file. Formats HEADING1 and DETAIL1 must exist in the printer file. Format TOTAL1 is optional, and you can use this to have the system compute totals for the level breaks that you define in the DREAM Writer Data Sequence.

C.22.3 CL Program Definition

Copy and revise the model CL program J98MODEL2 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.22.4 File Specifications

This program type requires that you define a key for both the master file and secondary file. The master file contains 1 in the Input column. The secondary master file has 2 in the Update column. Add files to retrieve descriptions, if necessary.

C.22.5 Special Considerations

This program type chains to the secondary master file in subroutine S003. Update the key field or key list prior to this chain. Key fields do not automatically update in this program.

This program type uses subroutine S004 to format fields for output to the report. Add any special logic necessary between the read of the primary master file and the update of the second master file.

This program type uses subroutine S005 to scrub and edit the fields in the second master file. Use the Detailed Programming Facility to associate fields in the primary master file with fields in the second master file. Add any special logic that is necessary to compute the proper value that the system loads to the output fields.

Subroutine S010 controls the printing of the report.

Subroutine S011 updates the records in the second master file.

C.22.6 Quick Start Generation

You can not generate this program type using the Quick Start CL Generator.

C.23 Y0010 - Conversion, Two Files with Error Report

C.23.1 Description

Use this program type to create a batch conversion program that you manipulate using DREAM Writer. This program type processes two master files. The system interprets the primary master file and updates the second master file. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters. Do not create user defined options and function exits.
C.23.2 Printer File Definition

This program type requires that formats HEADING1, DETAIL1, and ERROR1 exist in the printer file. Format TOTAL1 is optional for totals.

C.23.3 CL Program Definition

Copy and revise the model CL program J98MODEL6 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.23.4 File Specifications

This program type requires that you define a key for both the master file and secondary file. The master file contains 1 in the Input column. The secondary master file had 2 in the Update column. The printer file begins with R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.23.5 Special Considerations

If you are printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).

When printing the error report, format ERROR1 must contain the special fields for error message description (RRDSCA) and error message number (RREKEY).

When creating your DREAM Writer Version, enter 2 in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchal totaling and page breaks in the Data Sequence.

This program type chains to the secondary master file in subroutine S003. Update the key field or key list prior to the chain. Key fields do not automatically update in this program type.

This program type includes subroutine S005 for all calculations. Add all special logic necessary between the read of the primary master file and the update of the secondary master file.

This program updates the master file records in subroutine S010.

C.23.6 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.

C.24 Y0020 - Conversion, One File Update with Error Report

C.24.1 Description

Use this program type to create a batch conversion program that you manipulate using DREAM Writer. This program type processes a single master file by key. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters. You do not use Lockout Action Codes, user defined options, and function exits.
C.24.2 Printer File Definition
This program type requires that formats HEADING1, DETAIL1, and ERROR1 exist in the printer file. Format TOTAL1 is optional for totals.

C.24.3 CL Program Definition
Copy and revise the model CL program J98MODEL6 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.24.4 File Specifications
This program type requires that you define a key for a single master file. The master file contains 1 in the Update column. This program type does not update the master file. Clear the X from the Add column, if necessary. The printer file begins with R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.24.5 Special Considerations
If you are printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).
When printing the error report, format ERROR1 must contain the special fields for error message description (RRDSCA) and error message number (RREKEY).
When creating your DREAM Writer Version, enter 2 in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchal totaling and page breaks in the Data Sequence.
This program type uses subroutine S005 for all calculations. Add special logic necessary between the read of the primary master file and the update of the master file.
This program updates the master file records in subroutine S010.

C.24.6 Quick Start Generation
You cannot generate this program type using the Quick Start CL Generator.

C.25 Y0030 - Conversion, One File Write with Error Report

C.25.1 Description
Use this program type to create a batch conversion program that you manipulate using DREAM Writer. This program type processes a single master file by key. You choose the data for the batch update program using DREAM Writer Data Selection and Data Sequence parameters. You do not use Lockout Action Codes, user defined options, and function exits.

C.25.2 Printer File Definition
This program type requires that formats HEADING1, DETAIL1, and ERROR1 exist in the printer file. Format TOTAL1 is optional for totals.
C.25.3 CL Program Definition

Copy and revise the model CL program J98MODEL6 to create a CL program for use with this program type. Use the Quick Start CL Generator to automatically create your CL program.

C.25.4 File Specifications

This program type requires that you define a key for a single master file. The master file contains 1 in the Update column. As this program type updates the master file, enter X in the Add column. The printer file begins with R and has blank selection columns. Add files to retrieve descriptions if necessary.

C.25.5 Special Considerations

If you are printing totals using format TOTAL1, use the special keywords for Data Dictionary description (VC1ROW), data key field (VC1KEY), and data key description (VC1DSC).

When printing the error report, format ERROR1 must contain the special fields for error message description (RRDSCA) and error message number (RREKEY).

When creating your DREAM Writer Version, enter 2 in the Type Report Totaling field on the Additional Parameters screen. This allows hierarchal totaling and page breaks in the Data Sequence.

This program type uses subroutine S005 for all calculations. Add special logic necessary between the read of the primary master file and the update of the master file.

This program updates the master file records in subroutine S010.

C.25.6 Quick Start Generation

You cannot generate this program type using the Quick Start CL Generator.
This appendix contains these topics:

- Section D.1, "Program Status Data Structure - I00DSPROG,"
- Section D.2, "Copy Module - Retrieve Soft Coding Data Structure - I00SC,"
- Section D.3, "Item Master Information - P928011."

D.1 Program Status Data Structure - I00DSPROG

Figure D–1  Program Status Data Structure Source Code (1 of 2)
Figure D–2  Program Status Data Structure Source Code (2 of 2)

Figure D–3  Copy Module - Retrieve Soft Coding Data Structure report (1 of 7)
Figure D–4  Copy Module - Retrieve Soft Coding Data Structure report (2 of 7)
Figure D–5  Copy Module - Retrieve Soft Coding Data Structure report (3 of 7)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>129.00</td>
<td>Date (dd.mm.yyyy)</td>
</tr>
<tr>
<td>140.00</td>
<td>Source code section name</td>
</tr>
<tr>
<td>141.00</td>
<td>File Information Data Structure for Panel/Report file</td>
</tr>
<tr>
<td>143.00</td>
<td>Reserved 12</td>
</tr>
<tr>
<td>144.00</td>
<td>Reserved 8</td>
</tr>
<tr>
<td>145.00</td>
<td>Internal program file name</td>
</tr>
<tr>
<td>146.00</td>
<td>All Data Files (6)</td>
</tr>
<tr>
<td>147.00</td>
<td>Open indication (0=NOT)</td>
</tr>
<tr>
<td>148.00</td>
<td>Set of File Indication (inset of file)</td>
</tr>
<tr>
<td>149.00</td>
<td>Status code (0=Success, 1=Error)</td>
</tr>
<tr>
<td>151.00</td>
<td>Operation code</td>
</tr>
<tr>
<td>152.00</td>
<td>Name of KSP routine exception/error occurred</td>
</tr>
<tr>
<td>153.00</td>
<td>RPS source statement sequence number</td>
</tr>
<tr>
<td>154.00</td>
<td>User-Specified reason for error on RPS file</td>
</tr>
<tr>
<td>156.00</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Data - 07.01.91
Figure D–6  Copy Module - Retrieve Soft Coding Data Structure report (4 of 7)

<table>
<thead>
<tr>
<th>Source Listings</th>
<th>D-5</th>
</tr>
</thead>
</table>

| 229.03 | * | TC = Count |
| 224.03 | * | TC = Engine |
| 220.03 | * | TC = Execution |
| 222.03 | * | TC = Execution |
| 223.03 | * | TC = Execution |
| 225.03 | * | TC = Execution |
| 226.03 | * | TC = Execution |
| 227.03 | * | TC = Execution |
| 228.03 | * | TC = Execution |
| 229.03 | * | TC = Execution |
| 230.03 | * | TC = Execution |
| 231.03 | * | TC = Execution |
| 232.03 | * | TC = Execution |
| 233.03 | * | TC = Execution |
| 234.03 | * | TC = Execution |
| 235.03 | * | TC = Execution |
| 236.03 | * | TC = Execution |
| 237.03 | * | TC = Execution |
| 238.03 | * | TC = Execution |
| 239.03 | * | TC = Execution |
| 240.03 | * | TC = Execution |
| 241.03 | * | TC = Execution |
| 242.03 | * | TC = Execution |
| 243.03 | * | TC = Execution |
| 244.03 | * | TC = Execution |
| 245.03 | * | TC = Execution |
| 246.03 | * | TC = Execution |
| 247.03 | * | TC = Execution |
| 248.03 | * | TC = Execution |
| 249.03 | * | TC = Execution |
| 250.03 | * | TC = Execution |
| 251.03 | * | TC = Execution |
| 252.03 | * | TC = Execution |
| 253.03 | * | TC = Execution |
| 254.03 | * | TC = Execution |
| 255.03 | * | TC = Execution |
| 256.03 | * | TC = Execution |
| 257.03 | * | TC = Execution |
| 258.03 | * | TC = Execution |
| 259.03 | * | TC = Execution |
| 260.03 | * | TC = Execution |
| 261.03 | * | TC = Execution |
| 262.03 | * | TC = Execution |
| 263.03 | * | TC = Execution |
| 264.03 | * | TC = Execution |
| 265.03 | * | TC = Execution |
| 266.03 | * | TC = Execution |
| 267.03 | * | TC = Execution |
| 268.03 | * | TC = Execution |
| 269.03 | * | TC = Execution |
| 270.03 | * | TC = Execution |
| 271.03 | * | TC = Execution |
| 272.03 | * | TC = Execution |
| 273.03 | * | TC = Execution |
| 274.03 | * | TC = Execution |
| 275.03 | * | TC = Execution |
| 276.03 | * | TC = Execution |
| 277.03 | * | TC = Execution |
| 278.03 | * | TC = Execution |
| 279.03 | * | TC = Execution |
| 280.03 | * | TC = Execution |
| 281.03 | * | TC = Execution |
| 282.03 | * | TC = Execution |
| 283.03 | * | TC = Execution |
| 284.03 | * | TC = Execution |
| 285.03 | * | TC = Execution |
| 286.03 | * | TC = Execution |
| 287.03 | * | TC = Execution |
| 288.03 | * | TC = Execution |
| 289.03 | * | TC = Execution |
| 290.03 | * | TC = Execution |
| 291.03 | * | TC = Execution |
| 292.03 | * | TC = Execution |
| 293.03 | * | TC = Execution |
| 294.03 | * | TC = Execution |
| 295.03 | * | TC = Execution |
| 296.03 | * | TC = Execution |
| 297.03 | * | TC = Execution |
| 298.03 | * | TC = Execution |
| 299.03 | * | TC = Execution |
| 300.03 | * | TC = Execution |
| 301.03 | * | TC = Execution |
| 302.03 | * | TC = Execution |
| 303.03 | * | TC = Execution |
| 304.03 | * | TC = Execution |
| 305.03 | * | TC = Execution |
| 306.03 | * | TC = Execution |
| 307.03 | * | TC = Execution |
| 308.03 | * | TC = Execution |
| 309.03 | * | TC = Execution |
| 310.03 | * | TC = Execution |
| 311.03 | * | TC = Execution |
| 312.03 | * | TC = Execution |
| 313.03 | * | TC = Execution |
Figure D–7  Copy Module - Retrieve Soft Coding Data Structure report (5 of 7)
Figure D–8 Copy Module - Retrieve Soft Coding Data Structure report (6 of 7)
### Figure D–9  Copy Module - Retrieve Soft Coding Data Structure report (7 of 7)

<table>
<thead>
<tr>
<th>Sec No</th>
<th>Mod Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>503.40</td>
<td>7.95.68</td>
</tr>
<tr>
<td>503.50</td>
<td>7.95.68</td>
</tr>
<tr>
<td>503.60</td>
<td>7.95.68</td>
</tr>
<tr>
<td>503.70</td>
<td>7.95.68</td>
</tr>
<tr>
<td>503.80</td>
<td>7.95.68</td>
</tr>
<tr>
<td>503.90</td>
<td>7.95.68</td>
</tr>
<tr>
<td>504.00</td>
<td>7.95.68</td>
</tr>
</tbody>
</table>

**Print Source Code Data - 27.03.87**

**Section Fields for Subfile Mode and Cursor Position**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subfile Mode</td>
<td></td>
</tr>
</tbody>
</table>
D.3 Item Master Information - P928011

Figure D–10 Item Master Information report (1 of 32)
### Figure D–11  Item Master Information report (2 of 32)

The image contains a page from the JD Edwards World Computer Aided Software Engineering Guide, focusing on the Item Master Information report (2 of 32). The page includes a table with columns and rows, and a diagram illustrating data structures and program status data structure.

