

# CA 2E

## Tutorial

r8.5



This documentation and any related computer software help programs (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of the documentation for their own internal use, and may make one copy of the related software as reasonably required for back-up and disaster recovery purposes, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the provisions of the license for the product are permitted to have access to such copies.

The right to print copies of the documentation and to make a copy of the related software is limited to the period during which the applicable license for the Product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

EXCEPT AS OTHERWISE STATED IN THE APPLICABLE LICENSE AGREEMENT, TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

The use of any product referenced in the Documentation is governed by the end user's applicable license agreement.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Copyright © 2009 CA. All rights reserved.

## Contact CA

### Contact Technical Support

For your convenience, CA provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA products. At <http://ca.com/support>, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

### Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to [techpubs@ca.com](mailto:techpubs@ca.com).

If you would like to provide feedback about CA product documentation, complete our short [customer survey](#), which is also available on the CA support website, found at <http://ca.com/support>.



# Contents

---

## Chapter 1: About this Tutorial

Introduction .....	1-1
Chapter 2 Defining Requirements .....	1-1
Chapter 3 Data Modeling .....	1-1
Chapter 4 Designing Functions .....	1-1
Chapter 5 Generating, Compiling, and Executing .....	1-2
Chapter 6 Maintaining Your Application .....	1-2
Chapter 7 Advanced Functions .....	1-2
Chapter 8 Report Functions .....	1-2
Related Information .....	1-2
Getting Started .....	1-2
Defining a Data Model .....	1-2
Building Access Paths .....	1-3
Building Applications .....	1-3
Generating and Implementing Applications .....	1-3
CA 2E and the Application Development Life Cycle .....	1-4
Prerequisites for Using CA 2E .....	1-5

## Chapter 2: Defining Requirements

Introduction to Defining Requirements .....	2-1
Requirements Definition .....	2-1

## Chapter 3: Data Modeling

Data Model .....	3-1
Objectives .....	3-1
Identify Application Entities .....	3-2
Identify Business Relationships Between Entities .....	3-2
Identify Attributes and Unique Identifiers for the Entities .....	3-3
Identify the Attributes for Each Entity .....	3-3
Select Each Entity's Unique Identifier .....	3-4
Normalize the Entities .....	3-7
Generalize the Entities .....	3-8
CA 2E Data Model Diagram .....	3-9
CA 2E Relations .....	3-10

---

Objectives .....	3-10
Overview of CA 2E Relations .....	3-10
The Edit Database Relations Panel .....	3-11
Entering Relation Statements .....	3-12
Validation of Relation Statements .....	3-13
Defining Objects .....	3-14
Specifying Object Attributes .....	3-15
File Entries .....	3-16
File Entries for COURSE .....	3-16
File Entries for RACE .....	3-18
Adding More Relations .....	3-18
Declaring More Files .....	3-19
Defining Objects .....	3-20
File Attributes .....	3-20
Deleting Relations .....	3-20
Documenting Relations .....	3-21
Field Details and Conditions .....	3-21
Objectives .....	3-21
Overview of Field Details .....	3-22
Overview of Field Conditions .....	3-22
Field Details .....	3-22
Field Detail Display for Horse Gender .....	3-22
Field Conditions .....	3-23
Adding a Field Condition .....	3-23
Field Condition Details .....	3-25
Adding Field Condition Detail .....	3-25
Adding Another Condition .....	3-25
Viewing Field Conditions .....	3-26
Specifying a Check Condition .....	3-27
Exercises .....	3-28
Displaying Selected Relations .....	3-28
Extending Relations .....	3-29
Objectives .....	3-30
Overview of Relation Extension .....	3-30
Involution and the Horse Pedigrees .....	3-30
Adding More Relations .....	3-31
Sequence of Relations .....	3-32
Extending Relations .....	3-33
Extended Relations for HORSE File .....	3-33
Example Showing Use of the Sharing Field .....	3-34
Adding Details of Extended Relations .....	3-34
Field Entries for HORSE .....	3-35
Assigning Unique Names to Field Entries .....	3-36

---

Showing and Hiding Relation Extension Lines .....	3-36
Virtual Fields .....	3-37
Overview of Virtual Fields .....	3-37
Specifying Virtual Fields .....	3-38
Virtual Fields for Details of a Horse's Dam and Sire .....	3-38
Sequence of Relations and Virtual Fields .....	3-38
Adding Virtual Fields .....	3-39
Virtual Field Entries .....	3-40
Selecting Virtual Fields for Dam .....	3-40
Confirming Virtual Fields for Dam .....	3-40
Adding More Virtual Fields .....	3-41
Selecting Virtual Fields for Sire .....	3-41
Confirming Virtual Fields for Sire .....	3-42
File Entries .....	3-42
Renaming Fields .....	3-43
Field Details .....	3-44
Entering a New Field Name .....	3-44
Field Details After Renaming .....	3-45
Renaming Other Fields .....	3-45
Exercise .....	3-46
Displaying File Entries Again .....	3-46
Adding Virtual Fields for the Race and Race Entry Files .....	3-46
Adding More Virtual Entries .....	3-47
Exercise .....	3-48
CA 2E Access Paths .....	3-49
Objectives .....	3-49
Overview of Access Paths .....	3-50
Default Access Paths .....	3-50
View Default Access Paths for COURSE .....	3-50
Access Paths for the HORSE File .....	3-52
Adding Virtual Fields to the Retrieval Access Path .....	3-53
Adding New Access Paths for HORSE .....	3-56
Confirming Addition of Access Paths .....	3-57
Access Path Details .....	3-57
Select/Omit Sets .....	3-58
Static and Dynamic Selection .....	3-58
Selection Conditions for the Mares Access Path .....	3-58
Select/Omit Set for Mares Access Path .....	3-59
Specifying the Conditions for the Select/Omit Set .....	3-60
Selection Access Path for Stallions .....	3-61
Access Path Details .....	3-61
Naming the Select/Omit Set .....	3-62
Specifying Conditions for the Access Path .....	3-62

---

Access Path Relations .....	3-63
Modifying the Default RTV Access Path for HORSE .....	3-63
Displaying Access Path Relations .....	3-64
Access Path Relations for the Retrieval Index .....	3-65
Involution and Access Path Relations .....	3-65
Changing a Referenced Access Path .....	3-65
Selecting an Access Path for Mares .....	3-66
Selecting an Access Path for Stallions .....	3-66

## Chapter 4: Designing Functions

Introduction to Functions .....	4-1
Objectives .....	4-1
Overview of Functions .....	4-2
Default Functions .....	4-3
Default External Functions .....	4-3
Default Internal Functions .....	4-4
Defining Edit Course and Select Course .....	4-4
Displaying Default Functions For HORSE .....	4-5
Adding Functions .....	4-6
Select Mares and Select Stallions Functions .....	4-6
Creating New Functions .....	4-7
Understanding Function Details .....	4-7
Displaying Function Details .....	4-8
Device Designs .....	4-8
Overview of Device Designs .....	4-9
Displaying the Device Design for Edit Horse .....	4-9
Default Device Design Formats .....	4-10
Design Standards and Subfile Selector Options .....	4-11
Subfile Control Default .....	4-11
Subfile Record Default .....	4-12
Displaying the Rest of the Device Design .....	4-12
Editing the Default Device Design .....	4-13
Reducing the Width of the Device Design Layout .....	4-14
Folding the Device Design Layout Again .....	4-15
Panel Format Details .....	4-16
Edit Screen Format Details .....	4-16
Editing the Panel Format .....	4-17
Effect of Screen Format Changes on Device Design .....	4-18
Shortening Field Labels .....	4-19
Shortening Field Label for Dam Horse Code .....	4-20
Shortening Field Label for Sire Horse Code .....	4-20



---

Removing Field Labels .....	4-21
Removing Field Label for Dam Name .....	4-21
Removing Field Label for Sire Name .....	4-22
Centering a Field with Respect to its Label .....	4-23
Moving Fields to Right and Left .....	4-24
Using Function Keys .....	4-25
Using the Edit Screen Entry Details Panel .....	4-25
Adjusting the Label Spacing of the Dam Field .....	4-26
Modified Panel .....	4-27
Optional Exercise .....	4-28
Panel Format Relations .....	4-29
Editing Panel Format Relations .....	4-29
Completed Device Design .....	4-30
Exiting the Device Design .....	4-31
Exit Function Definition .....	4-32
Function Confirmation .....	4-32
Window Device Design .....	4-32
Invoking the Device Design .....	4-33
Default Device Design for Select Horse .....	4-33
Hiding Fields in the Subfile Record Format .....	4-34
Modified Subfile Control Format .....	4-35
Exercise .....	4-35
Window Options Editor .....	4-36
Changing Window Dimensions .....	4-37
Completed Device Design .....	4-37
Exiting the Device Design .....	4-38
Exercises .....	4-38
Action Diagrams .....	4-39
Overview of Action Diagrams .....	4-40
Default Action Diagram .....	4-41
Hidden Constructs and User Points .....	4-42
Edit Horse Action Diagram .....	4-43
Displaying Hidden Constructs in a Action Diagram .....	4-43
Adding Extra Validation to Edit Horse .....	4-43
The Process Response Construct .....	4-44
The Process Screen Construct .....	4-45
The Validate Subfile Record Construct .....	4-45
Adding a Validation Procedure .....	4-46
Subfile Record Relations .....	4-47
Editing the Action Diagram .....	4-47
Editing the Edit Horse Action Diagram .....	4-50
Inserting a Condition .....	4-50
Inserting a Single Action .....	4-50

---

---

Adding Another Condition by Copying .....	4-51
Specifying Details of Conditions .....	4-52
The Edit Action Condition Window .....	4-53
Contexts .....	4-53
Adding Details of the First Condition .....	4-53
Adding the Second Condition .....	4-54
Adding Details of the Second Condition .....	4-55
Adding Actions .....	4-55
Exiting the Action Diagram .....	4-55
Message Functions .....	4-57
Displaying the Message Functions .....	4-58
Adding a New Message Function .....	4-59
Defining Parameters .....	4-59
Specifying Parameters for the First Message Function .....	4-60
Message Function Details .....	4-61
Adding Text to the First Message .....	4-62
Returning to the Edit Horse Action Diagram .....	4-62
Displaying User Points for Edit Horse .....	4-64
Add Action for First Condition .....	4-64
The Edit Action Function Name Panel .....	4-65
Message Functions .....	4-65
Displaying the Message Functions .....	4-66
Selecting a Message Function .....	4-66
Function Details for the First Action .....	4-67
Action Diagram with the First Action Defined .....	4-68
Defining the Second Action .....	4-68
Adding the Second Message Function .....	4-69
Details of the Second Message Function .....	4-70
Defining Parameters for the Second Message Function .....	4-71
Message Function Details .....	4-71
Adding Message Function Text .....	4-72
Selecting the Second Message Function .....	4-72
Returning to the Action Diagram .....	4-73
Completed Action Diagram .....	4-73
Exiting the Action Diagram .....	4-74
Function Options .....	4-75
Objectives .....	4-75
Accessing the Function Options Panel .....	4-75
Default Function Options .....	4-76
Changing Function Options .....	4-77
Linking Functions .....	4-78
Objectives .....	4-78
Steps Required to Link Functions .....	4-79