#### Table

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Data Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>Data structure for commonly used indexes</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>Data structure with file servers</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>Program status data structure</td>
</tr>
</tbody>
</table>

#### Diagram

The diagram illustrates each VTX field is 40 long but may not use all 40. It includes a vocabulary overrides reference.
Figure D–12  Item Master Information report (3 of 32)
Figure D–13  Item Master Information report (4 of 32)
Figure D–14  Item Master Information report (5 of 32)
### Figure D–15  Item Master Information report (6 of 32)

<table>
<thead>
<tr>
<th>Line</th>
<th>Source</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>384.00</td>
<td>OB1</td>
<td>END</td>
</tr>
<tr>
<td>385.00</td>
<td>C*</td>
<td>IF ROLL DOWN key pressed, process read prior.</td>
</tr>
<tr>
<td>386.00</td>
<td>C*</td>
<td>-----</td>
</tr>
<tr>
<td>387.00</td>
<td>C*</td>
<td>-----</td>
</tr>
<tr>
<td>388.00</td>
<td>C*</td>
<td>-----</td>
</tr>
<tr>
<td>389.00</td>
<td>CER</td>
<td>EXITED</td>
</tr>
<tr>
<td>390.00</td>
<td>CER</td>
<td>MOVIE 'P0' 'IN'.46</td>
</tr>
<tr>
<td>391.00</td>
<td>CER</td>
<td>MOVIE 'P1' 'IN'.46</td>
</tr>
<tr>
<td>392.00</td>
<td>CER</td>
<td>MOVIE 'P2' 'IN'.46</td>
</tr>
<tr>
<td>393.00</td>
<td>CER</td>
<td>MOVIE 'P3' 'IN'.46</td>
</tr>
<tr>
<td>394.00</td>
<td>CER</td>
<td>MOVIE 'P4' 'IN'.46</td>
</tr>
<tr>
<td>395.00</td>
<td>CER</td>
<td>MOVIE 'P5' 'IN'.46</td>
</tr>
<tr>
<td>396.00</td>
<td>CER</td>
<td>MOVIE 'P6' 'IN'.46</td>
</tr>
<tr>
<td>397.00</td>
<td>CER</td>
<td>MOVIE 'P7' 'IN'.46</td>
</tr>
<tr>
<td>398.00</td>
<td>CER</td>
<td>MOVIE 'P8' 'IN'.46</td>
</tr>
<tr>
<td>399.00</td>
<td>CER</td>
<td>MOVIE 'P9' 'IN'.46</td>
</tr>
<tr>
<td>400.00</td>
<td>CER</td>
<td>MOVIE 'P10' 'IN'.46</td>
</tr>
<tr>
<td>401.00</td>
<td>CER</td>
<td>MOVIE 'P11' 'IN'.46</td>
</tr>
<tr>
<td>402.00</td>
<td>CER</td>
<td>MOVIE 'P12' 'IN'.46</td>
</tr>
<tr>
<td>403.00</td>
<td>CER</td>
<td>MOVIE 'P13' 'IN'.46</td>
</tr>
<tr>
<td>404.00</td>
<td>CER</td>
<td>MOVIE 'P14' 'IN'.46</td>
</tr>
<tr>
<td>405.00</td>
<td>CER</td>
<td>MOVIE 'P15' 'IN'.46</td>
</tr>
<tr>
<td>406.00</td>
<td>CER</td>
<td>MOVIE 'P16' 'IN'.46</td>
</tr>
<tr>
<td>407.00</td>
<td>CER</td>
<td>MOVIE 'P17' 'IN'.46</td>
</tr>
<tr>
<td>408.00</td>
<td>CER</td>
<td>MOVIE 'P18' 'IN'.46</td>
</tr>
<tr>
<td>409.00</td>
<td>CER</td>
<td>MOVIE 'P19' 'IN'.46</td>
</tr>
<tr>
<td>410.00</td>
<td>CER</td>
<td>MOVIE 'P20' 'IN'.46</td>
</tr>
<tr>
<td>411.00</td>
<td>CER</td>
<td>MOVIE 'P21' 'IN'.46</td>
</tr>
<tr>
<td>412.00</td>
<td>CER</td>
<td>MOVIE 'P22' 'IN'.46</td>
</tr>
<tr>
<td>413.00</td>
<td>CER</td>
<td>MOVIE 'P23' 'IN'.46</td>
</tr>
<tr>
<td>414.00</td>
<td>CER</td>
<td>MOVIE 'P24' 'IN'.46</td>
</tr>
<tr>
<td>415.00</td>
<td>CER</td>
<td>MOVIE 'P25' 'IN'.46</td>
</tr>
<tr>
<td>416.00</td>
<td>CER</td>
<td>MOVIE 'P26' 'IN'.46</td>
</tr>
<tr>
<td>417.00</td>
<td>CER</td>
<td>MOVIE 'P27' 'IN'.46</td>
</tr>
<tr>
<td>418.00</td>
<td>CER</td>
<td>MOVIE 'P28' 'IN'.46</td>
</tr>
<tr>
<td>419.00</td>
<td>CER</td>
<td>MOVIE 'P29' 'IN'.46</td>
</tr>
<tr>
<td>420.00</td>
<td>CER</td>
<td>MOVIE 'P30' 'IN'.46</td>
</tr>
<tr>
<td>421.00</td>
<td>CER</td>
<td>MOVIE 'P31' 'IN'.46</td>
</tr>
<tr>
<td>422.00</td>
<td>CER</td>
<td>MOVIE 'P32' 'IN'.46</td>
</tr>
<tr>
<td>423.00</td>
<td>CER</td>
<td>MOVIE 'P33' 'IN'.46</td>
</tr>
<tr>
<td>424.00</td>
<td>CER</td>
<td>MOVIE 'P34' 'IN'.46</td>
</tr>
<tr>
<td>425.00</td>
<td>CER</td>
<td>MOVIE 'P35' 'IN'.46</td>
</tr>
<tr>
<td>426.00</td>
<td>CER</td>
<td>MOVIE 'P36' 'IN'.46</td>
</tr>
<tr>
<td>427.00</td>
<td>CER</td>
<td>MOVIE 'P37' 'IN'.46</td>
</tr>
<tr>
<td>428.00</td>
<td>CER</td>
<td>MOVIE 'P38' 'IN'.46</td>
</tr>
<tr>
<td>429.00</td>
<td>CER</td>
<td>MOVIE 'P39' 'IN'.46</td>
</tr>
<tr>
<td>430.00</td>
<td>CER</td>
<td>MOVIE 'P40' 'IN'.46</td>
</tr>
<tr>
<td>431.00</td>
<td>CER</td>
<td>MOVIE 'P41' 'IN'.46</td>
</tr>
<tr>
<td>432.00</td>
<td>CER</td>
<td>MOVIE 'P42' 'IN'.46</td>
</tr>
<tr>
<td>433.00</td>
<td>CER</td>
<td>MOVIE 'P43' 'IN'.46</td>
</tr>
<tr>
<td>434.00</td>
<td>CER</td>
<td>MOVIE 'P44' 'IN'.46</td>
</tr>
<tr>
<td>435.00</td>
<td>CER</td>
<td>MOVIE 'P45' 'IN'.46</td>
</tr>
<tr>
<td>436.00</td>
<td>CER</td>
<td>MOVIE 'P46' 'IN'.46</td>
</tr>
<tr>
<td>437.00</td>
<td>CER</td>
<td>MOVIE 'P47' 'IN'.46</td>
</tr>
<tr>
<td>438.00</td>
<td>CER</td>
<td>MOVIE 'P48' 'IN'.46</td>
</tr>
<tr>
<td>439.00</td>
<td>CER</td>
<td>MOVIE 'P49' 'IN'.46</td>
</tr>
<tr>
<td>440.00</td>
<td>CER</td>
<td>MOVIE 'P50' 'IN'.46</td>
</tr>
<tr>
<td>441.00</td>
<td>CER</td>
<td>MOVIE 'P51' 'IN'.46</td>
</tr>
<tr>
<td>442.00</td>
<td>CER</td>
<td>MOVIE 'P52' 'IN'.46</td>
</tr>
<tr>
<td>443.00</td>
<td>CER</td>
<td>MOVIE 'P53' 'IN'.46</td>
</tr>
<tr>
<td>444.00</td>
<td>CER</td>
<td>MOVIE 'P54' 'IN'.46</td>
</tr>
<tr>
<td>445.00</td>
<td>CER</td>
<td>MOVIE 'P55' 'IN'.46</td>
</tr>
<tr>
<td>446.00</td>
<td>CER</td>
<td>MOVIE 'P56' 'IN'.46</td>
</tr>
<tr>
<td>447.00</td>
<td>CER</td>
<td>MOVIE 'P57' 'IN'.46</td>
</tr>
<tr>
<td>448.00</td>
<td>CER</td>
<td>MOVIE 'P58' 'IN'.46</td>
</tr>
<tr>
<td>449.00</td>
<td>CER</td>
<td>MOVIE 'P59' 'IN'.46</td>
</tr>
<tr>
<td>450.00</td>
<td>CER</td>
<td>MOVIE 'P60' 'IN'.46</td>
</tr>
<tr>
<td>451.00</td>
<td>CER</td>
<td>MOVIE 'P61' 'IN'.46</td>
</tr>
<tr>
<td>452.00</td>
<td>CER</td>
<td>MOVIE 'P62' 'IN'.46</td>
</tr>
<tr>
<td>453.00</td>
<td>CER</td>
<td>MOVIE 'P63' 'IN'.46</td>
</tr>
<tr>
<td>454.00</td>
<td>CER</td>
<td>MOVIE 'P64' 'IN'.46</td>
</tr>
<tr>
<td>455.00</td>
<td>CER</td>
<td>MOVIE 'P65' 'IN'.46</td>
</tr>
<tr>
<td>456.00</td>
<td>CER</td>
<td>MOVIE 'P66' 'IN'.46</td>
</tr>
<tr>
<td>457.00</td>
<td>CER</td>
<td>MOVIE 'P67' 'IN'.46</td>
</tr>
<tr>
<td>458.00</td>
<td>CER</td>
<td>MOVIE 'P68' 'IN'.46</td>
</tr>
<tr>
<td>459.00</td>
<td>CER</td>
<td>MOVIE 'P69' 'IN'.46</td>
</tr>
<tr>
<td>460.00</td>
<td>CER</td>
<td>MOVIE 'P70' 'IN'.46</td>
</tr>
<tr>
<td>461.00</td>
<td>CER</td>
<td>MOVIE 'P71' 'IN'.46</td>
</tr>
<tr>
<td>462.00</td>
<td>CER</td>
<td>MOVIE 'P72' 'IN'.46</td>
</tr>
<tr>
<td>463.00</td>
<td>CER</td>
<td>MOVIE 'P73' 'IN'.46</td>
</tr>
<tr>
<td>464.00</td>
<td>CER</td>
<td>MOVIE 'P74' 'IN'.46</td>
</tr>
<tr>
<td>465.00</td>
<td>CER</td>
<td>MOVIE 'P75' 'IN'.46</td>
</tr>
<tr>
<td>466.00</td>
<td>CER</td>
<td>MOVIE 'P76' 'IN'.46</td>
</tr>
<tr>
<td>467.00</td>
<td>CER</td>
<td>MOVIE 'P77' 'IN'.46</td>
</tr>
<tr>
<td>468.00</td>
<td>CER</td>
<td>MOVIE 'P78' 'IN'.46</td>
</tr>
<tr>
<td>469.00</td>
<td>CER</td>
<td>MOVIE 'P79' 'IN'.46</td>
</tr>
<tr>
<td>470.00</td>
<td>CER</td>
<td>MOVIE 'P80' 'IN'.46</td>
</tr>
</tbody>
</table>

Program that will display a record lock window when a record in use error is encountered.
Figure D–16  Item Master Information report (7 of 32)
**Figure D-17  Item Master Information report (8 of 32)**

```
543.00  CER  END
544.00  C*  
545.00  CER  ###FLM  1FREQ 'VX402' 
544.00  CER  MODE&GLOBAL  VED002
547.00  CER  GO TO END2
548.00  C*  
549.00  CER  END
550.00  C*  
551.00  CER  ###FLM  1FREQ 'VX403' 
552.00  CER  MODE&GLOBAL  VED003
553.00  CER  GO TO END2
554.00  C*  
555.00  CER  END
556.00  C*  
557.00  CER  ###FLM  1FREQ 'VX404' 
558.00  CER  MODE&GLOBAL  VED004
559.00  CER  GO TO END2
561.00  C*  
562.00  CER  END
563.00  C*  
564.00  CER  ###FLM  1FREQ 'VX405' 
565.00  CER  MODE&GLOBAL  VED005
566.00  C*  
568.00  CER  END
569.00  C*  
570.00  CER  END2  END
571.00  C*  
572.00  C*  SUBROTTING D001 - Clear Fields
574.00  C*  
575.00  C*  
576.00  C*  Processing: 1. Reset all video screen and data file fields
577.00  C*  for next transaction.
578.00  C*  
579.00  C*  2. Clear action code only if requested.
580.00  CER  S001  ENDER
581.00  C*  
582.00  C*  
583.00  C*  
584.00  C*  
586.00  CER  MOV  S1BLANKE  #SCLF 
587.00  CER  MOV  S1BLANKE  #SFRC
588.00  CER  MOV  S1BLANKE  #SFLC
589.00  CER  MOV  S1BLANKE  #SFLC
590.00  CER  MOV  S1BLANKE  VED01C
591.00  CER  MOV  S1BLANKE  VED02
592.00  CER  MOV  S1BLANKE  VED03
593.00  CER  MOV  S1BLANKE  VED04
594.00  CER  MOV  S1BLANKE  VED05
595.00  CER  MOV  S1BLANKE  VED06
596.00  CER  MOV  S1BLANKE  VED08
597.00  CER  MOV  S1BLANKE  VED02
598.00  CER  MOV  S1BLANKE  VED01
599.00  CER  MOV  S1BLANKE  VED001
600.00  CER  MOV  S1BLANKE  VED002
601.00  CER  MOV  S1BLANKE  VED003
602.00  CER  MOV  S1BLANKE  VED004
603.00  CER  MOUER  IVR14 
604.00  C*  
605.00  C*  
606.00  C*  
607.00  C*  
608.00  CER  MOV  'ALL'  'BLANKE'  S001 
609.00  CER  MOV  'ALL'  'IN 4L' 
610.00  CER  MOV  'ALL'  'ACTION' 
611.00  CER  MOV  'ALL'  'COMMAND'  S001 
612.00  CER  MOV  'ALL'  'COMMAND'  S002 
613.00  CER  MOV  'ALL'  'COMMAND'  S003 
614.00  CER  MOV  'ALL'  'COMMAND'  S004 
615.00  CER  MOV  'ALL'  'COMMAND'  S005 
616.00  CER  MOV  'ALL'  'COMMAND'  S006 
617.00  CER  MOV  'ALL'  'COMMAND'  S007 
618.00  CER  MOV  'ALL'  'COMMAND'  S008 
619.00  CER  MOV  'ALL'  'COMMAND'  S009 
620.00  CER  MOV  'ALL'  'COMMAND'  S010 
621.00  CER  MOV  'ALL'  'COMMAND'  S011 
622.00  CER  MOV  'ALL'  'COMMAND'  S012 
623.00  CER  END
624.00  C*  
```

These fields will only be cleared if the user presses the function key to clear the screen. We want to save certain information like key fields and descriptions of they don’t get cleared everyday S001 is executed.
**Figure D–19 Item Master Information report (10 of 32)**

```
781.00  CDR     MOVE    ' '  $$REPLCM  2
782.00  CDR     SETDD  RND2003
783.00  CDR     END
784.00  CDR
785.00  CDR
786.00  CDR     Edit result of read and action code.
787.00  CDR
788.00  CDR     *INQ99  IFREQ '1'
789.00  CDR     *INQ11  COMP 'O'
790.00  CDR     END
791.00  CDR
792.00  CDR
793.00  CDR
794.00  CDR     If indicator 41 on, invalid key for action code.
795.00  CDR
796.00  CDR     *INQ11  IFREQ '1'
797.00  CDR     *INQ12  COMP 'O'
798.00  CDR     END
799.00  CDR
800.00  CDR
801.00  CDR     If indicator 99 on, record in use.
802.00  CDR
803.00  CDR     *INQ99  IFREQ '1'
804.00  CDR     CALL 'PUBLOCK'  91
805.00  CDR
806.00  CDR
807.00  CDR     If not inquiry, skip remainder of subroutine.
808.00  CDR
809.00  CDR
810.00  CDR     *INQ4  CASKY 'O'  ENDD01
811.00  CDR
812.00  CDR
813.00  CDR     Release record lock on master file.
814.00  CDR     JDE user uses this or NOTNL.
815.00  CDR
816.00  CDR     to release record locks
817.00  CDR     ENDD01
818.00  CDR
819.00  CDR
820.00  CDR     If errors, skip remainder of subroutine.
821.00  CDR
822.00  CDR
823.00  CDR     ENDD01
824.00  CDR
825.00  CDR
826.00  CDR
827.00  CDR
828.00  CDR
829.00  CDR
830.00  CDR
831.00  CDR
832.00  CDR
833.00  CDR
834.00  CDR
835.00  CDR
836.00  CDR
837.00  CDR
838.00  CDR
839.00  CDR
840.00  CDR
841.00  CDR
842.00  CDR
843.00  CDR
844.00  CDR
845.00  CDR
846.00  CDR
847.00  CDR
848.00  CDR
849.00  CDR
850.00  CDR
851.00  CDR
852.00  CDR
853.00  CDR
854.00  CDR
855.00  CDR
856.00  CDR     ENDD01  ENDE1
857.00  CDR
858.00  CDR
859.00  CDR     Copy Common Subroutine – Right Justify Numeric Fields
860.00  CDR
861.00  CDR
862.00  CDR
863.00  CDR
864.00  CDR
865.00  CDR
866.00  CDR
867.00  CDR
868.00  CDR
869.00  CDR
870.00  CDR
871.00  CDR
872.00  CDR
873.00  CDR
874.00  CDR
875.00  CDR
876.00  CDR
877.00  CDR
```

D-18  JD Edwards World Computer Aided Software Engineering Guide
Figure D–21  Item Master Information report (12 of 32)
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>932.00</td>
<td>CSE MOVE QUANTITY  PRINT  6</td>
</tr>
<tr>
<td>933.00</td>
<td>CSE MOVE *BRAND  SENET  4</td>
</tr>
<tr>
<td>934.00</td>
<td>CSE MOVE *BRAND  *BRAND  7</td>
</tr>
<tr>
<td>935.00</td>
<td>CSE MOVE *BRAND  *BRAND  7</td>
</tr>
<tr>
<td>936.00</td>
<td>CSE MOVE *BRAND  *BRAND  7</td>
</tr>
<tr>
<td>937.00</td>
<td>CSE MOVE *BRAND  *BRAND  7</td>
</tr>
<tr>
<td>938.00</td>
<td>CSE CALL '20028'  61</td>
</tr>
<tr>
<td>939.00</td>
<td>CSE</td>
</tr>
<tr>
<td>940.00</td>
<td>CSE PARM SENET</td>
</tr>
<tr>
<td>941.00</td>
<td>CSE DRAM SENET</td>
</tr>
<tr>
<td>942.00</td>
<td>CSE DRAM SENET</td>
</tr>
<tr>
<td>943.00</td>
<td>CSE DRAM SENET</td>
</tr>
<tr>
<td>944.00</td>
<td>CSE DRAM SENET</td>
</tr>
<tr>
<td>945.00</td>
<td>CSE DRAM SENET</td>
</tr>
<tr>
<td>946.00</td>
<td>CSE MOVE *BRAND  VEDIT</td>
</tr>
<tr>
<td>947.00</td>
<td>C*</td>
</tr>
<tr>
<td>948.00</td>
<td>C*</td>
</tr>
<tr>
<td>949.00</td>
<td>C*</td>
</tr>
<tr>
<td>950.00</td>
<td>C*</td>
</tr>
<tr>
<td>951.00</td>
<td>C*</td>
</tr>
<tr>
<td>952.00</td>
<td>C*</td>
</tr>
<tr>
<td>953.00</td>
<td>C*</td>
</tr>
<tr>
<td>954.00</td>
<td>C*</td>
</tr>
<tr>
<td>955.00</td>
<td>C*</td>
</tr>
<tr>
<td>956.00</td>
<td>C*</td>
</tr>
<tr>
<td>957.00</td>
<td>C*</td>
</tr>
<tr>
<td>958.00</td>
<td>C*</td>
</tr>
<tr>
<td>959.00</td>
<td>C*</td>
</tr>
<tr>
<td>960.00</td>
<td>C*</td>
</tr>
<tr>
<td>961.00</td>
<td>C*</td>
</tr>
<tr>
<td>962.00</td>
<td>C*</td>
</tr>
<tr>
<td>963.00</td>
<td>C*</td>
</tr>
<tr>
<td>964.00</td>
<td>C*</td>
</tr>
<tr>
<td>965.00</td>
<td>C*</td>
</tr>
<tr>
<td>966.00</td>
<td>C*</td>
</tr>
<tr>
<td>967.00</td>
<td>C*</td>
</tr>
<tr>
<td>968.00</td>
<td>C*</td>
</tr>
<tr>
<td>969.00</td>
<td>C*</td>
</tr>
<tr>
<td>970.00</td>
<td>C*</td>
</tr>
<tr>
<td>971.00</td>
<td>C*</td>
</tr>
<tr>
<td>972.00</td>
<td>C*</td>
</tr>
<tr>
<td>973.00</td>
<td>C*</td>
</tr>
<tr>
<td>974.00</td>
<td>C*</td>
</tr>
<tr>
<td>975.00</td>
<td>C*</td>
</tr>
<tr>
<td>976.00</td>
<td>C*</td>
</tr>
<tr>
<td>977.00</td>
<td>C*</td>
</tr>
<tr>
<td>978.00</td>
<td>C*</td>
</tr>
<tr>
<td>979.00</td>
<td>C*</td>
</tr>
<tr>
<td>980.00</td>
<td>C*</td>
</tr>
<tr>
<td>981.00</td>
<td>C*</td>
</tr>
<tr>
<td>982.00</td>
<td>C*</td>
</tr>
<tr>
<td>983.00</td>
<td>C*</td>
</tr>
<tr>
<td>984.00</td>
<td>C*</td>
</tr>
<tr>
<td>985.00</td>
<td>C*</td>
</tr>
<tr>
<td>986.00</td>
<td>C*</td>
</tr>
<tr>
<td>987.00</td>
<td>C*</td>
</tr>
<tr>
<td>988.00</td>
<td>C*</td>
</tr>
<tr>
<td>989.00</td>
<td>C*</td>
</tr>
<tr>
<td>990.00</td>
<td>C*</td>
</tr>
<tr>
<td>991.00</td>
<td>C*</td>
</tr>
<tr>
<td>992.00</td>
<td>C*</td>
</tr>
<tr>
<td>993.00</td>
<td>C*</td>
</tr>
<tr>
<td>994.00</td>
<td>C*</td>
</tr>
<tr>
<td>995.00</td>
<td>C*</td>
</tr>
<tr>
<td>996.00</td>
<td>C*</td>
</tr>
<tr>
<td>997.00</td>
<td>C*</td>
</tr>
<tr>
<td>998.00</td>
<td>C*</td>
</tr>
<tr>
<td>999.00</td>
<td>C*</td>
</tr>
<tr>
<td>1000.00</td>
<td>C*</td>
</tr>
<tr>
<td>1001.00</td>
<td>C*</td>
</tr>
<tr>
<td>1002.00</td>
<td>C*</td>
</tr>
<tr>
<td>1003.00</td>
<td>C*</td>
</tr>
<tr>
<td>1004.00</td>
<td>C*</td>
</tr>
<tr>
<td>1005.00</td>
<td>C*</td>
</tr>
<tr>
<td>1006.00</td>
<td>C*</td>
</tr>
<tr>
<td>1007.00</td>
<td>C*</td>
</tr>
<tr>
<td>1008.00</td>
<td>C*</td>
</tr>
<tr>
<td>1009.00</td>
<td>C*</td>
</tr>
<tr>
<td>1010.00</td>
<td>C*</td>
</tr>
</tbody>
</table>