---

Modifying the Edit Horse Action Diagram .....	4-80
Obtaining the Action Diagram User Points .....	4-81
Adding a New CASE Construct .....	4-81
Inserting a Condition .....	4-82
Entering the Condition .....	4-82
Entering the Condition Details .....	4-82
Selecting the Zoom#1 Condition .....	4-83
Specifying a Function as the Action .....	4-84
Naming the Function .....	4-84
Selecting a Function .....	4-85
Creating the New Function and Access Path .....	4-85
Selecting an Access Path .....	4-86
Creating a New Access Path .....	4-86
Specifying the Access Path Details .....	4-87
Defining the Access Path Key .....	4-88
Selecting the Access Path .....	4-88
Function Parameters .....	4-89
Objectives .....	4-90
Understanding Parameter Usage and Role .....	4-90
Specifying Function Parameters .....	4-90
Defining Function Parameters .....	4-91
Specifying Parameters Using an Access Path .....	4-92
Defining Parameter Details .....	4-92
Completing the Parameter Details .....	4-93
Selecting the New Function .....	4-93
Returning to the Edit Action - Function Details Window .....	4-94
Suppressing the Confirm Prompt and File Update .....	4-95
Inserting *MOVE as an Action .....	4-96
Specifying Parameters for the *MOVE Function .....	4-96
Specifying Defer Confirm .....	4-97
Reload Subfile .....	4-97
Specifying Reload Subfile .....	4-98
The Completed Action Diagram .....	4-98
Saving the Action Diagram .....	4-99
Updating the Edit Horse Function's Action Bar .....	4-99
Updating the Edit Horse Panel's Zoom Action Text .....	4-100
Working with the Selector Choice .....	4-101
Modifying an Action .....	4-101
Changing the Action Text .....	4-102
Showing the New Action Text .....	4-102
Exiting the Action Bar Editor .....	4-103
Exiting the Modified Panel .....	4-103
Saving the Modified Panel .....	4-103

---

---

Edit Database Relations Panel .....	4-104
Exercise .....	4-105

## Chapter 5: Generating, Compiling, and Executing

Implementing Access Paths and Functions .....	5-1
Objectives .....	5-1
Overview of Implementation .....	5-1
Source Generation and Compilation of Access Paths and Functions .....	5-2
Display all Access Paths for Selection .....	5-2
Selecting All Access Paths for Batch Generation and Compilation .....	5-3
Completing the Request .....	5-4
Display all Functions for Selection .....	5-4
Selecting All External Functions for Batch Generation and Compilation .....	5-5
Completing the Request .....	5-6
Submit Batch Generation and Creation.....	5-7
Generating and Creating Objects .....	5-7
List of Objects to be Generated and Created .....	5-8
Confirming the Job List .....	5-9
Successful Submit for Generation and Compilation.....	5-9
Examining the Job List.....	5-10
Converting Condition Values to a Database File .....	5-11
YCVTCNDVAL Command Prompt.....	5-12
Confirming Conversion of Condition Values.....	5-13
Exiting the CA 2E Design Model .....	5-13
Re-synchronizing an CA 2E Design Model .....	5-14
Executing and Testing Compiled Programs .....	5-15
Calling the Program .....	5-15
The Edit Horse Function Panel .....	5-16
Adding Data to the HORSE File .....	5-17
Confirming Data Entries for the Horse File .....	5-18
Switching from New to Open Mode .....	5-19
Exiting the Edit Horse Program .....	5-19
Exercises .....	5-20

## Chapter 6: Maintaining Your Application

Application Maintenance .....	6-1
Animating an Interactive Device Design .....	6-2
Overview of CA 2E Animation .....	6-2
Animate the Edit Horse Device Design .....	6-3
Animate Functions Panel .....	6-4

---

Animating Edit Horse .....	6-4
Entering Sample Data for Edit Horse .....	6-5
Editing and Maintaining Multiple Functions .....	6-6
Open Functions Panel .....	6-7
Animating the Display Racing Results Function .....	6-7
Converting Command Key Navigation .....	6-8
Working with Toolkit Panel Designs .....	6-9
Toolkit Command Key Navigation .....	6-9
Reassigning a Command Key in Toolkit .....	6-10
Setting Up Action Bar Navigation for Edit Horse .....	6-11
Assigning Action Bar Navigation .....	6-12
Testing the Function Link .....	6-15
Animating Edit Horse .....	6-16
Activate the Action Bar .....	6-17
Display Racing Results Panel Design .....	6-18
Exiting Both Functions .....	6-19
Working with Model Object Lists .....	6-20
Objectives .....	6-20
Overview of Model Objects and Model Object Lists .....	6-21
Editing Your Session List .....	6-22
Edit Model Object List Panel .....	6-23
Viewing Model Object Types .....	6-23
List Entry Differs from Model Object .....	6-23
Positioning a Model Object List .....	6-25
Returning to the Top of the List .....	6-26
Viewing a Subset of a Model Object List .....	6-27
Opening Multiple Functions at One Time .....	6-28
Working with Open Functions .....	6-30
Displaying the Unsubsetting Model List .....	6-30
Editing Model Objects .....	6-31
Exercise .....	6-32
Editing Conditions for Entry Status .....	6-32
Exercise .....	6-33
Displaying the Unsubsetting Model List .....	6-33
Editing Relations and Creating Objects .....	6-33
Deleting a Model List Entry .....	6-34
All Objects List .....	6-36
Accessing the All Objects List .....	6-36
How *ALLOBJ and Model Lists Differ .....	6-37
Displaying Alternate Views of Detail Information .....	6-37
Exercise .....	6-37
Restoring the Deleted Session List Entry .....	6-38
Accessing Other Model Lists .....	6-38

---

---

Returning to Your Session List .....	6-38
Working with Model Object Lists .....	6-39
Function Versioning .....	6-40
Objectives .....	6-41
Overview of Versions .....	6-41
Using Versions to Update an Existing Function .....	6-42
Accessing the Session List .....	6-42
Positioning to the Edit Horse .....	6-43
Creating a Version for Edit Horse .....	6-44
Working with Versions .....	6-45
Naming the Version .....	6-46
Implementation Name for the New Version .....	6-47
Editing the New Function .....	6-47
Inserting a Message Function .....	6-48
Specifying a Parameter for the Message Function .....	6-49
Editing the Message Text .....	6-50
Using a Substitution Variable .....	6-51
Selecting the New Message .....	6-51
Checking the Default Parameters .....	6-52
Returning to the Action Diagram .....	6-53
Submitting the Function Version for Generation .....	6-54
Viewing Job List Commands .....	6-54
Displaying Alternate Views .....	6-56
Testing Edit Horse - Version 1 .....	6-56
Comparing Two Versions of a Function .....	6-57
Entering Parameters for YCMPMDLOBJ .....	6-57
Viewing Differences between Versions .....	6-58
Making the Edit Horse - Version 1 Current .....	6-59
Naming the New Current Version .....	6-60
Regenerating the New Current Version .....	6-61
Exercise .....	6-61
Viewing Model Object Information for the Versions .....	6-62
Refreshing List Entries for the Versions .....	6-63
Exercise .....	6-64
Model Object Cross References .....	6-64
Objectives .....	6-65
Overview of Model Object Cross References .....	6-65
Model Object Usages .....	6-65
Accessing Your Session List .....	6-65
Display Model Usages Panel .....	6-66
Usage Reason .....	6-67
Exercise .....	6-68
Using Usage Levels .....	6-68

---

Exiting Model Usages .....	6-68
Model Object References .....	6-69
Accessing the *ALLOBJ List .....	6-69
Positioning *ALLOBJ to an Implementation Name .....	6-69
Displaying References for Edit Horse .....	6-70
Display Model References Panel .....	6-70
Displaying Only External Functions .....	6-71
Creating a Model List of the References .....	6-72
Naming the List of References .....	6-72
Using the List of References .....	6-73
Exiting Display Model References .....	6-73
Redisplay Your Session List .....	6-73
Impact Analysis .....	6-74
Objectives .....	6-74
Overview of Impact Analysis .....	6-74
Change Type .....	6-75
Simulating a Change to a Model Object .....	6-75
Positioning the List to Course Code .....	6-76
Positioning the Usages to Course Code .....	6-76
Simulating a *PUBLIC Change .....	6-77
Interpreting the Results .....	6-78
Converting the Simulation Usages to a List .....	6-78
Optional Exercise .....	6-79
Component Change Processing .....	6-79
Accessing Edit Database Relations .....	6-80

## Chapter 7: Advanced Functions

Span Access Path and Edit Transaction Function .....	7-1
Objectives .....	7-1
Overview .....	7-1
Overview of Processing Steps .....	7-2
Entering Your Design Model .....	7-2
Creating a Span Access Path .....	7-3
Defining the Span Access Path .....	7-3
Entering the Access Path Formats .....	7-4
Selecting the First Access Path Format .....	7-5
Confirming Selection of the First Format .....	7-5
Specifying the Second Format .....	7-6
Adding Virtual Fields to the Access Path Formats .....	7-6
Details for the First Access Path Format .....	7-9
Specifying the Key for the First Access Path Format .....	7-10

---

Details for the Second Access Path Format .....	7-10
Requesting Batch Generation of the SPN Access Path .....	7-11
The Edit Transaction Function .....	7-12
Objectives .....	7-12
Defining the Edit Transaction Function .....	7-12
The Default Device Design .....	7-13
The Modified Device Design .....	7-14
Defining Optional Entry Fields .....	7-15
Introduction to Function Fields .....	7-17
Objectives .....	7-17
Overview of Function Fields .....	7-17
Adding Function Fields .....	7-18
Naming the Function Field .....	7-18
Displaying Existing Fields .....	7-19
Defining Two New Function Fields .....	7-19
Overriding the Field Defaults .....	7-20
Displaying the Function Fields .....	7-21
Defining Function Field Parameters .....	7-21
Selecting the Function Field .....	7-23
Accepting the Parameters .....	7-23
The Modified Device Design .....	7-24
Defining a Function Field Action Diagram .....	7-24
Adding a CASE Construct .....	7-26
Adding an Action .....	7-27
Adding a Second Condition .....	7-28
Adding a Second Action .....	7-28
Defining the Conditions and Actions .....	7-29
Specifying the First Condition .....	7-29
Defining Field Conditions for the Status Field .....	7-30
Specifying the First Action .....	7-31
Specifying the Second Condition .....	7-32
Defining the Second Action .....	7-32
Complete Action Diagram .....	7-33
Returning to the Device Design .....	7-35
Adding the Second Function Field .....	7-36
Specifying the Second Function Field .....	7-36
Defining Parameters for the No. of finishers Function Field .....	7-37
The Added Function Field .....	7-38
Readjusting the Device Design .....	7-38
Exiting the Device Design .....	7-39
Submitting the Function for Generation .....	7-39
Exercises .....	7-40



---

## Chapter 8: Report Functions

Introduction to the Print File Function .....	8-1
Objectives .....	8-1
Defining the Print File Function .....	8-2
Specifying a New Access Path .....	8-2
Adding Virtual Entries to the Access Path .....	8-3
Edit Access Path Details .....	8-4
Specifying Access Path Details .....	8-5
The New Key Order .....	8-5
Generating and Compiling the New Access Path .....	8-5
Selecting the Access Path .....	8-6
The Report Design .....	8-6
The Default Report Layout .....	8-7
Displaying the Report Formats .....	8-8
Dropping Formats from a Report Design .....	8-9
Adding Function Fields .....	8-9
Displaying Existing Function Fields .....	8-10
Defining New Function Fields .....	8-11
Specifying Function Parameters .....	8-12
The Default Parameters .....	8-13
Selecting the Function Fields .....	8-14
Confirming the Function Details .....	8-14
Adding the Total Number of Horses Function Field .....	8-15
Confirming the Parameters .....	8-16
Exercise .....	8-16
The Report Design with Function Fields .....	8-17
Completing the Report .....	8-17
The Modified Report Device Design .....	8-18
The Completed Report Layout .....	8-18
Saving the Report Device Design .....	8-20
Generating and Compiling the Function .....	8-20
Running Your Program .....	8-20
Introduction to the Print Object Function .....	8-21
Objectives .....	8-21
Defining the Print Object Function .....	8-22
Adding a Print Race Entries Function .....	8-22
Specifying a Query Access Path .....	8-23
Displaying the Access Path Format Entries .....	8-24
Specifying an Alternative Key .....	8-25
Compiling the QRY Access Path .....	8-26
Selecting the Access Path for the Function .....	8-26
Displaying the Report Device Design .....	8-27

---

The Default Report Device Design .....	8-27
Exercise - Updating the Report Layout .....	8-28
Exiting the Device Design .....	8-28
Combining the Report Functions .....	8-29
Embedding the Print Race Entries Function .....	8-30
Selecting the PRTOBJ Function .....	8-31
The Modified Device Structure .....	8-31
Saving the Device Structure .....	8-32
Displaying the Report Device Design .....	8-32
The New Report Device Design .....	8-33
View the Rest of the Report Device Design .....	8-33
Modifying the Combined Report Device Design .....	8-34
Specifying Parameters .....	8-35
Editing the Action Diagram .....	8-36
The Print Details Construct .....	8-37
Editing the Print Race Entries Function .....	8-37
The Action Diagram Editor Subfile Selector Values .....	8-38
Editing the Function .....	8-38
Displaying the Function Parameters .....	8-39
Editing the Function Parameters .....	8-39
Specifying a Restrictor Parameter .....	8-40
Confirming Parameter Details .....	8-41
Exit Action Diagram .....	8-41
Generating and Compiling the Functions .....	8-41
Testing the Program .....	8-42

## Glossary

## Index

# Chapter 1: About this Tutorial

---

This tutorial provides a navigational guide to the new CA 2E® user. It will guide you through the steps required to develop an application using CA 2E. After completing this tutorial, you will have designed an CA 2E working application. The tutorial includes all of the basic facilities of CA 2E as well as some advanced features.

## Introduction

This chapter describes the organization of the tutorial, sources for additional information about CA 2E, conventions used in the tutorial, the application development life cycle, guidelines for getting started with the tutorial, and a glossary of terms introduced in the tutorial.

**Note:** Be sure to read this chapter before beginning the tutorial.

The following sections summarize each chapter of the tutorial.

## Chapter 2 Defining Requirements

Determines the business requirements of the application to be developed.  
Describes how to determine the data required from the business requirements.  
Applies the rules of normalization and generalization to the list of data elements to develop an entity relationship diagram.

## Chapter 3 Data Modeling

Describes how to translate the entity relationship diagram into an CA 2E data model diagram. Describes how to enter files and field details, field validation, and default access paths created by CA 2E. Introduces involution. Describes how to create alternate access paths that use selection criteria and how to join files together.

## Chapter 4 Designing Functions

Introduces CA 2E functions (Edit File, Select Record) and how to design these in CA 2E.

## Chapter 5 Generating, Compiling, and Executing

Describes how to generate source to implement the access paths and functions created in the first four chapters and how to compile the source to produce executable i OS objects - files and programs.

## Chapter 6 Maintaining Your Application

Presents tools to help you maintain an CA 2E generated application, including prototyping, working with model object lists, and creating function versions.

## Chapter 7 Advanced Functions

Describes the use of the Edit Transaction function using the Span access path.

## Chapter 8 Report Functions

Describes how to produce a basic report program from the CA 2E design model.

## Related Information

This module provides only the information about CA 2E that is necessary to complete the Tutorial. Additional information can be found in the following list of CA 2E manuals.

**Note:** CA 2E panels provide extensive online help; simply press the Help key on your keyboard to display more information for the current panel. Be sure to use this facility as you do the tutorial.

## Getting Started

This module describes the libraries that make up CA 2E and the development environment. This module also outlines the installation and upgrade process.

## Defining a Data Model

This module provides instructions on using CA 2E to design and maintain a data model.

## **Building Access Paths**

This module provides instruction on how to build, modify, delete and document access paths and how to create arrays.

## **Building Applications**

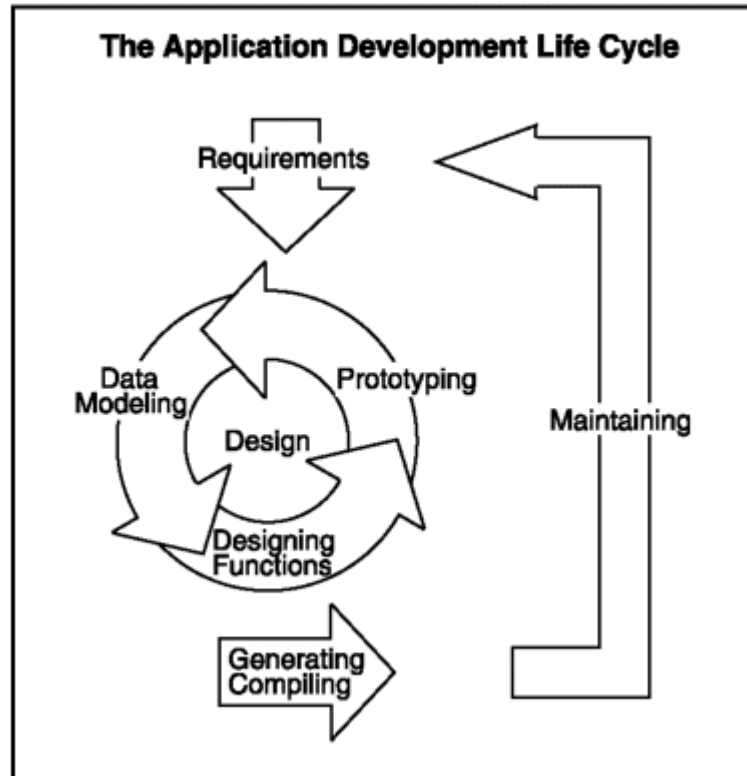
This module provides instructions on building functions in CA 2E. The module tells you how to set up system default values, build functions, edit device designs and action diagrams, and how to generate and compile functions.

## **Generating and Implementing Applications**

This module describes model objects, model object lists, and impact analysis and provides instructions and recommendations to successfully generate, compile and implement an CA 2E application.

## CA 2E and the Application Development Life Cycle

CA 2E supports the use of a top down, structured, data-driven application development life cycle. The application development life cycle is illustrated in the following diagram.



The application development life cycle provides direction during the design and generation of an CA 2E application. This tutorial provides instructions to complete each of the stages listed in the diagram.

1. Determine the application requirements.
2. Describe data in the form of an CA 2E data model. The data model is the basis for everything designed within CA 2E.
3. Design the functions that will operate on your data.
4. Prototype your functions to test your design.
5. Return to your design model to specify additional functionality.
6. Generate and compile your application.

## Prerequisites for Using CA 2E

Before starting this tutorial, make sure the following conditions are satisfied:

- Installation

CA 2E must be installed on your IBM i. This manual assumes that installation has been successfully completed. If not, please refer to the *Installation Guide* for additional details.

- Authorization

You must be signed on an IBM i with a user profile that is authorized to use CA 2E. You must also be authorized to carry out programming operations such as creating files and RPG or COBOL programs. If you do not have the required profile or authorizations, consult your security officer. Refer to the *Installation Guide* for more details. To begin, you will require access to either the Command entry (QCMD), the Programmer's menu (QPMMENU), or the CA 2E Main menu.

- Library List

Your library list must include certain libraries before you can create an CA 2E design model; for example, the CA 2E and Toolkit product libraries. Refer to the worksheet that was filled out when CA 2E was installed at your company for a list of the required libraries. The worksheet is in the *Installation Guide*. You can add any missing libraries using the i OS Add Library List Entry (ADDLIB) command. For example,

```
ADDLIB Y2SY  
ADDLIB Y1SY
```

- Creating an CA 2E Design Model

You must have an CA 2E design model library in which to store your CA 2E design model. If you do not already have a design model you may create a new one using the CA 2E Create Model Library (YCRTMDLLIB) command as follows:

```
YCRTMDLLIB MDLLIB(MYMDL) OBJPFX(MY)+  
SYSTEXT('My Model') DSNSTD(*CUATEXT)
```

This command allows you to create your model interactively. You can also do this as a batch process. In this example the design standard, \*CUATEXT, has been specified so that all functions that include interactive panel displays will be presented with either Action Bars or Windows. The normal default for the design standard is \*CUAENTRY.

**Note:** If someone else creates your model, be sure they read the above paragraph and set the DSNSTD parameter to \*CUATEXT; otherwise, the tutorial will not work properly.

The message “Model library MYMDL created” should appear following successful completion of the command. The use of this command is only necessary when you are creating a new CA 2E design model. The next topic discusses how to access an existing CA 2E design model.

The YCRTMDLLIB command also creates a library to contain the source and application objects that you will generate from your CA 2E design model. The name of this library is given by the GENLIB parameter, which in this case defaults to the name MYGEN. Furthermore it creates a Toolkit Library list with the same name as the CA 2E design model library.

■ Entering the CA 2E Design Model

To enter CA 2E, type the Start CA 2E (YSTRY2) command, specifying the name of your CA 2E design model as the LIBLST parameter from any i OS Command Entry line:

YSTRY2 MYMDL

Type the command as shown.

```
MAIN                                2E Main Menu                                System:      2EDV1
Select one of the following:

    1. User tasks
    2. Office tasks
    3. General system tasks
    4. Files, libraries, and folders
    5. Programming
    6. Communications
    7. Define or change the system
    8. Problem handling
    9. Display a menu
   10. Information Assistant options
   11. PC Support tasks

    90. Sign off

Selection or command
==> ystry2 mymdl

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F13=Information Assistant
F23=Set initial menu
```

Press Enter.

The CA 2E Main menu appears. The first panel of the CA 2E Main Menu provides a set of Design Model options and access to CA 2E commands grouped according to function.