External program used to edit dates.
Figure D–23  Item Master Information report (14 of 32)
Figure D–25  Item Master Information report (16 of 32)
Figure D–26  Item Master Information report (17 of 32)
### Figure D–27  Item Master Information report (18 of 32)

```plaintext
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1110</td>
<td>CER</td>
<td>.440.1</td>
</tr>
<tr>
<td>1120</td>
<td>CER</td>
<td>D-REDU</td>
</tr>
<tr>
<td>1122</td>
<td>CER</td>
<td>.EM</td>
</tr>
<tr>
<td>1124</td>
<td>CER</td>
<td>.440,8M</td>
</tr>
<tr>
<td>1128</td>
<td>CER</td>
<td>MOVE</td>
</tr>
<tr>
<td>1129</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1132</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1133</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1134</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1135</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1136</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1137</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1138</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1139</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1140</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1141</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1142</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1143</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1144</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1145</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1146</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1147</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1148</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1149</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1150</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1151</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1152</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1153</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1154</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1155</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1156</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1157</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1158</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1159</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1160</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1161</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1162</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1163</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1164</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1165</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1166</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1167</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1168</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1169</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1170</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1171</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1172</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1173</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1174</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1175</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1176</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1177</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1178</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1179</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1180</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1181</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1182</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1183</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1184</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1185</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1186</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1187</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1188</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1189</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1190</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1191</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1192</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1193</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1194</td>
<td>CER</td>
<td>END</td>
</tr>
<tr>
<td>1195</td>
<td>CER</td>
<td>END</td>
</tr>
</tbody>
</table>
```

---

**D-26  JD Edwards World Computer Aided Software Engineering Guide**
Figure D–28  Item Master Information report (19 of 32)
Figure D–29  Item Master Information report (20 of 32)
Figure D–30  Item Master Information report (21 of 32)

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1550</td>
<td>CBR</td>
<td>JSTRN</td>
</tr>
<tr>
<td>1551</td>
<td>CBR</td>
<td>$ERR (PFR)</td>
</tr>
<tr>
<td>1552</td>
<td>CBR</td>
<td>MOV '1'</td>
</tr>
<tr>
<td>1553</td>
<td>CBR</td>
<td>GONN       4993</td>
</tr>
<tr>
<td>1554</td>
<td>CBR</td>
<td>END</td>
</tr>
<tr>
<td>1555</td>
<td>CBR</td>
<td>END</td>
</tr>
<tr>
<td>1556</td>
<td>CBR</td>
<td>END</td>
</tr>
<tr>
<td>1557</td>
<td>CBR</td>
<td>END</td>
</tr>
<tr>
<td>1558</td>
<td>C*</td>
<td>ScrC and edit - Item Category Code 002</td>
</tr>
<tr>
<td>1559</td>
<td>C*</td>
<td>FerMB</td>
</tr>
<tr>
<td>1560</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1561</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1562</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1563</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1564</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1565</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1566</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1567</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1568</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1569</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1570</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1571</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1572</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1573</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1574</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1575</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1576</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1577</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1578</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1579</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1580</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1581</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1582</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1583</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1584</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1585</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1586</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1587</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1588</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1589</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1590</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1591</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1592</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1593</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1594</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1595</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1596</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1597</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1598</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1599</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1600</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1601</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1602</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1603</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1604</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1605</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1606</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1607</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1608</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1609</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1610</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1611</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1612</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1613</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1614</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1615</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1616</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1617</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1618</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1619</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1620</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1621</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1622</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1623</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1624</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1625</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
<tr>
<td>1626</td>
<td>C*</td>
<td>MOV 'L'.</td>
</tr>
</tbody>
</table>

Source Listings  D-29
### Figure D–31  Item Master Information report (22 of 32)