Type **1** in the command entry line as shown to display the CA 2E Designer (\*DSNR) Menu.

```

MAIN                               2E Main Menu
Level . :   1                               System:   2EDV1

Select one of the following:

Design Model          1. Display Designer (*DSNR) menu
                       2. Display Programmer (*PGMR) menu
                       3. Display User (*USER) menu

                       8. Work with Model Object Lists
                       9. Change to work with another model

Commands            50. 2E commands in alphabetical order

                       51. Commands to set up or alter a model
                       52. Commands to copy a model
                       53. Commands to create an application
                       54. Commands to document a model

                       More...

Selection or command
===> 1

```

---

F3=Exit   F6=Messages   F9=Prev. request   F10=Cmd Entry   F14=Submitted jobs

Press Enter.

The first panel of the CA 2E Designer (\*DSNR) Menu displays. This menu provides a list of tasks available to users with \*DSNR authority. Type **1** in the command entry line as shown to enter your CA 2E design model for editing.

```

DSNR                               2E Designer (*DSNR) Menu
Level . :   2                               System:   2EDV1

Select one of the following:

Enter Model          1. Edit Database Relations
                       2. Services Menu
                       3. Edit Default Model Object List
                       4. Edit Session List (changed objects)
                       5. Work with Model Objects
                       6. Load model and display command line

                       8. Work with Model Object Lists
                       9. Change to work with another model

Open Access:       ? 10. Change Open Access Model Value
enter with *NO      11. Edit Database Relations
                       12. Services Menu

                       More...

Selection or command
===> 1

```

---

F3=Major menu   F6=Messages   F9=Prev. request   F10=Command entry   F24=More

Press Enter.

Your CA 2E design model will now be loaded. An CA 2E window with the message Starting CA 2E Session appears. The load process may take several seconds.

When it is finished the Edit Database Relations panel will appear.

```

EDIT DATABASE RELATIONS                               My Model
=> Rel lvl:
? Typ Object Relation Seq Typ Referenced object
┌───┴──────────┬──────────┬────┬──┬──────────┬──────────┐
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
│               │          │    │   │              │          │
└───┴──────────┴──────────┴────┴──┴──────────┴──────────┘

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

```

In general, you can begin working with your model at this point. However, if this is the first time you entered your tutorial model, read and complete the following instructions before continuing.

## ■ Other Methods of Entering Your Design Model

Following are two alternative methods of entering your design model.

- Go directly to the CA 2E Designer (\*DSNR) Menu using the Menu parameter on the Start CA 2E (YSTRY2) command from any i OS Command Entry line as follows:

YSTRY2 MYMDL MENU(DSNR)

- Bypass the Designer (\*DSNR) Menu and directly enter your model for editing using the Toolkit Change Library List (YCHGLIBL) command and the Edit Model (YEDTMDL) command from any i OS Command Entry line.

YCHGLIBL MYMDL  
YEDTMDL

The Y2 command is a short form of the YEDTMDL command.

YCHGLIBL MYMDL  
Y2

- Leaving the CA 2E Design Model

It is unlikely that you will complete the tutorial in a single session. If at any time you want to leave your CA 2E design model, you can do so by pressing F3 repeatedly until you return to the Edit Database Relations panel

At that point, press F3 again to display the Exit menu. Press F3 now for practice.

```

      .....
      Exit Database Relations
      .....
      Select one of the following:
      1. Exit without resynchronising
      2. Exit and resynchronise data model
      3. Return to editing
      .....
      Option: 1
      .....
      F12=Cancel
      .....
  
```

Normally when you exit your design model, you will accept the default option and press Enter. However, if you plan to continue with the tutorial, type **3** instead of accepting the default to continue your editing session and press Enter. This returns you to the Edit Database Relations panel.

- Setting Your Model Profile to Log Changes to Your Model

Your model profile establishes the basic working environment for your interactive session. CA 2E automatically creates a model profile for each user that is granted access to a model. Before beginning work on your CA 2E model, you need to check, and possibly override, the default settings in your model profile.

**Note:** You need to do this step before you enter any information for your tutorial design model and you only need to do it once.

This process assumes that the Edit Database Relations panel is displayed on your screen. If not, re-enter your model as explained previously using the Start CA 2E (YSTRY2) command.

- Accessing the Display Services Menu

One way to access your model profile is by using the Display Services Menu. This menu is an important CA 2E tool that gives you access to many CA 2E support functions while you are working on a model. From the Edit Database Relations panel, press F17 to access the Display Services Menu. This function key is available from many CA 2E panels.

From the Display Services Menu, type **11** in the Option field to Edit your model profile.

DISPLAY SERVICES MENU		My model
<b>Generation</b>	1. Submit model create request (YSEBMDLCRT) 2. Convert model data menu 3. Job list menu	
<b>Documentation</b>	6. Documentation menu 7. Convert model panel designs (YCVTMDLPNL)	
<b>Model</b>	8. Display all access paths 9. Display all functions 10. Display model values (YDSPMDLVAL) 11. Edit model profile (YEDTMDLPRF) 12. Work with model lists (YWRKMDLLST) 13. Edit model list (YEDTMDLLST *SESSION) 14. Impact analysis menu	
<b>Change Control</b>	21. Go to 2ECM menu	
Option: <u>11</u> (press F4 to prompt commands)		
F3=Exit F6=Messages F8=Submitted jobs F9=Command line F10=Display job log		

#### ■ Editing Your Model Profile

Press Enter to display the first screen of the Edit Model Profile panel.

The model profile lets you define defaults for various processes and file specifications for an interactive session. Notice that the default value for the Session list field is your user profile name. Also, check that the value of the Log changed objects field is **Y**. If it is not, type **Y**.

Edit Model Profile		
Model profile . . . : JAR		
Model . . . . . : MYMDL		
Session list . . . . .	<u>JAR</u>	Name, *MDLVAL
Log changed objects . . . . .	<u>Y</u>	Y=Yes, N=No
Component change processing . .	<u>N</u>	Y=Yes, N=No
View only . . . . .	<u>N</u>	Y=Yes, N=No
Model list for commands . . .	<u>JAR</u>	Name, *USER
User option file . . . . .	<u>QAU00PT</u>	Name, QAU00PT
Library name . . . . .	<u>*LIBL</u>	Name, *LIBL
User option member . . . . .	<u>QAU00PT</u>	Name, *FILE
Full screen mode . . . . .	<u>N</u>	Y=Yes, N=No
F3=Exit F5=Refresh F12=Cancel		More...

Press Enter.

Press F3 to return to the Display Services Menu.

- Understanding a Session List

The basic components of an CA 2E model are called *model objects*. CA 2E provides a model object list tool that lets you manipulate logical groups of model objects. One example of a model object list is the *session list*. When the Log changed objects field in your model profile is Y, CA 2E automatically keeps track of each change you make to your CA 2E model and adds the changed model object to your session list. The session list is cumulative and therefore it persists across your interactive sessions.

Notice the Edit model list (YEDTMDLLST \*SESSION), option on the Display Services Menu. You can use this option at any time to view and work with the contents of your session list; in other words, you can use this option to view the model objects you changed in your CA 2E model.

You will use the session list again later in the *Maintaining Your Application* chapter of this tutorial.

- Exiting the Display Services Menu

When you finish using the tasks on the Display Services Menu, exit by pressing F3 to return to the Edit Database Relations panel.

- Deleting User-defined Data from your CA 2E Design Model

This step applies only if you wish to delete the data you entered from your CA 2E design model. To do so use the CA 2E Clear Model (YCLRMDL) command as follows:

```
YCLRMDL MDLLIB(MYMDL) GENLIB(*GENLIB)
```

If you run this command, your CA 2E design model will still exist, but everything you have added to the design model will be deleted.



# Chapter 2: Defining Requirements

This chapter gives the business requirements and describes the data and functions required to support the Horse Racing application.

## Introduction to Defining Requirements

This tutorial demonstrates how to use CA 2E to address all phases of the life cycle in the development and maintenance of a horse race application. A sample horse racing form is illustrated below.

NEWMARKET

11Aug 1995

1:30

NEWMARKET STEEPLECHASE

Going: Heavy 2m 5f

Prize of \$6000 for five year olds and older that have never won a race

Weight: 135 lbs

			Age	Lbs	
1	500-02	FAITHFUL DOBBIN (11) (Mrs. Monica Hackett)			
		W Harney	7	130	—C F Swan
2	3523/2	PEGASUS (16) (D V Wakefield)			
		Gerard Stack	6	130	—H Rogers
3	2313/OF	LEFT BANK (3) (Mrs. S Lait)			
		Noele Meade	7	130	—D H O'Connor
4	3211-03	BONFIRE (Mr. A Mishie)			
		A Mishie	7	132	—P D Hickey
5	000-02	COOPER TRIP (Mrs. W E Fletcher)			
		W E Fletcher	6	134	—E F Hutton
6	40- 0	LARKSPUR LANDING (Ms Greenbaum)			
		A C Smith	6	128	—C J Reynolds
7	6/3 P	BLUSWAD SHUZ (E A Presley)			
		P Presley	7	135	—A A Memphis

Seven runners

Last year: Gorky Park, 5-11-7, C F Swan, 5-1 (B Hughes) , 11 ran

## Requirements Definition

Initial requirements for the application illustrated in this tutorial include the form from the preceding page. Management requires a system that tracks the horses and jockeys entered in races. In addition, they want to track the parentage and racing results of each horse and require certain reports.

## Business Requirements

After analysis of the form and further discussion with management, it is determined that the application requires the following:

- The business objects or entities: Course, Horse, Jockey, and Race. These will become files.
- A record of the Dam (mother) for each horse to ensure that it is female and older than the horse.
- A record of the Sire (father) for each horse to ensure that it is male and older than the horse.
- A record of horses entered in each race.
- The following two reports:
  - A list of all horses including a count and total value
  - A list of the races each horse has entered

## Data Requirements

Having identified the business requirements, it is necessary to examine them and determine the data needed to meet the requirements.

Analyze the form carefully to make a list of all the data needed to create the form. Do not be concerned about defining the data as a grouping of data (entity/file) or a property of the group (attribute/field). Some examples are shown here:

- Jockey
- Horse name
- Horse code
- Entry number
- Finishing position
- Race name
- Race distance
- Courses
- Course name
- Race time
- Going conditions
- Prize money
- Race date
- Dam (Horse's mother)



- Sire (Horse's father)
- Horse's date of birth

Having specified the requirements of this application, you can now begin to build the CA 2E data model. This data model will serve as the foundation for the application design in CA 2E. The data modeling process is illustrated in the *Data Modeling* chapter.



# Chapter 3: Data Modeling

---

This chapter introduces the following topics:

- Data Model
- CA 2E Relations
- Field Details and Conditions
- Extending Relations
- Virtual Fields
- Access Paths

## Data Model

In this topic you will build the CA 2E data model that supports the requirements of the horse race application.

New terms introduced:

- Entity
- Relationship
- Entity Relationship Diagram
- Attribute
- Unique Identifier
- Primary key
- Foreign key
- Normalization
- Generalization
- CA 2E Relation
- CA 2E Data Model Diagram

## Objectives

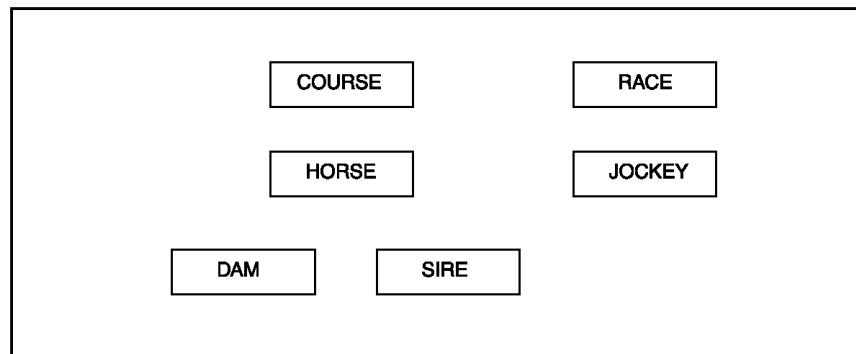
Use an Entity Relationship Diagram (ERD) to design the horse race application data model. Normalize and generalize the entities to refine the ERD. Translate the ERD into CA 2E relations. The following steps explain how to achieve these objectives.

## Identify Application Entities

The first step in entity relationship modeling is to identify the entities (files) that are present in your application using the data elements you identified during your requirements analysis. An *entity* is a thing or object of significance for which you need to gather and store information.

Each entity must be uniquely identifiable. This means that each instance (occurrence) of an entity must be separate and distinctly identifiable from all other instances of that entity. For example, for the entity, Horse, each individual horse (instance) must be uniquely identifiable in the data model.

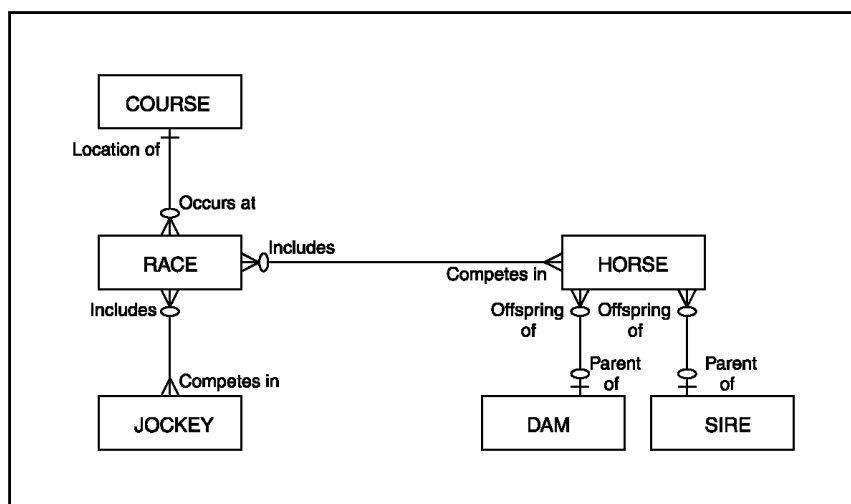
Each entity can be represented in a diagram by a box containing the name of the entity. The name is in singular and shown in all uppercase letters. This tutorial includes the following entities:



## Identify Business Relationships Between Entities

The second step in entity relationship modeling is to identify the business relationships that exist between the entities. A business relationship (or *relationship* for short) is a named, significant association between two entities.

The relationships present in the CA 2E data model at this point can be represented as shown.



## Identify Attributes and Unique Identifiers for the Entities

The third step of entity relationship modeling involves the following:

- Identifying the attributes (fields) that belong to each entity
- Selecting the attributes that serve as a unique identifier for each entity
- This unique identifier is also known as the primary key for the entity.
- Normalizing the entities
- Generalizing the entities

The following sections discuss each of these in more detail.

## Identify the Attributes for Each Entity

An *attribute* is any detail that serves to qualify, identify, classify, quantify, or express the state of an entity. An attribute can also be any description of a thing of significance. The attributes must describe the entity against which they are shown.

Reviewing the data elements you identified during requirements analysis, and taking into consideration the entities you have already identified, it is possible to identify the attributes. List the attributes beside the appropriate entity as in the following table:

Entity	Attribute
COURSE	Course code Course name
HORSE	Horse code Horse name Horse gender Date of birth
JOCKEY	Jockey code Jockey name Jockey gender
SIRE	Sire code Sire name Sire Date of birth
DAM	Dam code Dam name Dam Date of birth
RACE	Course code Course name Race date Race time Race name Going conditions Distance Prize money Horse code Horse name Horse gender Date of birth Jockey code Jockey name Finishing position Handicap Entry status

### Select Each Entity's Unique Identifier

Each entity must be uniquely identifiable so that each instance of the entity is separate and distinctly recognizable from all other instances of that entity. The *unique identifier*, also known as the *primary key*, can be any of the following:

- An attribute
- A combination of attributes
- A combination of relationships

- A combination of attributes and relationships

A primary key should consist of the fewest number of attributes necessary to make every instance of the entity unique. Review the attributes of each entity and select the primary key for each entity. Primary keys are identified in the following table:

Entity	Attribute	Primary Key
COURSE	Course code Course name	Course code
HORSE	Horse code Horse name Horse gender Date of birth	Horse code
JOCKEY	Jockey code Jockey name Jockey gender	Jockey code
SIRE	Sire code Sire name Sire Date of birth	Sire code
DAM	Dam code Dam name Dam Date of birth	Dam code
RACE	Course code Course name Race date Race time Race name Going conditions Distance Prize money Horse code Horse name Horse gender Date of birth Jockey code Jockey name Finishing position Handicap Entry status	Course code Race date Race time

## Identify Foreign Keys

Note that the RACE entity contains the primary keys of the HORSE (Horse code) and JOCKEY (Jockey code) entities. The Horse code and Jockey code are needed as attributes in the RACE entity to identify the horses and jockeys that competed in each race. However, neither the Horse code nor the Jockey code are needed to identify the RACE entity; in other words, neither Horse code nor Jockey code are primary keys of RACE. Such non-key attributes of an entity that are primary keys of another entity are known as *foreign keys*. In the following table, foreign keys are identified.

Entity	Attribute	Primary Key	Foreign Key
COURSE	Course code Course name	Course code	N/A
HORSE	Horse code Horse name Horse gender Date of birth	Horse code	N/A
JOCKEY	Jockey code Jockey name Jockey gender	Jockey code	N/A
SIRE	Sire code Sire name Sire Date of birth	Sire code	N/A
DAM	Dam code Dam name Dam Date of birth	Dam code	N/A
RACE	Course code Course name Race date Race time Race name Going conditions Distance Prize money Horse code Horse name Horse gender Date of birth Jockey code Jockey name Finishing position Handicap Entry status	Course code Race date Race time	Horse code Jockey code



## Normalize the Entities

*Normalization* of data is a procedure that ensures that a data model conforms to some useful standards.

For data and entity relationship models, the following standards have been defined to minimize duplication of data, to provide the flexibility necessary to support different functional requirements, and to enable the data model to be mapped onto a relational database design effectively.

- Eliminate repeating data groups
- Eliminate partial key dependencies
- Eliminate non-key dependencies (attributes that depend upon another non-key attribute in the entity)

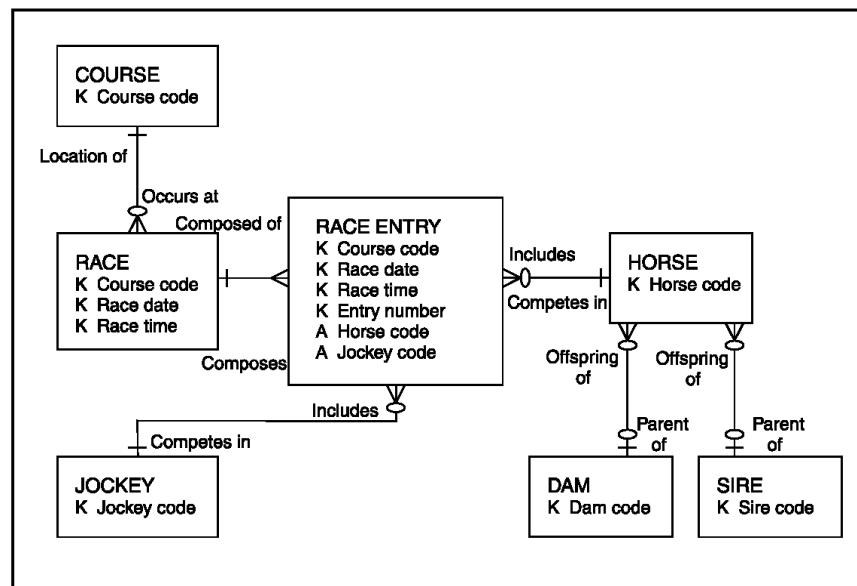
Although the steps are listed and briefly discussed here, you may wish to consult additional sources for a more detailed description.

Applying these steps causes the entities and attributes to be redefined as follows:

Entity	Attribute	Primary Key	Foreign Key
COURSE	Course code Course name	Course code	N/A
HORSE	Horse code Horse name Horse gender Date of birth	Horse code	N/A
JOCKEY	Jockey code Jockey name Jockey gender	Jockey code	N/A
SIRE	Sire code Sire name Sire Date of birth	Sire code	N/A
DAM	Dam code Dam name Dam Date of birth	Dam code	N/A
RACE	Course code Race date Race time Race name Going conditions Distance Prize money	Course code Race date Race time	N/A

Entity	Attribute	Primary Key	Foreign Key
RACE ENTRY	Course code	Course code	Horse code
	Race date	Race date	Jockey code
	Race time	Entry number	
	Entry number		
	Horse code		
	Jockey code		
	Finishing position		
	Handicap		
	Entry status		

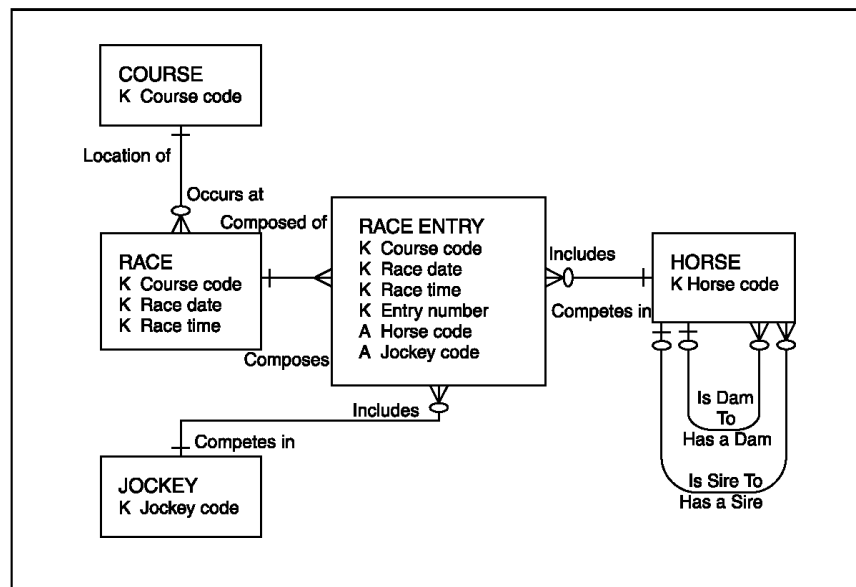
This process has resulted in the creation of another entity named RACE ENTRY. Now our Entity Relationship Diagram looks like this. Note that *K* indicates a key attribute and *A* indicates a foreign key (non-key attribute).