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1227</td>
<td>CCR</td>
<td>G0405U</td>
<td></td>
</tr>
<tr>
<td>1228</td>
<td>CCR</td>
<td>MOVEF00002</td>
<td>EUN</td>
</tr>
<tr>
<td>1229</td>
<td>CCR</td>
<td>MOVE A00002</td>
<td>EUN</td>
</tr>
<tr>
<td>1230</td>
<td>CCR</td>
<td>MOVE Q00003</td>
<td>ORMT</td>
</tr>
<tr>
<td>1231</td>
<td>CCR</td>
<td>CALL 'S0495'</td>
<td></td>
</tr>
<tr>
<td>1232</td>
<td>CCR</td>
<td>−−−−−−−−−−−−−−</td>
<td></td>
</tr>
<tr>
<td>1233</td>
<td>CCR</td>
<td>F70005</td>
<td></td>
</tr>
<tr>
<td>1234</td>
<td>CCR</td>
<td>$FREQ '1'</td>
<td></td>
</tr>
<tr>
<td>1235</td>
<td>CCR</td>
<td>MOVE '4'</td>
<td></td>
</tr>
<tr>
<td>1236</td>
<td>CCR</td>
<td>SETCOM</td>
<td></td>
</tr>
<tr>
<td>1237</td>
<td>CCR</td>
<td>END</td>
<td></td>
</tr>
<tr>
<td>1238</td>
<td>CCR</td>
<td>END</td>
<td></td>
</tr>
<tr>
<td>1239</td>
<td>CCR</td>
<td>−−−−−−−−−−−−−−</td>
<td></td>
</tr>
<tr>
<td>1240</td>
<td>CCR</td>
<td>−−−−−−−−−−−−−−</td>
<td></td>
</tr>
<tr>
<td>1241</td>
<td>CCR</td>
<td>Scrub and edit - Item Category Code 003</td>
<td></td>
</tr>
<tr>
<td>1242</td>
<td>CCR</td>
<td>MOVEL00003</td>
<td>Q0003</td>
</tr>
<tr>
<td>1243</td>
<td>CCR</td>
<td>Set default value - Item Category Code 003</td>
<td></td>
</tr>
<tr>
<td>1244</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1245</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1246</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1247</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1248</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1249</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1250</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1251</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1252</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1253</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1254</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1255</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1256</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1257</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1258</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1259</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1260</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1261</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1262</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1263</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1264</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1265</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1266</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1267</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1268</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1269</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1270</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1271</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1272</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1273</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1274</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1275</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1276</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1277</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1278</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1279</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1280</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1281</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1282</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1283</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1284</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1285</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1286</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1287</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1288</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1289</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1290</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1291</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1292</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1293</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1294</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1295</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1296</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1297</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1298</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1299</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1300</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1301</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1302</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
<tr>
<td>1303</td>
<td>CCR</td>
<td>$Q0003</td>
<td></td>
</tr>
</tbody>
</table>
Figure D–32  Item Master Information Report (23 of 32)
Figure D–33  Item Master Information report (24 of 32)

```
1781.00  CER  QIX004  AMEMX004  $EDIT
1782.00  CER  MOVE ' ' $EDIT
1783.00  CER  END
1784.00  CER  $SAMPLE  IFREQ ' 1'  $EDIT
1785.00  CER  MOVE ' 1'  $EDIT
1786.00  CER  SETOR  5990
1787.00  CER  END
1788.00  CER  END
1789.00  C*  Edit from User Defined Codes - Item Category Code 004
1790.00  C*  -----------------------------
1791.00  C*  ---------------------------------
1792.00  CER  $CHECK  IFREQ ' 1'  $EDIT
1793.00  CER  MOVE QIX004  $EDIT
1794.00  CER  CALL '80805'  $EDIT
1795.00  CER  CALL  '80805'  $EDIT
1796.00  CER  END
1797.00  CER  END
1798.00  CER  END
1799.00  CER  END
1800.00  CER  END
1801.00  CER  END
1802.00  CER  END
1803.00  CER  END
1804.00  CER  END
1805.00  C*  ---------------------------------
1806.00  C*  ---------------------------------
1807.00  C*  Set default value - Item Category Code 008
1808.00  C*  -----------------------------
1809.00  CER  MOVEQ3000  QXI005
1810.00  C*  ---------------------------------
1811.00  C*  ---------------------------------
1812.00  C*  ---------------------------------
1813.00  CER  QXI005  IFREQ ' 8BLANK'
1814.00  CER  QXI005  IFREQ ' 8BLANK'
1815.00  CER  QXI005  IFREQ ' 8BLANK'
1816.00  CER  QXI005  IFREQ ' 8BLANK'
1817.00  CER  QXI005  IFREQ ' 8BLANK'
1818.00  CER  QXI005  IFREQ ' 8BLANK'
1819.00  CER  QXI005  IFREQ ' 8BLANK'
1820.00  CER  QXI005  IFREQ ' 8BLANK'
1821.00  CER  QXI005  IFREQ ' 8BLANK'
1822.00  CER  QXI005  IFREQ ' 8BLANK'
1823.00  CER  QXI005  IFREQ ' 8BLANK'
1824.00  CER  QXI005  IFREQ ' 8BLANK'
1825.00  CER  QXI005  IFREQ ' 8BLANK'
1826.00  CER  QXI005  IFREQ ' 8BLANK'
1827.00  CER  QXI005  IFREQ ' 8BLANK'
1828.00  CER  QXI005  IFREQ ' 8BLANK'
1829.00  CER  QXI005  IFREQ ' 8BLANK'
1830.00  C*  ---------------------------------
1831.00  C*  Edit allowed values - Item Category Code 008
1832.00  C*  ---------------------------------
1833.00  CER  QXI005  IFREQ ' 8BLANK'
1834.00  CER  QXI005  IFREQ ' 8BLANK'
1835.00  CER  QXI005  IFREQ ' 8BLANK'
1836.00  CER  QXI005  IFREQ ' 8BLANK'
1837.00  CER  QXI005  IFREQ ' 8BLANK'
1838.00  CER  QXI005  IFREQ ' 8BLANK'
1839.00  CER  QXI005  IFREQ ' 8BLANK'
1840.00  CER  QXI005  IFREQ ' 8BLANK'
1841.00  CER  QXI005  IFREQ ' 8BLANK'
1842.00  CER  QXI005  IFREQ ' 8BLANK'
1843.00  CER  QXI005  IFREQ ' 8BLANK'
1844.00  CER  QXI005  IFREQ ' 8BLANK'
1845.00  CER  QXI005  IFREQ ' 8BLANK'
1846.00  CER  QXI005  IFREQ ' 8BLANK'
1847.00  CER  QXI005  IFREQ ' 8BLANK'
1848.00  CER  QXI005  IFREQ ' 8BLANK'
1849.00  CER  QXI005  IFREQ ' 8BLANK'
1850.00  CER  QXI005  IFREQ ' 8BLANK'
1851.00  CER  QXI005  IFREQ ' 8BLANK'
1852.00  CER  QXI005  IFREQ ' 8BLANK'
1853.00  CER  QXI005  IFREQ ' 8BLANK'
1854.00  CER  QXI005  IFREQ ' 8BLANK'
1855.00  CER  QXI005  IFREQ ' 8BLANK'
1856.00  CER  QXI005  IFREQ ' 8BLANK'
1857.00  CER  QXI005  IFREQ ' 8BLANK'
```
Figure D–34  Item Master Information report (25 of 32)
Figure D–35  Item Master Information report (26 of 32)
**Figure D–36  Item Master Information report (27 of 32)**

```
2066.00 C* Dictionary parameters for - Date Last Ship
2067.00 C*
2068.00 CRR MOVE *BLANK PRETAX
2069.00 CRR MOVE *BLANK PRETAX $1
2070.00 CRR CALL "$9999999"
2071.00 C* D-----
2072.00 CRR FRRM "D-35"
2073.00 IFRG "D-35"
2074.00 CRR MOVE PRETAX EXIST 10
2075.00 CRR MOVE PRETAX EXIST 1
2076.00 CRR MOVE PRETAX EXIST 50
2077.00 CRR MOVE PRETAX EXIST 20
2078.00 CRR MOVE PRETAX EXIST 1
2079.00 CRR MOVE PRETAX EXIST 4
2080.00 CRR MOVE PRETAX EXIST 2
2081.00 CRR MOVE PRETAX EXIST 40
2082.00 CRR MOVE PRETAX EXIST 40
2083.00 CRR MOVE PRETAX EXIST 1
2084.00 CRR MOVE PRETAX EXIST 20
2085.00 CRR MOVE PRETAX EXIST 110
2086.00 CRR MOVE PRETAX EXIST 2
2087.00 CRR DO BA
2088.00 CRR END
2089.00 CRR END
2090.00 C* D-----
2091.00 C* Dictionary parameters for - Item ID
2092.00 C*
2093.00 CRR MOVE *BLANK PRETAX
2094.00 CRR MOVE *BLANK PRETAX $1
2095.00 CRR CALL "$9999999"
2096.00 C*
2097.00 CRR FRRM "D-35"
2098.00 IFRG "D-35"
2099.00 CRR MOVE PRETAX EXIST 10
2000.00 CRR MOVE PRETAX EXIST 1
2001.00 CRR MOVE PRETAX EXIST 50
2002.00 CRR MOVE PRETAX EXIST 20
2003.00 CRR MOVE PRETAX EXIST 1
2004.00 CRR MOVE PRETAX EXIST 4
2005.00 CRR MOVE PRETAX EXIST 2
2006.00 CRR MOVE PRETAX EXIST 40
2007.00 CRR MOVE PRETAX EXIST 40
2008.00 CRR MOVE PRETAX EXIST 1
2009.00 CRR MOVE PRETAX EXIST 20
2010.00 CRR MOVE PRETAX EXIST 110
2011.00 CRR MOVE PRETAX EXIST 2
2012.00 CRR DO BA
2013.00 CRR END
2014.00 CRR END
2015.00 C* D-----
2016.00 C* Dictionary parameters for - quantity On Hand
2017.00 C*
2018.00 CRR MOVE *BLANK PRETAX
2019.00 CRR MOVE *BLANK PRETAX $1
2020.00 CRR CALL "$9999999"
2021.00 C*
2022.00 CRR FRRM "D-35"
2023.00 IFRG "D-35"
2024.00 CRR MOVE PRETAX EXIST 10
2025.00 CRR MOVE PRETAX EXIST 1
2026.00 CRR MOVE PRETAX EXIST 50
2027.00 CRR MOVE PRETAX EXIST 20
2028.00 CRR MOVE PRETAX EXIST 1
2029.00 CRR MOVE PRETAX EXIST 4
2030.00 CRR MOVE PRETAX EXIST 2
2031.00 CRR MOVE PRETAX EXIST 40
2032.00 CRR MOVE PRETAX EXIST 40
2033.00 CRR MOVE PRETAX EXIST 1
2034.00 CRR MOVE PRETAX EXIST 20
2035.00 CRR MOVE PRETAX EXIST 110
2036.00 CRR MOVE PRETAX EXIST 2
2037.00 CRR DO BA
2038.00 CRR END
2039.00 CRR END
2040.00 C* D-----
2041.00 C* Dictionary parameters for - quantity On Hand
2042.00 C*
2043.00 CRR MOVE *BLANK PRETAX
2044.00 CRR MOVE *BLANK PRETAX $1
2045.00 CRR CALL "$9999999"
2046.00 C*
2047.00 CRR FRRM "D-35"
2048.00 IFRG "D-35"
2049.00 CRR MOVE PRETAX EXIST 10
2050.00 CRR MOVE PRETAX EXIST 1
2051.00 CRR MOVE PRETAX EXIST 50
2052.00 CRR MOVE PRETAX EXIST 20
2053.00 CRR MOVE PRETAX EXIST 1
2054.00 CRR MOVE PRETAX EXIST 4
2055.00 CRR MOVE PRETAX EXIST 2
2056.00 CRR MOVE PRETAX EXIST 40
2057.00 CRR MOVE PRETAX EXIST 40
2058.00 CRR MOVE PRETAX EXIST 1
2059.00 CRR MOVE PRETAX EXIST 20
2060.00 CRR MOVE PRETAX EXIST 110
2061.00 CRR MOVE PRETAX EXIST 2
2062.00 CRR DO BA
2063.00 CRR END
2064.00 CRR END
2065.00 C* D-----
2066.00 C* Dictionary parameters for - quantity On Hand
2067.00 C*
2068.00 CRR MOVE *BLANK PRETAX
2069.00 CRR MOVE *BLANK PRETAX $1
2070.00 CRR CALL "$9999999"
2071.00 C*
2072.00 CRR FRRM "D-35"
2073.00 IFRG "D-35"
2074.00 CRR MOVE PRETAX EXIST 10
2075.00 CRR MOVE PRETAX EXIST 1
2076.00 CRR MOVE PRETAX EXIST 50
2077.00 CRR MOVE PRETAX EXIST 1
2078.00 CRR MOVE PRETAX EXIST 50
2079.00 CRR MOVE PRETAX EXIST 20
2080.00 CRR MOVE PRETAX EXIST 1
2081.00 CRR MOVE PRETAX EXIST 4
2082.00 CRR MOVE PRETAX EXIST 2
```
Figure D–37  Item Master Information report (28 of 32)

```
2003.00  CER  MOVE  PRET  0001  2
2004.00  CER  MOVE  PREVAL  0001  40
2005.00  CER  MOVE  PVAL  0001  40
2006.00  CER  MOVE  PRIVAL  0001  40
2007.00  CER  MOVE  PRIVIL  0001  40
2008.00  CER  MOVE  PReH  0001  30
2009.00  CER  MOVE  PRK  0001  1
2009.00  CER  MOVE  PRN  0001  10
2010.00  CER  S-JGSD  0001  110
2020.00  CER  MOVE  INQUT  0001  8
2030.00  CER  DO  0001  EA
2034.00  CER  MELT  0001  EA
2035.00  CER  END
2036.00  CER  END
2037.00  C*-----------------------
2039.00  C*  Dictionary parameters for - Item Type
2100.00  C*-----------------------
2101.00  CER  MOVE  "BLANK"  0001  PRETAL
2102.00  CER  MOVE  "NT"  0001  PRETAL
2103.00  CER  CALL  '265000E'  01
2104.00  C*-----------------------
2105.00  CER  FREQS  0001
2106.00  CER  FREQ  0001
2107.00  CER  MOVE  "G"  0001  DATA  20
2108.00  CER  MOVE  "S"  0001  DATA  20
2109.00  CER  MOVE  "N"  0001  DATA  20
2110.00  CER  MOVE  "H"  0001  DATA  20
2111.00  CER  MOVE  "K"  0001  DATA  20
2112.00  CER  MOVE  "L"  0001  DATA  20
2113.00  CER  MOVE  "M"  0001  DATA  20
2114.00  CER  MOVE  "P"  0001  DATA  20
2115.00  CER  MOVE  "Q"  0001  DATA  20
2116.00  CER  MOVE  "R"  0001  DATA  20
2117.00  CER  MOVE  "T"  0001  DATA  20
2118.00  CER  MOVE  "U"  0001  DATA  20
2119.00  CER  MOVE  "V"  0001  DATA  20
2120.00  CER  MOVE  "W"  0001  DATA  20
2121.00  CER  MOVE  "X"  0001  DATA  20
2122.00  CER  MOVE  "Y"  0001  DATA  20
2123.00  CER  S-JGSD  0002  110
2124.00  CER  DO  0001  EA
2125.00  CER  MELT  0001  EA
2126.00  CER  END
2127.00  CER  END
2128.00  C*-----------------------
2129.00  C*  Dictionary parameters for - Item Unit of Measure
2130.00  C*-----------------------
2131.00  C*-----------------------
2132.00  CER  MOVE  "BLANK"  0001  PRETAL
2133.00  CER  MOVE  "NT"  0001  PRETAL
2134.00  CER  CALL  '265000E'  01
2135.00  C*-----------------------
2136.00  CER  FREQS  0001
2137.00  CER  FREQ  0001
2138.00  CER  MOVE  "G"  0001  DATA  20
2139.00  CER  MOVE  "S"  0001  DATA  20
2140.00  CER  MOVE  "N"  0001  DATA  20
2141.00  CER  MOVE  "H"  0001  DATA  20
2142.00  CER  MOVE  "K"  0001  DATA  20
2143.00  CER  MOVE  "L"  0001  DATA  20
2144.00  CER  MOVE  "M"  0001  DATA  20
2145.00  CER  MOVE  "P"  0001  DATA  20
2146.00  CER  MOVE  "Q"  0001  DATA  20
2147.00  CER  MOVE  "R"  0001  DATA  20
2148.00  CER  MOVE  "T"  0001  DATA  20
2149.00  CER  MOVE  "U"  0001  DATA  20
2150.00  CER  MOVE  "V"  0001  DATA  20
2151.00  CER  MOVE  "W"  0001  DATA  20
2152.00  CER  S-JGSD  0002  110
2153.00  CER  DO  0001  EA
2154.00  CER  MELT  0001  EA
2155.00  CER  END
2156.00  CER  END
2157.00  CER  END
2158.00  CER  END
2159.00  C*-----------------------
```
Figure D–38  Item Master Information report (29 of 32)
### Figure D–39  Item Master Information report (30 of 32)

```
2227.00 CER MOVE PFKY RAKD003  4
2228.00 CER MOVE PFKT RAKD003  2
2230.00 CER MOVE PFKUL RAKD003  40
2231.00 CER MOVE PFKV RAKD003  40
2241.00 CER MOVE PFLUL RAKD003  40
2242.00 CER MOVE PFLV RAKD003  40
2243.00 CER MOVE PFLUL RAKD003  40
2244.00 CER MOVE PFLV RAKD003  40
2245.00 CER MOVE PFLUL RAKD003  20
2246.00 CER MOVE PFLV RAKD003  110
2247.00 CER MOVE PAKD003 EA
2248.00 CER DO EA
2249.00 CER MOVE 10 RAKD003
2250.00 CER END
2251.00 CER END
2252.00 CER -------------------------------
2253.00 C*  Dictionary parameters for - Item Category Code 003
2254.00 C*  -------------------------------
2255.00 C*  -------------------------------
2256.00 CER MOVE 'BLANK' RAKD001
2257.00 CER MOVE '2004' RAKD001
2258.00 CER CALL 'H90002'  91
2259.00 C*  -------------------------------
2260.00 CER DFHM 199002
2261.00 CER FDNM 199002
2262.00 CER MOVE PFDGM RAKD004  40
2263.00 CER MOVE PFDAT RAKD004  1
2264.00 CER MOVE PFDUC RAKD004  1
2265.00 CER MOVE PFDUC RAKD004  50
2266.00 CER MOVE PFDUC RAKD004  50
2267.00 CER MOVE PFDUC RAKD004  1
2268.00 CER MOVE PFDUC RAKD004  4
2269.00 CER MOVE PFKT RAKD004  2
2270.00 CER MOVE PFKUL RAKD004  40
2271.00 CER MOVE PFKV RAKD004  40
2272.00 CER MOVE PFLUL RAKD004  40
2273.00 CER MOVE PFLV RAKD004  40
2274.00 CER MOVE PFKUL RAKD004  40
2275.00 CER MOVE PFLUL RAKD004  40
2276.00 CER MOVE PFLV RAKD004  40
2277.00 CER MOVE PAKD004 EA
2278.00 CER DO EA
2279.00 CER MOVE 10 RAKD004
2280.00 CER ENDC
2281.00 CER ENDC
2282.00 CER END
2283.00 C*  -------------------------------
2284.00 C*  Dictionary parameters for - Item Category Code 005
2285.00 C*  -------------------------------
2286.00 C*  -------------------------------
2287.00 CER MOVE 'BLANK' RAKD001
2288.00 CER MOVE '2004' RAKD001
2289.00 CER CALL 'H90002'  91
2290.00 C*  -------------------------------
2291.00 CER DFHM 199002
2292.00 CER FDNM 199002
2293.00 CER MOVE PFDGM RAKD005  40
2294.00 CER MOVE PFDAT RAKD005  1
2295.00 CER MOVE PFDUC RAKD005  1
2296.00 CER MOVE PFDUC RAKD005  50
2297.00 CER MOVE PFDUC RAKD005  50
2298.00 CER MOVE PFDUC RAKD005  1
2299.00 CER MOVE PFDUC RAKD005  4
2300.00 CER MOVE PFKT RAKD005  2
2301.00 CER MOVE PFKUL RAKD005  40
2302.00 CER MOVE PFKV RAKD005  40
2303.00 CER MOVE PFLUL RAKD005  40
2304.00 CER MOVE PFLV RAKD005  40
2305.00 CER MOVE PFKUL RAKD005  40
2306.00 CER MOVE PFLUL RAKD005  40
2307.00 CER MOVE PFLV RAKD005  40
2308.00 CER MOVE PAKD005 EA
2309.00 CER DO EA
2310.00 CER MOVE 10 RAKD005
2311.00 CER ENDC
2312.00 CER ENDC
2313.00 CER ENDC
```
Figure D–40  Item Master Information report (31 of 32)
Figure D–41  Item Master Information report (32 of 32)
This appendix contains these topics:

- Section E.1, "Subroutines."
- Section E.2, "Flows."

### E.1 Subroutines

Using subroutines:

- Allows for standard names to make program maintenance easier.
- Launch primarily from Mainline.

The following table includes internal RPG subroutines within JD Edwards World programs:

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S00EX</td>
<td>Processes all function exits. Choose a function exit and the system launches one of the following programs:</td>
</tr>
<tr>
<td></td>
<td>- Display Functions (F24) to launch P9601H</td>
</tr>
<tr>
<td></td>
<td>- Field Sensitive Help (F1) launches X96CCX. After X96CCX launches, the system launches subroutine S00VL.</td>
</tr>
<tr>
<td></td>
<td>- Display Error Message (F7) launches P0000E</td>
</tr>
<tr>
<td></td>
<td>- HELP launches P00HELP</td>
</tr>
<tr>
<td></td>
<td>- Clear Screen (F22) launches subroutine S001</td>
</tr>
<tr>
<td></td>
<td>- Launches all programs to process all user defined function keys</td>
</tr>
<tr>
<td>S00VL</td>
<td>Retrieves values with Field Level Help. After X96CCX launches, the system launches subroutine S00VL.</td>
</tr>
<tr>
<td>S00OP</td>
<td>Subfile Options.</td>
</tr>
<tr>
<td>S001</td>
<td>Clears all database and screen fields. This usually only clears key fields and VC0 fields if you choose Clear Screen (F22).</td>
</tr>
<tr>
<td>S002</td>
<td>Checks for level breaks for reports.</td>
</tr>
<tr>
<td></td>
<td>- Activates level break markers.</td>
</tr>
<tr>
<td></td>
<td>- Retrieves the total line description</td>
</tr>
<tr>
<td>Subroutine</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>S003</td>
<td>Validates the key fields. Launches subroutine S998 if the system invokes auto inquire. Sets the file pointer.</td>
</tr>
<tr>
<td></td>
<td>■ Performs a SETLL and CHAIN if the program is a single record maintenance program</td>
</tr>
<tr>
<td></td>
<td>■ Performs a SETLL for subfile programs</td>
</tr>
<tr>
<td></td>
<td>Launches subroutine S004 to load screen and report fields</td>
</tr>
<tr>
<td></td>
<td>Monitors that subfile records load if this is a subfile</td>
</tr>
<tr>
<td></td>
<td>Loads subfile records that the system does not use with blanks</td>
</tr>
<tr>
<td>S004</td>
<td>Display and load the screen or report fields.</td>
</tr>
<tr>
<td>S005</td>
<td>Scrubs and edits screen and report fields.</td>
</tr>
<tr>
<td></td>
<td>■ Moves screen data to database fields</td>
</tr>
<tr>
<td></td>
<td>■ Activates error indicators if a field value is in error</td>
</tr>
<tr>
<td></td>
<td>■ Updates records in the database file if this is a subfile</td>
</tr>
<tr>
<td></td>
<td>■ Updates the subfile</td>
</tr>
<tr>
<td>S010</td>
<td>For reports with level breaks it:</td>
</tr>
<tr>
<td></td>
<td>■ Prints the total</td>
</tr>
<tr>
<td></td>
<td>■ Clears the level break totals</td>
</tr>
<tr>
<td></td>
<td>■ Prints the grand total when it reaches the end of the file</td>
</tr>
<tr>
<td></td>
<td>■ Prints the detail</td>
</tr>
<tr>
<td></td>
<td>■ Adds to the new level break totals</td>
</tr>
<tr>
<td></td>
<td>Launches subroutine S020 if this is a report with subheadings</td>
</tr>
<tr>
<td></td>
<td>If this is not a report, S010 updates, adds, or deletes records from the database file.</td>
</tr>
<tr>
<td></td>
<td>Deactivates the Clear Screen (F22) function and executes S001 to clear the buffer before reading another record.</td>
</tr>
<tr>
<td>S020</td>
<td>Print Report Subheadings.</td>
</tr>
<tr>
<td>S998</td>
<td>Loads Data Dictionary values, one time only. Retrieves row description for level breaks and subheadings, if applicable.</td>
</tr>
<tr>
<td>S999</td>
<td>Housekeeping, one time only.</td>
</tr>
<tr>
<td></td>
<td>■ Sets auto inquiry</td>
</tr>
<tr>
<td></td>
<td>■ Defines key lists</td>
</tr>
<tr>
<td></td>
<td>■ Retrieves processing options and level breaks, if applicable</td>
</tr>
<tr>
<td></td>
<td>■ Retrieves vocabulary overrides</td>
</tr>
<tr>
<td></td>
<td>■ Loads error messages</td>
</tr>
<tr>
<td></td>
<td>■ Opens file</td>
</tr>
<tr>
<td></td>
<td>■ Retrieves the current date</td>
</tr>
<tr>
<td></td>
<td>■ Defines work fields using *LIKE</td>
</tr>
<tr>
<td></td>
<td>■ Prints cover page and Helps in a report</td>
</tr>
</tbody>
</table>

**E.2 Flows**
E.2.1 Interactive Non-Subfile Program

Figure E–1 Interactive Non-Subfile Program flow

Mainline

$999

S003

If auto inquiry

Write Screens

S010

Update/add records to file if no error

S004

Load screen fields

$998

S003

If auto inquiry

Read Screens

S998

Validate key fields, set file pointer

S00EX

C0001

Action Code

Process Function keys

Return F1 values to screen fields

S00VL

If one-time only - load Data Dictionary editing information

S001

Edit screen data and move to file fields. Turn on error indicators

S005

Load screen data

$998

Set auto inquiry

S999

If auto inquiry

Set auto inquiry

Retrieval processing options
Retrieve vocabulary overrides
Load error messages
File opens
Date retrieval
E.2.2 Subfile Program with Options

Figure E–2 Subfile Program With Options flow

- Set auto inquiry
- Key lists
- Retrieve processing options
- Retrieve vocabulary overrides
- File opens
- Date retrieval
- If auto inquiry
- Write screens
- One-time only - load Data Dictionary editing information

- Load subfile records
- S003
- Validate key fields(s), set file pointer, monitor for no subfile records loaded, load remaining subfile records with blanks
- S005
- S998
- If auto inquiry
- S010
- S001
- Process selection exits
- Edit the action code
- Clear fields
- If Clear Screen function key is pressed
- S001
- Clear fields
- S004
- S003
- S005
- S001

- C0001
- Process function keys
- Return F1 values to screen fields
- S00EX
- S00VL
- S998
E.2.3 Report Program without Subheadings

Figure E–3 Report Program Without Subheadings flow
E.2.4 Report Program with Subheadings

Figure E–4 Report Program With Subheadings flow
Following is the code to create the basic shell for program type B0010.

Figure F–1  Create/Modify (Basic Shell) screen

![Create/Modify (Basic Shell) screen](image-url)
Figure F–2  Program Code for Program Type B0010 (1 of 11)
Figure F–3  Program Code for Program Type B0010 (2 of 11)
Figure F–4  Program Code for Program Type B0010 (3 of 11)
Figure F–5  Program Code for Program Type B0010 (4 of 11)
Figure F–6  Program Code for Program Type B0010 (5 of 11)
Figure F–7  Program Code for Program Type B0010 (6 of 11)
Figure F–10  Program Code for Program Type B0010 (9 of 11)
Figure F–11  Program Code for Program Type B0010 (10 of 11)

Sample Code

Figure F–12  Program Code for Program Type B0010 (11 of 11)

Sample Code
This appendix contains the topic:

- **Section G.1, "Example: Voucher Processing Functional Server."**

Several JD Edwards World programs access functional servers. The purpose of functional servers is to provide a central location for standard business rules about entering documents, such as vouchers, invoices, and journal entries. These business rules establish the following:

- Data dictionary default values
- Field edits and valid values
- Error processing
- Relationships between fields or applications

The advantages of using a functional server include:

- Reduces maintenance of entry programs because edit rules reside in one central location.
- Allows you to standardize documents across all applications because you create them using the same business rules.
- Separates the user interface (screen appearance and interaction) from the functions of a program.

The steps for setting up business rules for an entry program are:

1. Create a DREAM Writer version for a specific functional server program (for example, XT0411Z1 for voucher entry).
2. Set the processing options within the version according to your company requirements.
3. Specify the version you want the entry program to use in the processing options for that entry program.

You can have all your entry programs use the same DREAM Writer version (and thus, use the same rules) or you can set up different DREAM Writer versions. JD Edwards World provides DREAM Writer version ZJDE0001 as the default functional server version for your entry programs.

---

**Caution:** Only the person responsible for system-wide setup should make changes to the functional server version. For more information about how to set up DREAM Writer versions, see Understand DREAM Writer in the *JD Edwards World Technical Foundation Guide*. 
G.1 Example: Voucher Processing Functional Server

The following programs use the voucher processing functional server. JD Edwards World provides two demo versions of the functional server, ZJDE0001 and ZJDE0002.

- Speed Voucher Entry (P040015)
- Standard Voucher Entry (P04105)
- Void Payment Entry (P4704103)
- Credit Tied to Debit Bill (P041010)
- Multi-Voucher (P041017)
- Calculate Withholding (P04580)
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
</tr>
<tr>
<td>A0010 - Interactive Subfile Inquiry program, C-2</td>
</tr>
<tr>
<td>A0020 - Interactive Single Record Inquiry program, C-3</td>
</tr>
<tr>
<td>Abbreviations for the program types index, 30-1</td>
</tr>
<tr>
<td>About abbreviations for the program types index, 30-1</td>
</tr>
<tr>
<td>action diagramming, 27-1</td>
</tr>
<tr>
<td>additional tools, 24-1</td>
</tr>
<tr>
<td>changing generated source, 17-1</td>
</tr>
<tr>
<td>clone status all/only active toggle, 30-4</td>
</tr>
<tr>
<td>creating or modifying program types, 30-1</td>
</tr>
<tr>
<td>edit screen, 9-4</td>
</tr>
<tr>
<td>foundation information, 2-1</td>
</tr>
<tr>
<td>glossary K, 30-4</td>
</tr>
<tr>
<td>logic modules, 31-1</td>
</tr>
<tr>
<td>master source code, 1-6</td>
</tr>
<tr>
<td>option and function exits, 10-1</td>
</tr>
<tr>
<td>options for program types, 30-2</td>
</tr>
<tr>
<td>program design language, 13-1, 16-1</td>
</tr>
<tr>
<td>program generator, 1-5</td>
</tr>
<tr>
<td>program specifications, 1-6</td>
</tr>
<tr>
<td>program types, 1-6</td>
</tr>
<tr>
<td>program types cross reference, 30-3</td>
</tr>
<tr>
<td>program types index, 30-2</td>
</tr>
<tr>
<td>quick start application tool, 26-1</td>
</tr>
<tr>
<td>source modifications, 16-1</td>
</tr>
<tr>
<td>special characters, 9-1</td>
</tr>
<tr>
<td>the detailed programming facility screen, 11-1</td>
</tr>
<tr>
<td>user defined PDL, 34-1</td>
</tr>
<tr>
<td>using the source code inventory and database, 28-1</td>
</tr>
<tr>
<td>Accessing</td>
</tr>
<tr>
<td>CASE profiles, 4-5</td>
</tr>
<tr>
<td>data item formula revisions, 15-4</td>
</tr>
<tr>
<td>program generator, 6-1</td>
</tr>
<tr>
<td>program generator options, 6-2</td>
</tr>
<tr>
<td>program generator specifications, 5-1</td>
</tr>
<tr>
<td>quick start CL generator, 25-1</td>
</tr>
<tr>
<td>the logic module index, 31-3</td>
</tr>
<tr>
<td>Action diagram</td>
</tr>
<tr>
<td>building, 27-1</td>
</tr>
<tr>
<td>overview, 27-1</td>
</tr>
<tr>
<td>viewing, 27-2</td>
</tr>
<tr>
<td>Adding</td>
</tr>
<tr>
<td>new Q &amp; A dialogue, 33-3</td>
</tr>
<tr>
<td>PDL to a field, 15-4</td>
</tr>
<tr>
<td>Additional tools, 24-1</td>
</tr>
<tr>
<td>All help instructions, 31-2</td>
</tr>
<tr>
<td>Answer Entry screen (P98552), 33-3, 33-4</td>
</tr>
<tr>
<td>Application development cycle, 1-1</td>
</tr>
<tr>
<td>Arrays for EMK, @MK and @ER, B-1</td>
</tr>
<tr>
<td>Assignments in PDL, 14-5</td>
</tr>
<tr>
<td>Authorities for objects, 4-8</td>
</tr>
</tbody>
</table>

| **B** |
| B0010 - Interactive Single Record Maintenance program, C-4 |
| Blocks of statements, 14-3 |
| Browse for screens or reports using quick start, 26-5 |
| Browse screen, 11-13, 12-3 |
| Building an action diagram, 27-1 |

| **C** |
| C0010 - Batch Report with Totals program, C-4 |
| C0020 - Batch Report with Totals and Subheadings program, C-5 |
| C0025 - Batch Report with Totals and Subheadings program, C-6 |
| CAP |
| overview, 1-1 |
| status |
| changing, 18-2 |
| CASE |
| benefits, 1-3 |
| menus, 1-7 |
| profiles |
| accessing, 4-5 |
| understanding, 4-4 |
| program types, C-2 |
| specifications inquiry overview, 31-11 |
| CASE Profiles screen (P98009), 4-5 |
| CASE Specifications Inquiry screen (P93130), 31-11 |
| Changing |
| a dialogue, 33-8 |
| CAP status, 18-2 |
| program specifications, 26-7 |
| Checklist |
| data file design aid, A-1 |
program generator, A-2
report design aid, A-2
screen design aid, A-1

CL
overview, 19-1
program compile using quick start, 25-3

Comments in PDL, 14-4

Compiling
CL programs using quick start, 25-3
using quick start, 26-6, 26-8

Computer Assisted Design, 1-1
Computer Assisted Programming, 1-1

Conditional directives, 32-11

Conditions in PDL, 14-8

Control language, 19-1

Copy File screen (CPYF), 4-2

Copying
a dialogue, 33-8
model control language, 19-1
program specifications, 31-8

Create/Modify Logic Modules screen (P93001SEU), 31-6

Create/Modify Program Types screen (P93001), 30-4, 34-1

Creating
*ENTRY PLIST entries, 11-12
a partial KLIST for a file, 11-17
formula library entry, 31-7
JDESRC file for use with Program Generator, 4-2
logic modules, 31-6
report programs, 23-1
subheadings, 23-6
total formats, 23-3
user defined PDL, 34-1

Customizing model control language, 19-2

D

D0010 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number program, C-7
D0020 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number program, C-9
D0030 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number with Read Next Modified Record program, C-10
D0040 - Interactive Subfile Maintenance with Action Code, with Options, by Key program, C-11
D0050 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Relative Record Number program, C-12
D0060 - Interactive Subfile Maintenance with Action Code, without Options, by Key program, C-14
D0070 - Interactive Subfile Maintenance with Action Code, with Options, by Relative Record Number program, C-15
D0080 - Interactive Subfile Maintenance without Action Code, with Options, by Relative Record Number program, C-16
D0090 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number, Balance program, C-17
D0100 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Key program, C-19

Data dictionary
editing disabled, 11-16
glossary update, 26-9

Data fields to select using quick start, 26-4
Data file design aid checklist, A-1
Data Item Formula Revisions screen (P93109), 14-11, 15-2, 15-3, 34-5

Data structure
I00DSROG source, D-1
I00SC source, D-2

Database operations in PDL, 14-5
Database update function for subfiles, 11-11
Date and time standards, B-4

Define Generator Specification screen (P93100M), 7-1, 8-3, 9-3, 10-1, 11-1, 12-4, 18-2, 18-3, 34-3
Define Program Generator Specifications screen (P93100M), 6-2

Defining
applications using quick start, 26-2
general instructions, 9-1
option and function exits, 10-1
processing options, 12-1
program purpose and type, 7-1

Deleting a dialogue, 33-12
Detailed information
CAD, 1-5
CAP, 1-5

Detailed Programming Facility, 11-1, 21-2, 22-2
Detailed Programming Facility screen (P93105), 11-2, 11-17, 15-4, 34-4
Development libraries, 4-1

Dialogue
changing, 33-8
copying, 33-8
deleting, 33-12
existing, 33-6
rename, 33-9
reviewing the flow, 33-7
running, 33-10

Dialogue Copy screen (P98536), 33-8, 33-9
Dialogue Descriptions screen (P98541), 33-3
Dialogue Flow Revisions screen (P98531), 33-7
Dialogue Lists screen (P98530), 33-7, 33-8, 33-9, 33-10
Dialogue Selection screen (P98533), 7-4, 33-11
Dialogue Test screen (P98535), 33-10, 33-12
Dialogue Test screen (P98537), 33-13

Directives
conditional, 32-11
exception, 32-10
functional, 32-1
substitution, 32-7
understanding, 32-1
Disable data dictionary editing, 11-16
Display Action Diagram screen (P92705), 27-3, 27-4
DREAM Writer considerations for report formats, 23-7

E
E0010 - Interactive Window program, C-20
Edit screen, 9-2, 9-4
Editing, parsing, and source generation of PDL, 15-1
Enabling database update function for subfiles, 11-11
ENTRY PLIST entries, 11-12
Error handling using arrays, B-1
Exception directives, 32-10

F
Features, 1-3
Field Definition screen, 23-4
Field protection, 11-14
File specifications, 8-1
File Specifications screen (P93102), 8-4, 8-7
Formula library entry for creating or modifying, 31-7
Formula Library Entry screen (P93109), 14-10, 31-7
Full data field parameters
   accessing, 11-4
   understanding, 11-4
Full Data Field Parameters screen (P93125), 11-4,
   11-9, 11-10, 11-12, 11-14, 11-15, 11-16
Function exits
   previous profile, 4-8
Function exits set up, 10-1
Functional directives, 32-1
Functional servers
   example, G-2
   overview, G-1

G
General instructions for help text, 9-1
Generated source code
   changing, 17-2
   regenerating, 18-1
   resolving errors, 18-3
Generator Updates screen (P9366), 31-11
Global program regeneration, 31-2
Glossary K, 30-4
Glossary revisions screen (P92001), 26-10
Glossary updates, 26-9
Guidelines for program types, C-2

H
Help file updates, 9-4
Help instructions
   rebuild for a single program, 31-2
   rebuild for all programs, 31-2
Help Instructions Master file (F98HELP), 9-2

I
I00DSPROG data structure source, D-1
I00SC data structure source, D-2
Interactive non-subfile program flow, E-3

J
Job queues, 4-3

K
Keywords
   in blocks of statements, 14-3
   in comments, 14-4
KLIST standards, B-3

L
Line structure, 29-1
Loading VCO description fields, 11-9
Locating
   a dialogue flow, 33-7
   screens or reports using quick start, 26-5
Logic Module Cross Reference screen (P93952), 31-4
Logic modules, 31-1
   accessing the index, 31-3
   creating or modifying, 31-6
   detail, 31-2
   maintaining, 31-5
   viewing cross reference, 31-3
   viewing op codes, 31-4
Logic translation feature, 27-5
Loops in PDL, 14-7

M
Maintaining logic modules
   remove logic module, 31-5
   resequence logic module, 31-5
Master dialogue questions, 33-1
Miscellaneous keywords and syntax in PDL, 14-10
Model Control Language Programs
   customizing, 19-2
   overview, 19-1
   provided by JD Edwards World, 19-2
Modifying
   formula library entry, 31-7
   logic modules, 31-6
   program specifications, 26-7
Moving program specifications, 31-8
Multi-member source file, 4-1

N
Naming convention standards, B-3
New Q & A dialogue, 33-3

O
Object authorities, 4-8

Index-3
job control, 4-8
job queues, 4-8
source file, 4-8
source library, 4-8
Op Codes screen (P93108), 31-4
Operators in assignments, 14-5
Option & Function exits screen (P93104), 10-1
Options
defining, 10-1
overview, 10-1

P

Parameter Copy/Move screen (P93890), 31-8
Partial KLIST created for a file, 11-17
PDL
editing, parsing, and source generation
parsing, 15-1
source code generation, 15-2
statements, 13-1
constants, 14-3
database files, 14-2
keywords, 14-2
operations, 14-2
operators, 14-2
punctuation, 14-3
variables, 14-2
user defined
creating, 34-1
Prerequisites
common user defined codes, 3-2
Program Generator files, 3-1
source code for copy modules, 3-5
source code for JD Edwards World files, 3-5
Primary module, 30-4
Print Program Specification screen (P98300), 31-9
Printing program generator specifications, 31-9
Process for quick start, 26-2
Processing options defined, 12-1
Processing Options Setup screen (P98304), 12-5
Program calls in PDL, 14-7
Program code sample, F-1
Program Design Language (PDL), 13-1, 16-1
Program Generator
accessing, 6-1
checklist, A-2
files, 3-1
program design language, 3-2
source modifications/helps, 3-1
specifications, 3-1
merging updates, 31-11
printing specifications, 31-9
reviewing options, 6-2
specifications
accessing, 5-1
Program purpose and type definition, 7-1
Program Purpose and Type screen (P93100), 7-2, 18-2
Program specifications
modifications using quick start, 26-7