## Generalize the Entities

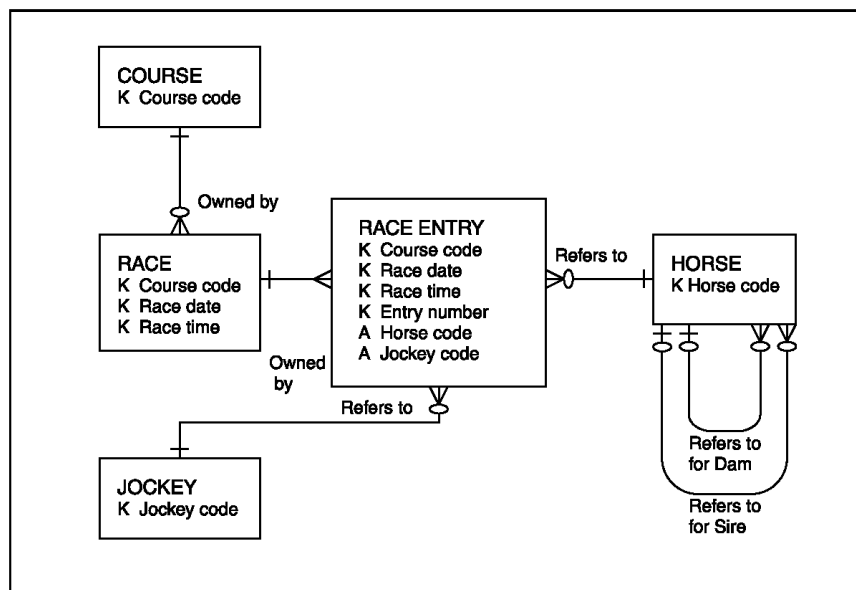
At this point, there is another source of redundant data, namely, the HORSE, DAM, and SIRE entities. A horse could appear in two of these three entities since a horse could both compete in races and also be a Dam or Sire. This situation can be resolved by combining these three entities into just one: HORSE. This process is called *entity generalization*.

The entity relationship diagram now looks like this:



## CA 2E Data Model Diagram

The relationships that are identified in the entity relationship diagram can be translated to CA 2E relations. There are eight different CA 2E relations, four of which are commonly required. The purpose of the basic relations will be explained in the course of this tutorial, and more details may be obtained from the CA 2E manuals. The CA 2E Data Model Diagram for this horse race application looks like this:



Having identified the entities (files), attributes (fields), and relationships (relations), we are ready to specify and define them within the data model. This process is illustrated in the following topics.

## CA 2E Relations

In this topic you will enter simple relations, define new CA 2E objects for use in the relations, and display the CA 2E entries arising from the relations.

New terms introduced:

- CA 2E relation
- Has relation
- Known by relation
- Owned by relation
- Refers to relation
- CA 2E model object
- file, file attribute
- field, field attribute (data type)
- file entry

New panels are introduced:

- Edit Database Relations
- Define Objects
- Edit File Entries

## Objectives

You will specify relations to define the HORSE, COURSE, JOCKEY, RACE, and RACE ENTRY entities.

## Overview of CA 2E Relations

The first step in using CA 2E is to define your basic data model. Do this by typing a number of relation statements to define the entities in your data model.

Relation statements have the form:

Object + Relation + Object

where:

*Object* is an CA 2E model object. Model objects are used to represent real world entities, such as COURSE, HORSE, or RACE. There are several different types of CA 2E model objects. Initially we need only consider two types: fields (items of data) and files. In general, the entities we identified will become files and the attributes we identified will become fields.

*Relation* is an CA 2E relation type. The CA 2E relations establish the relationships between fields and files, and files and other files.

## The Edit Database Relations Panel

When you enter your data model, the first panel you see is the Edit Database Relations panel. This panel is the top-level panel in CA 2E.

**Note:** If you have not already entered your model you can do so now by entering the following command on any i OS Command Entry line.

```
YSTRY2      MYMDL
```

Using the Edit Database Relations panel, you may enter relation statements that declare the entities in your data model. The panel has three main columns in which the relations are entered. These columns are labeled Object, Relation, and Referenced object.

EDIT DATABASE RELATIONS		My Model		
=>	Rel lvl:	Seq	Typ	Referenced object
?	Object	Relation		
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				
31				
32				
33				
34				
35				
36				
37				
38				
39				
40				
41				
42				
43				
44				
45				
46				
47				
48				
49				
50				
51				
52				
53				
54				
55				
56				
57				
58				
59				
60				
61				
62				
63				
64				
65				
66				
67				
68				
69				
70				
71				
72				
73				
74				
75				
76				
77				
78				
79				
80				
81				
82				
83				
84				
85				
86				
87				
88				
89				
90				
91				
92				
93				
94				
95				
96				
97				
98				
99				
100				

Bottom

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options  
 F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

**Note:** The Subfile selector column on the left lets you enter line selection options. The line of input-capable fields along the top of the panel lets you select which model objects within your CA 2E design model to display. Both will be explained later in this tutorial.

## Entering Relation Statements

Working top down, define the most basic entities in your data model: COURSE and RACE.

To do this you will use three different CA 2E relations:

### Known by

Declares a field that is part of the primary key that uniquely identifies a file

### Has

Declares a non-key field (attribute) on a file

### Owned by

Specifies a relationship between two files; the primary key of the owning file becomes part of the primary key of the owned file

For example, in the tutorial model, RACE is Owned by COURSE. COURSE is uniquely identified by a Course code. Since RACE is Owned by COURSE, each RACE is uniquely identified by a combination of the COURSE at which it is run, a date, and a time. As a result, the primary key for RACE consists of Course code, Race Date, and Race Time.

Type in relations as shown and use the Tab key or the Field Exit key to advance to the next field or line.

EDIT DATABASE RELATIONS		My Model	
=>	?	Rel lvl:	
Typ	Object	Relation	Seq Typ Referenced object
—	<b>Course</b>	<b>Known by</b>	<b>Course code</b>
—	<b>Course</b>	<b>Has</b>	<b>Course name</b>
—	<b>Race</b>	<b>Owned by</b>	<b>Course</b>
—	*****	<b>Known by</b>	<b>Race date</b>
—	*****	<b>Known by</b>	<b>Race time</b>
—	*****	<b>Has</b>	<b>Race name</b>
—	*****	<b>Has</b>	<b>Going conditions</b>
—	*****	*****	<b>Distance</b>
—	*****	*****	<b>Prize money</b>
—	_____	_____	_____
—	_____	_____	_____
—	_____	_____	_____
—	_____	_____	_____

Bottom

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options  
 F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

**Note:** Following are two shortcuts for entering relations.

- Press Enter after entering all the relations shown.

```

EDIT DATABASE RELATIONS                                My model
=>
?  Typ Object                                          Rel lvl:
   FIL Course                                         Relation                               Seq Typ Referenced object
   FIL Course                                         Known by                             FLD Course code
   FIL Course                                         Has                                  FLD Course name
   FIL Race                                           Owned by                             FIL Course
   FIL Race                                           Known by                             FLD Race date
   FIL Race                                           Known by                             FLD Race time
   FIL Race                                           Has                                  FLD Race name
   FIL Race                                           Has                                  FLD Going conditions
   FIL Race                                           Has                                  FLD Distance
   FIL Race                                           Has                                  FLD Prize money

```

```

EDIT DATABASE RELATIONS                               My model
=> ?                                                    Rel lvl:
Type Object                                           Relation Seq Type Referenced object
FIL Course      Known by        FLD Course code
FIL Course      Has             FLD Course name
FIL Race        Owned by        FIL Course
FIL Race        Known by        FLD Race date
FIL Race        Known by        FLD Race time
FIL Race        Has            FLD Race name
FIL Race        Has            FLD Going conditions
FIL Race        Has            FLD Distance
FIL Race        Has            FLD Prize money

Z(n)=Details   F=Functions   E(n)=Entries   S(n)=Select   Bottom
F10=Define obj F17=Services   F23=More options

```

Press F10 to display the Define Objects panel where you will enter the details needed to define the objects.

## Defining Objects

The Define Objects panel shows each of the new objects you entered on the Edit Database Relations panel.

DEFINE OBJECTS		My model		
Object type	Object name	Object attr	Referenced field	Field Edit usage field
FIL	Course			-
FLD	Course code			CDE -
FLD	Course name			ATR -
FIL	Race			-
FLD	Race time			CDE -
FLD	Race date			CDE -
FLD	Race name			ATR -
FLD	Going conditions			ATR -
FLD	Distance			ATR -
FLD	Prize money			ATR -
				-
				-
				-
				-
				- +

F3=Exit

Based on the relation statements you entered, each new object is automatically assigned one of the following types:

- FIL (for files)
- FLD (for fields)

In addition, fields are assigned a usage as follows:

- CDE (key fields)



- ATR (non-key fields)

## Specifying Object Attributes

Using the Define Objects panel, you must specify an object attribute for each new object. You can view a list of the available object attributes by typing ? in the object attribute field. Object attributes enable CA 2E to take design defaults.

- For files, the object attribute specifies a file type, which will be one of the following in this tutorial:
  - Reference file (REF) for relatively static master files
  - Capture file (CPT) for transaction files

In the example, COURSE and RACE are Reference (REF) files.

- For fields, the object attribute specifies a data type. (Another term for data type is field type.) For example, Course code is of type CDE, and Course name is of type TXT.

The data type causes CA 2E to automatically assign default characteristics to a field; for example, length and display format. You can override these defaults later.

In this tutorial you will use the following data types:

**CDE**—Alphanumeric code field

**TXT**—Descriptive text

**VAL**—Monetary value

**DT#**—ISO date

**TM#**—ISO time

**STS**—Status

**QTY**—Quantity

**NBR**—Pure numeric field value



```

EDIT DATABASE RELATIONS                               My model
=>                                                     Rel lvl:
? Typ Object                                          Relation      Seq Typ Referenced object
E FIL Course                                         Known by       FLD Course code
FIL Course                                           Has            FLD Course name
FIL Race                                             Owned by       FIL Course
FIL Race                                             Known by       FLD Race date
FIL Race                                             Known by       FLD Race time
FIL Race                                             Has            FLD Race name
FIL Race                                             Has            FLD Going conditions
FIL Race                                             Has            FLD Distance
FIL Race                                             Has            FLD Prize money

```

Bottom

Z(n)=Details    F=Functions    E(n)=Entries    S(n)=Select    F23=More options  
F10=Define object    F17=Services    F24=More keys

Press Enter to display the Edit File Entries panel. This panel shows the file entries CA 2E automatically created for the COURSE file based on the relations you entered for the file.

```

EDIT FILE ENTRIES                               My model
File . . . . . : Course

? Field              Type              Ocr  Et  Ksq GEN name      Length  Renamed
■ Course code        COE                K    1  ABCD                6
_ Course name        TXT                A    ABTX                 25

SEL: Z=Details, R=Replace field, U=Usage, M=Mapped field parameters, L=Locks,
F3=Exit

```

For each file entry, this panel displays the field names, the field type, whether the field is a key field (K) or an attribute (A), the CA 2E generated implementation name, and the field length. Note that CA 2E assigned the implementation name and default length based on the data type you specified; for example, for fields of type CDE, CA 2E assigns a 4-character implementation name ending in CD and a default length of 6.