Program specifications to copy or move, 31-8

Program Types
A0010 - Interactive Subfile Inquiry, C-2
A0020 - Interactive Single Record Inquiry, C-3
B0010 - Interactive Single Record Maintenance, C-4
C0010 - Batch Report with Totals, C-4
C0020 - Batch Report with Totals and Subheadings, C-5
C0025 - Batch Report with Totals and Subheadings, C-6
D0010 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number, C-7
D0020 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number, C-9
D0030 - Interactive Subfile Maintenance without Action Code, without Options, by Relative Record Number with Read Next Modified Record, C-10
D0040 - Interactive Subfile Maintenance with Action Code, with Options, by Key, C-11
D0050 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Relative Record Number, C-12
D0060 - Interactive Subfile Maintenance with Action Code, without Options, by Key, C-14
D0070 - Interactive Subfile Maintenance with Action Code, with Options, by Relative Record Number, C-15
D0080 - Interactive Subfile Maintenance without Action Code, with Options, by Relative Record Number, C-16
D0090 - Interactive Subfile Maintenance with Action Code, without Options, by Relative Record Number, Balance, C-17
D0100 - Interactive Subfile Maintenance with Two Master Files, with Action Code, with Options, by Key, C-19
E0010 - Interactive Window, C-20
guidelines, C-2
overview, C-2
X0010 - Batch Update with Report, C-21
X0020 - Batch Update, C-22
X0030 - Batch Update with Subroutine S001, C-23
X0040 - Batch Update with Report, C-23
Y0010 - Conversion, Two Files with Error Report, C-24
Y0020 - Conversion, One File Update with Error Report, C-25
Y0030 - Conversion, One File Write with Error Report, C-26

Program types
B0010 example, F-1
conversion, 1-7
creating or modifying, 30-1
cross reference, 30-3
index, 30-2
interactive, 1-6
Program Types Index screen (P93900), 30-2
Program Types X-Reference screen (P93953), 30-3
Programming standards, B-1
Programs and IDs
CPYF (copy file), 4-2
P00051 (user defined code revisions), 3-2
P2710 (translation table), 27-5
P92001 (glossary revisions), 26-10
P92705 (display action diagrams), 27-3, 27-4
P93001 (create/modify program types), 30-4, 34-1
P9301SEU (create/modify logic modules), 31-6
P93100 (program purpose and type), 7-2, 18-2
P93100M (define generator specifications), 7-1, 8-3, 9-1, 11-1, 12-4, 18-2, 18-3, 34-3
P93100M (define generator specifications), 6-2
P93102 (file specifications), 8-4, 8-7
P93104 (option & function exits), 10-1
P93105 (detailed programming facility), 11-2, 11-17, 15-4, 34-4
P93108 (logic module op codes), 31-4
P93109 (data item formula revisions), 14-11, 15-2, 15-3, 15-4, 34-5
P93109 (formula library entry), 14-10, 31-7
P93125 (full data field parameters), 11-4, 11-9, 11-10, 11-12, 11-14, 11-15, 11-16
P93130 (CASE specifications inquiry), 31-11
P93513 (quick start application tool), 26-2
P93513J (quick start CL generator), 25-1
P93515V (quick start CL generator), 26-4, 26-5, 26-6, 26-7, 26-8, 26-9, 26-10
P93515V (quick start CL generator), 25-3
P9366 (generator updates), 31-11
P93890 (parameter copy/move), 31-8
P93900 (program types index), 30-2
P93952 (logic module cross reference), 31-4
P93953 (program types x-reference), 30-3
P98009 (CASE profiles), 4-5
P9801 (software versions repository), 4-5, 8-8
P98300 (build action diagrams), 27-1
P98300 (print program specification), 31-9
P98304 (processing options setup), 12-1
P98529 (simple question & answer), 33-1
P98530 (dialogue lists), 33-7, 33-8, 33-9, 33-10
P98531 (dialogue flow revisions), 33-7
P98533 (dialogue selection), 7-4, 33-11
P98534 (quiz answer review), 33-11
P98535 (dialogue test), 33-10, 33-12
P98536 (dialogue copy), 33-8, 33-9
P98537 (dialogue test), 33-13
P98541 (dialogue descriptions), 33-3
P98551 (question entry), 33-3
P98552 (answer entry), 33-3, 33-4
RM/M (remove member), 4-3
Project management, 4-3
Protecting fields from being cleared, 11-14
Q
Q & A dialogue, 33-3
Question and answer system overview, 33-1
Question Entry screen (P98551), 33-3
Questions in a master dialogue, 33-1
Quick reference of program types, C-2
Quick Start
tasking tool overview, 26-1
CL generator for creating programs, 21-1, 22-2, 25-1
generating subfile inquiry programs, 21-2
generator to create subfile maintenance programs, 22-2
steps for process, 26-2
Quick Start Application Tool screen (P93513), 26-2
Quick Start Application Tool screen (P93515V), 26-4, 26-5, 26-6, 26-7, 26-8, 26-9, 26-10
Quick Start C L Generator screen (P93515V), 25-3
Quick Start CL Generator screen (P93513), 25-1
Quiz Answer Review screen (P98534), 33-11
Quiz to determine program type, 33-12
R
RDA, 23-1
Rebuild help instructions, 31-2
Record Formats List screen, 23-6
Regenerating source code, 31-2
Remove logic module, 31-5
Remove Member (RM/M), 4-3
Renaming a dialogue, 33-9
Report Design Aid
creating reports, 23-1
Report design aid checklist, A-2
Report format considerations for DREAM
Writer, 23-7
Report program with subheadings flow, E-6
Report program without subheadings flow, E-5
Report programs subheadings and totals, 23-1
Reports or screens
browse or update using quick start, 26-5
compiling using quick start, 26-6
Resequence logic module, 31-5
Reviewing
dialogue flow, 33-7
questions, 33-1
source modifications, 31-10
RPG subroutines, E-1
Running a dialogue, 33-10
Running a quiz, 33-12
S
Sample program code, F-1
Screen design aid checklist, A-1
Screens
Answer Entry, 33-3, 33-4
Browse, 11-13, 12-3
Build Action Diagrams, 27-1
CASE Profiles, 4-5
CASE Specifications Inquiry, 31-11
Copy File (CPYF), 4-2
Create/Modify Logic Modules, 31-6
Create/Modify Program Types, 30-4, 34-1
Data Item Formula Revisions, 14-11, 15-2, 15-3, 15-4, 34-5
Define Generator Specification, 7-1, 8-3, 9-3, 10-1, 11-1, 12-4, 18-2, 18-3, 34-3
Define Generator Specifications, 6-2
Detailed Programming Facility, 11-2, 11-17, 15-4, 34-4
Dialogue Copy, 33-8, 33-9
Dialogue Descriptions, 33-3
Dialogue Flow Revisions, 33-7
Dialogue Lists, 33-7, 33-8, 33-9, 33-10
Dialogue Selection, 7-4, 33-11
Dialogue Test, 33-10, 33-12, 33-13
Display Action Diagram, 27-3, 27-4
Edit, 9-2, 9-4
Field Definition, 23-4
File Specifications, 8-4, 8-7
Formula Library Entry, 14-10, 31-7
Full Data Field Parameters, 11-4, 11-9, 11-10, 11-12, 11-14, 11-15, 11-16
Generator Updates, 31-11
Glossary, 26-10
Logic Module Cross Reference, 31-4
Op Codes, 31-4
Option & Function exits, 10-1
Parameter Copy/Move, 31-8
Print Program Specification, 31-9
Processing Options Setup, 12-5
Program Purpose and Type, 7-2, 18-2
Program Types Cross Reference, 30-3
Program Types Index, 30-2
Question Entry, 33-3
Quick Start Application Tool, 26-2, 26-4, 26-5, 26-6, 26-7, 26-8, 26-9, 26-10
Quick Start CL Generator, 25-1, 25-3
Quiz Answer Review, 33-11
Record Formats List, 23-6
Remove Member, 4-3
Simple Question & Answer, 33-1
Software Versions Repository, 4-5, 8-8
Source Entry Utility, 19-1
Translation Table, 27-5
User Defined Code Revisions, 3-2

Screens or reports
   browse or update using quick start, 26-5
   compiling using quick start, 26-6
Selecting data fields using quick start, 26-4
Serial numbers, 29-1, 29-2
Simple Question & Answer screen (P98529), 33-1
Software Action Request (SAR), 4-3
Software Versions Repository screen (P9801), 4-5, 8-8, 19-1
Solving generation problems, 18-3
Source code
   for copy modules, 3-5
   for JD Edwards World files, 3-5
   inventory and database, 28-1
   regenerating, 18-1
   when to regenerate, 18-1
Source Entry Utility screen, 19-1
Source Listings, D-1
Source modifications
   about, 16-1
   code review, 31-10
Source sequence line numbers, 29-1
Source sequence line structure, 29-1
Source serial numbers, 29-1
Special characters
   about, 9-1
   within the help file, 9-2
Standards for programming, B-1
Standards using functional servers, G-1
Statements in PDL, 13-1
Structure of serial numbers, 29-2
Subfile Inquiry Program
   components, 21-1
   overview, 21-1
Subfile Maintenance Program
   components, 22-1
   overview, 22-1
   special considerations, 22-2
Subfile program with options flow, E-4
Subfiles database update function, 11-11
Subheadings
   formats, 23-6
   overview, 23-1
Submit program to compile using quick start, 26-8
Subroutines for RPG, E-1
Substitution directives, 32-7
Syntax
   in assignments, 14-5
   in blocks of statements, 14-3
   in comments, 14-4
System integration, 1-1
   application development cycle, 1-1
   fundamentals, 1-2
   history of program generator, 1-2
   specifications, 1-2

T
Terms and concepts, 1-4
Totaling
   overview, 23-1
Translation Table screen (P92710), 27-5

U
Understanding
   assignments, 14-5
   operator and syntax, 14-5
   rules, 14-5
   blocks of statements, 14-3
   keywords and syntax, 14-3
   rules, 14-3

Index-6
CASE profiles, 4-4
comments
  keywords and syntax, 14-4
  rules, 14-4
conditions
  keywords and syntax, 14-8
  rules, 14-9
  symbols, 14-9
database operations
  keywords and syntax, 14-5
  rules, 14-6
directives, 32-1
full data field parameters, 11-4
loops
  keywords and syntax, 14-7
  rules, 14-8
miscellaneous keywords and syntax
  keywords and syntax, 14-10
  rules for include, 14-10
  rules for return, 14-12
program calls
  keywords and syntax, 14-7
  rules, 14-7
source sequence line numbers, 29-1
  source sequence line structure, 29-1
  source serial numbers, 29-1
  structure of the serial number, 29-2
Updating
  data dictionary, 26-9
  glossary, 26-9
  help file, 9-4
  screens or reports using quick start, 26-5
Usage indicator standards, B-2
User Defined Code Revisions screen (P00051), 3-2
User Defined Codes, 3-2
User-provided prerequisites, 4-1
Using
  CASE specifications inquiry, 31-11
  program generator updates, 31-11

V

VCO description fields for screens, 11-9
Viewing
  an action diagram, 27-2
  logic module cross reference, 31-3
  logic module op codes, 31-4

W

Work field standards, B-4
Working with
  file specifications, 8-1
  the question and answer system, 33-1
  user provided prerequisites
    development libraries, 4-1
    job queues, 4-3
    multi-member source file, 4-1
    overview, 4-1
    project management, 4-3

X

X0010 - Batch Update with Report program, C-21
X0020 - Batch Update program, C-22
X0030 - Batch Update with Subroutine S001 program, C-23
X0040 - Batch Update with Report program, C-23

Y

Y0010 - Conversion, Two Files with Error Report program, C-24
Y0020 - Conversion, One File Update with Error Report program, C-25
Y0030 - Conversion, One File Write with Error Report program, C-26