**Note:** You cannot update fields on this panel.

Press F3 to return to the Edit Database Relations panel.



EDIT DATABASE RELATIONS				My model			
=>				Rel lvl:			
? Typ	Object			Relation	Seq	Typ	Referenced object
■ FIL	Course			Known by		FLD	Course code
— FIL	Course			Has		FLD	Course name
— FIL	Race			Owned by		FIL	Course
— FIL	Race			Known by		FLD	Race date
— FIL	Race			Known by		FLD	Race time
— FIL	Race			Has		FLD	Race name
— FIL	Race			Has		FLD	Going conditions
— FIL	Race			Has		FLD	Distance
— FIL	Race			Has		FLD	Prize money
—							
—							
—							
—							
—							
Z(n)=Details F=Functions E(n)=Entries S(n)=Select				Bottom			
F10=Define object F17=Services				F23=More options			
				F24=More keys			

Display a new panel by pressing Roll Up.

## Declaring More Files

You can define the RACE ENTRY, HORSE, and JOCKEY files using the Known by, Has, and Owned by relations as you did for the COURSE and RACE files.

In addition, since RACE ENTRY requires references to the HORSE and JOCKEY files, you will also use the Refers to relation. The Refers to relation specifies a relationship between two files. It causes the key entries of the referenced file (HORSE and JOCKEY) to become foreign keys of the referencing file (RACE ENTRY).

Note that a file may reference another file more than once, as shown later in this tutorial.

Type the **RACE ENTRY**, **HORSE**, and **JOCKEY** relations.

EDIT DATABASE RELATIONS				My model			
=>				Rel lvl:			
? Typ	Object			Relation	Seq	Typ	Referenced object
—	Race Entry			Owned by			Race
—	*****			Known by			Entry number
—	*****			Has			Finishing position
—	*****			*****			Handicap
—	*****			*****			Entry Status
—	*****			Refers to			Horse
—	*****			*****			Jockey
—	Horse			Known by			Horse code
—	*****			Has			Horse name
—	*****			*****			Horse gender
—	*****			*****			Horse value
—	*****			*****			Date of birth
—	Jockey			Known by			Jockey code
—	*****			Has			Jockey name
—	*****			*****			Jockey gender
Z(n)=Details F=Functions E(n)=Entries S(n)=Select				Bottom			
F10=Define object F17=Services				F23=More options			
				F24=More keys			

Press F10 to define the new objects.

## Defining Objects

The Define Objects panel displays with the object attribute column blank.

## File Attributes

Specify the appropriate object attributes, in order to define the new objects to CA 2E. This requires the following two additional attributes:

**CPT**—Database capture file

**NBR**—Pure numeric field value

Type the object attributes.

DEFINE OBJECTS		My model			
Object type	Object name	Object attr	Referenced field	Field Edit usage	Field Edit field
FIL	Race Entry	CPT		CDE	—
FLD	Entry number	CDE		ATR	—
FLD	Finishing position	NBR		ATR	—
FLD	Handicap	QTY		ATR	—
FLD	Entry Status	STS		ATR	—
FIL	Horse	REF			—
FIL	Jockey	REF			—
FLD	Horse code	CDE		CDE	—
FLD	Horse name	TXI		ATR	—
FLD	Horse gender	STS		ATR	—
FLD	Horse value	VAL		ATR	—
FLD	Date of birth	DT#		ATR	—
FLD	Jockey code	CDE		CDE	—
FLD	Jockey name	TXI		ATR	—
FLD	Jockey gender	STS		ATR	— +

F3=Exit

Press Enter.

If all the object attributes you specified were valid, CA 2E automatically exits the Define Objects panel and returns to the Edit Database Relations panel. You have now finished the top-level definition of your data model.

## Deleting Relations

You can delete a relation from the data model by typing **D** against the relation. Once deleted, retyping them can reinstate relations.

## Documenting Relations

To obtain a listing of the relations that you have entered, use the Document Model Relations (YDOCMDLREL) command. First press F17 from the Edit Database Relations panel to display the Display Services Menu. Then do either of the following.

- Press F9 to display a command entry line, type the **YDOCMDLREL** command, and press Enter
- Select the Documentation menu option to display a list of documentation commands and select Document model relations

When you finish, press F3 until you return to the Edit Database Relations panel.

## Field Details and Conditions

Having entered relations to define your data model, you must now add more detailed information about the fields. This includes defining simple validation rules and specifying any required overrides to the defaults CA 2E assigned for field properties; for example, field length.

This topic describes the field details and how to add field condition information to your data model.

New terms introduced:

- field condition
- VAL condition type
- LST condition
- check condition
- selection line

New panels introduced:

- Edit Field Details
- Edit Field Conditions
- Edit Field Condition Details

## Objectives

You will define conditions to specify the allowed values for the Horse gender field.

## Overview of Field Details

Field details specify the properties of a given field, such as length, text, validation, and implementation name.

## Overview of Field Conditions

Field conditions define allowed values for fields. Conditions record both the values that a field may take and the meaning that the values represent.

Conditions can be used in a number of ways. The most common uses of conditions are:

- Validating the entry of data
- Specifying selection or omission of data from a logical view of the data
- Specifying the processing conditions in a program that operates on the data

## Field Details

To obtain the field details for a field, you will select the appropriate relation statement line on the Edit Database Relations panel and type **Z2** against this object. The details of the referenced object will be displayed. Note that the Subfile selector options, Z1 and Z, both give details of the first object in the relation.

## Field Detail Display for Horse Gender

In this case, the Horse gender field of the HORSE file will be used to illustrate conditions. Type **Z2** against the HORSE Has Horse gender relation.

EDIT DATABASE RELATIONS				My model	
=>				Rel lvl:	
? Typ	Object	Relation	Seq	Typ	Referenced object
— FIL	Course	Known by	—	FLD	Course code
— FIL	Course	Has	—	FLD	Course name
— FIL	Horse	Known by	—	FLD	Horse code
— FIL	Horse	Has	—	FLD	Horse name
<u>Z2</u> FIL	Horse	Has	—	FLD	Horse gender
— FIL	Horse	Has	—	FLD	Horse value
— FIL	Horse	Has	—	FLD	Date of birth
— FIL	Jockey	Known by	—	FLD	Jockey code
— FIL	Jockey	Has	—	FLD	Jockey name
— FIL	Jockey	Has	—	FLD	Jockey gender
— FIL	Race	Owned by	—	FIL	Course
— FIL	Race	Known by	—	FLD	Race date
— FIL	Race	Known by	—	FLD	Race time
— FIL	Race	Has	—	FLD	Race name
— FIL	Race	Has	—	FLD	Going conditions

More...

Z(n)=Details   F=Functions   E(n)=Entries   S(n)=Select
F23=More options

F10=Define object   F17=Services
F24=More keys



Press Enter to display the Edit Field Details panel for Horse gender.

```

EDIT FIELD DETAILS                               My model
Field name . . . . : Horse gender                Document'n seq. . :
Type . . . . . : STS                            Field usage: ATR
Internal length. . : 1 Data type : A              GEN name: ADST
                                           K'bd shift: _ Lowercase : _
Headings . . . . . :-
Text . . . . . : Horse gender
Left hand side text. : Horse gender
Right hand side text : Value
Column headings. . . : Horse
                       gender
Control . . . . . :-
Default condition : *NONE
Check condition . : *NONE
Valid system name. : _ Mandatory fill . . . : _ Translate cnd values: _

F3=Exit no update  F8=Change name/type  F9=Conditions  F10=Appearance  F20=Har

```

The Edit Field Details panel shows information for the field. Most of the details, for example, field length, default appropriately according to the field's data type.

It is possible to override the default values if required. We will discuss how to change the field name at a later stage in this tutorial.

## Field Conditions

One use of field conditions is to check that the value entered for a field is valid during data entry. This is done by specifying a Check condition. By default, no field validation takes place. This is indicated by \*NONE for the Check condition field on the Edit Field Details panel.

A horse's gender can only be one of two values: male or female. Conditions to check for these values can be attached to the Horse gender field of the HORSE file as follows.

## Adding a Field Condition

From the Edit Field Details panel for the Horse gender field, press F9 to display the Edit Field Conditions panel. This panel shows the available conditions for the field. Initially, there are none.

```

EDIT FIELD CONDITIONS                               My model
Field name. . . . . : Horse gender                Attr. : STS
Enter condition . . : ☒                and type to add new condition.
                    type . . : ____ (Type: LST, VAL)

? Condition                                Type Op File/From value      Display/To value      MN

```

SEL: Z-Details, D-Delete, U-Where used, N-Narrative.  
F3=Exit

To add a field condition, type the name and type of the condition. In this tutorial, one value the Horse gender field can take is Stallion.

You defined Horse gender as a status field (field type STS). Status fields may only take discrete values, so the only condition types allowed for this type of field are single values (VAL) or a list of values (LST). Stallion is a single value so it is of type VAL.

Type the condition name **Stallion** with a type of **VAL**.

```

EDIT FIELD CONDITIONS
Field name . . . . . : Horse gender
Enter condition . . . : Stallion
                        type . : VAL (Type: LST, VAL)
                        Attr. : STS
                        and type to add new condition.

? Condition              Type Op File/From value      Display/To value      MN

```

Press Enter to display the Edit Field Condition Details panel.

## Field Condition Details

The Edit Field Condition Details lets you specify the actual value that is to be stored on the database file to represent a particular condition. For example, you can distinguish between the definition of the condition (Stallion) and the value used to represent the condition on file (M).

## Adding Field Condition Detail

The condition value (or Status value) for Stallion is to be represented by an M in this tutorial. Note that condition values must be unique for each field. For example, M could not represent two different gender conditions.

Type the condition value **M**.

The screenshot shows a dialog box titled "EDIT FIELD CONDITION DETAILS" with a subtitle "My model". The dialog contains the following fields and values:

- Field name . . . . : Horse gender
- Length on file . . : 1
- Attr. : STS
- Mode : \*ADD
- Condition . . . . : Stallion
- Type . . . . . : VAL
- Status value . . . : M (with a cursor over it)
- File value . . . . . : (empty)
- Mnemonic . . . . . : -

At the bottom left, there is a label "F3=Exit".

Press Enter.

After pressing Enter, the panel is redisplayed, with the message "Condition 'Stallion' added" to indicate that the condition has been successfully added.

## Adding Another Condition

In this step you will add another condition value for mares. You can overwrite the displayed values in order to enter this additional field condition.

In this case, you can overwrite Stallion with Mare, and M with F. Type the details of the next condition.

```

EDIT FIELD CONDITION DETAILS          My model
Field name . . . . . : Horse gender      Attr. : STS   Mode : *ADD
Length on file . . . : 1

Condition . . . . . : Mare
Type . . . . . : VAL

Status value . . . : File value           Mnemonic
                   F                     -

F3=Exit
Condition 'Stallion' added.

```

Press Enter to redisplay the panel with the message "Condition 'Mare' added."

Press F3 to exit and return to the Edit Field Conditions panel.

## Viewing Field Conditions

On the Edit Field Conditions panel for the Horse gender field, you should now see a list of the conditions that you have just added. Note that a special condition has been added automatically. This is the condition called \*ALL values, shown at the top of the list of conditions. The \*ALL values condition can be used when you wish to specify that a field may take any one of the individual values listed.

```

EDIT FIELD CONDITIONS          My model
Field name . . . . . : Horse gender      Attr. : STS
Enter condition . . . : (Type: LST, VAL) and type to add new condition.
type . . . : (Type: LST, VAL)

? Condition      Type Op File/From value      Display/To value      MN
- *ALL values    LST **
- Mare          VAL  F                      F
- Stallion      VAL  H                      H

SEL: Z-Details, D-Delete, U-Where used, H-Narrative.
F3=Exit

```

Press F3 to return to the Edit Field Details panel.

## Specifying a Check Condition

Notice the Check condition field on the Edit Field Details panel. This field indicates the values that are allowed for the Horse gender field. Initially the Check condition is \*NONE, indicating that any value is allowed for the field. If you specify the list condition \*ALL values, it indicates that the field must satisfy one of the particular value conditions that you have defined; in other words, it provides validation during data entry. If you specify one of the conditions that you have just defined, it imposes specific validation on the field.

You can type ? in the Check condition field, to view a selection list of existing conditions. In this tutorial, all horses must be either mares or stallions. The check condition \*ALL Values is appropriate.

Type **\*ALL values** for the Check condition field.

```

EDIT FIELD DETAILS                               My Model
Field name . . . . : Horse gender                Document'n seq. . . :
Type . . . . . : STS                            Field usage: ATR
Internal length. . : 1 Data type : A              GEN name: ADST
                                           K'bd shift: _ Lowercase : _
Headings. . . . . :-                            Old DDS name:
Text . . . . . : Horse gender
Left hand side text. : Horse gender
Right hand side text : Value
Column headings. . : Horse
                      gender
Control. . . . . :-
Default condition : *NONE
Check condition . : *ALL values
Valid system name. : _ Mandatory fill . . . : _ Translate cnd values: _

F3=Exit no update  F8=Change name/type  F9=Conditions  F10=Appearance  F20=Mar

```

Press Enter.

```

EDIT FIELD DETAILS                               My Model
Field name . . . . : Horse gender                Document'n seq. . . :
Type . . . . . : STS                            Field usage: ATR
Internal length. . : 1 Data type : A              GEN name: ADST
                                           K'bd shift: _ Lowercase : _
Headings. . . . . :-                            Old DDS name:
Text . . . . . : Horse gender
Left hand side text. : Horse gender
Right hand side text : Value
Column headings. . : Horse
                      gender
Control. . . . . :-
Default condition : *NONE
Prompt function . : Condition Value Displayer (*CVD, *DDL)
Check condition . : *ALL values
Valid system name. : _ Mandatory fill . . . : _ Translate cnd values: _

F3=Exit no update  F8=Change name/type  F9=Conditions  F10=Appearance  F20=Mar

```

Note that a Prompt function field now displays. As a result, when the end user of the compiled application positions the cursor on the Horse gender field and either enters ? or presses F4, a selection list of the conditions you defined will be displayed.

Press F3 to return to the Edit Database Relations panel.

## Exercises

Before continuing with the rest of the tutorial, complete the following exercises using the same process you used to define conditions for Horse gender.

1. Add conditions to define the allowed values for the Jockey gender field.
2. Add conditions to define the allowed values for the Going conditions field on the RACE file; for example, Firm, Good, and Soft.
3. Add conditions to define the allowed values for the Entry Status field on the RACE ENTRY file; for example, Not yet run, Finished, Scratched, and Disqualified.

## Displaying Selected Relations

Now that you have specified the relations, you can use the features of the Edit Database Relations panel to display selected objects.

On the top of the Edit Database Relations panel are fields that let you position the display at the part of the model you want to view, and let you control which CA 2E relations are displayed. The line that contains these fields is referred to as the *selection line*. Type ? in any section of the line to display the values available for selection. If you enter a letter (or string of letters) followed by an asterisk (\*) in the Object or Referenced object columns, only those CA 2E objects whose names begin with that letter (or letters) will be selected and displayed.

In the next topic, you will work with the HORSE file. You can simplify the display by selecting and displaying only those relations that define the HORSE file. Do this by typing the file name followed by an asterisk on the selection line. You will use this technique often in this tutorial.

Type **Horse\*** in the Object column of the selection line.

EDIT DATABASE RELATIONS				My model			
=>		Horse*		Rel lvl:			
?	Typ	Object		Relation	Seq	Typ	Referenced object
—	FIL	Course		Known by		FLD	Course code
—	FIL	Course		Has		FLD	Course name
—	FIL	Horse		Known by		FLD	Horse code
—	FIL	Horse		Has		FLD	Horse name
—	FIL	Horse		Has		FLD	Horse gender
—	FIL	Horse		Has		FLD	Horse value
—	FIL	Horse		Has		FLD	Date of birth
—	FIL	Jockey		Known by		FLD	Jockey code
—	FIL	Jockey		Has		FLD	Jockey name
—	FIL	Jockey		Has		FLD	Jockey gender
—	FIL	Race		Owned by		FIL	Course
—	FIL	Race		Known by		FLD	Race date
—	FIL	Race		Known by		FLD	Race time
—	FIL	Race		Has		FLD	Race name
—	FIL	Race		Has		FLD	Going conditions
More...							
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options							
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys							

Press Enter. CA 2E displays only those relations that define the HORSE file.

EDIT DATABASE RELATIONS				My model			
=>		Horse*		Rel lvl:			
?	Typ	Object		Relation	Seq	Typ	Referenced object
■	FIL	Horse		Known by		FLD	Horse code
—	FIL	Horse		Has		FLD	Horse name
—	FIL	Horse		Has		FLD	Horse gender
—	FIL	Horse		Has		FLD	Horse value
—	FIL	Horse		Has		FLD	Date of birth
—							
—							
—							
—							
—							
—							
—							
—							
—							
—							
Bottom							
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options							
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys							

**Note:** If the selection line is blank, you are returned to the beginning of the CA 2E design model when the panel is reloaded, for example, if you press F5 or Enter.

## Extending Relations

This topic further refines the CA 2E data modeling language and introduces relation extension.

New terms introduced:

- Extended relations
- Sharing

- For text
- Involution (self-referencing relation)
- Relation sequence

## Objectives

You will add relations to the HORSE file to specify each horse's parents (pedigree).

## Overview of Relation Extension

As discussed previously, one file can be associated with another by means of the Refers to relation. For example, our data model contains the RACE ENTRY Refers to HORSE, and RACE ENTRY Refers to JOCKEY relations. Note that since the referenced file in these two relations are different, there is no difficulty in distinguishing between them.

A file can also reference another file more than once. In order to distinguish among multiple references to the same file, CA 2E provides an extended form of the Refers to relation that lets you specify differences among the relations using For text.

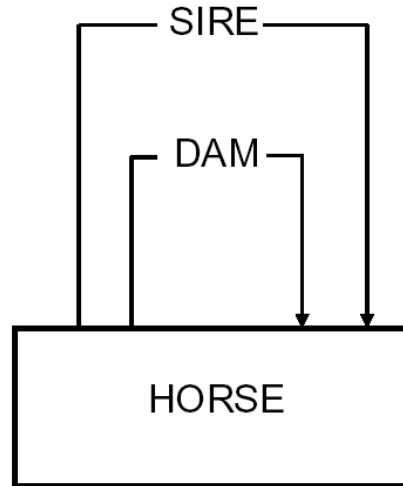
In addition, a file can refer to itself. This is known as *involution* or a *self-referencing relation*.

## Involution and the Horse Pedigrees

One of the purposes of this application is to record details of each horse's parents (Dam and Sire). To do so you will need to use both the extended form of the Refers to relation and involution.



Since a horse's parents are also horses, to record the horse's parents requires two HORSE Refers to HORSE relations, one for each parent. This is shown in the following diagram.



Because the HORSE file refers to itself more than once, you will need to use the extended form of the Refers to relation in order to distinguish between the two relations.

## Adding More Relations

If you have not already done so, type **Horse\*** on the selection line on the Edit Database Relations panel to display only the relations that define the HORSE file. Press Enter.

You are now ready to add the relations that define the horse's parents. To add references to a horse's parents type the Horse Refers to Horse relation twice; once for the Dam and once for the Sire. Type the HORSE Refers to HORSE relations.

[illegible]

Press Enter.

## Sequence of Relations

Before extending the Refers to relations for the HORSE file, you need to ensure that these two relations remain at the end of the relations for the HORSE file. The importance of this will become apparent later in the discussion of virtual fields.

Whenever the data model is reloaded, for example, when you press F5, CA 2E redisplay the relations in the following order:

- First, alphabetically by file name
- Then within each file in relation order, for example, Owned by before Known by before Refers to before Has
- Then by referenced objects in the order in which you entered them

You can override this order by entering sequence numbers to the relations in the Seq column on the Edit Database Relations panel.

You will type sequence numbers 10 to 70 so that the Refers to relations remain at the end of the list when the panel is reloaded. If you do not do this the two Refers to relations will be placed after the Known by relation instead.

Type the sequence numbers.

Press Enter.

Press F5.

EDIT DATABASE RELATIONS		My model	
=>	horse*	Rel lvl:	
? Typ	Object	Relation	Seq Typ Referenced object
■	FIL Horse	Known by	10 FLD Horse code
—	FIL Horse	Has	20 FLD Horse name
—	FIL Horse	Has	30 FLD Horse gender
—	FIL Horse	Has	40 FLD Horse value
—	FIL Horse	Has	50 FLD Date of birth
—	FIL Horse	Refers to	60 FIL Horse
	For:		Sharing: *ALL
—	FIL Horse	Refers to	70 FIL Horse
	For:		Sharing: *ALL
—			
—			
—			
—			
—			
—			
—			
Bottom			
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options			
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys			

Extending a relation introduces the two following fields:

- For field

The For field lets you specify the reason for each relation. In this tutorial, you will use this field to distinguish the two HORSE Refers to HORSE relations; in other words, HORSE Refers to HORSE for Dam, and HORSE Refers to HORSE for Sire.

- Sharing field

The entry in the Sharing field indicates whether the common keys of the two relations are to be shared. \*ALL is the default. This field applies only when the referenced file has more than one key field entry. In this tutorial, since the HORSE file has only one key field entry (Horse code), you can take the default.

## Example Showing Use of the Sharing Field

Following is a brief description of a situation where you would need to enter a value for the Sharing field. Suppose your model includes a BREED file that is Known by Breed code and that the HORSE file is Owned by BREED. The Owned by relation causes Breed code to become a key entry on HORSE giving HORSE two keys, Breed code and Horse code. To indicate that a horse can be bred only with another horse of the same breed (same Breed code), enter **\*ALL** for the Sharing field. To indicate that a horse can be bred with a horse of a different breed, enter **\*NONE** for the Sharing field.

## Adding Details of Extended Relations

You can now specify text to describe the extended relations: Dam for the first relation, and Sire for the second. Type the extended relation details, **Dam** and **Sire**.

EDIT DATABASE RELATIONS		My model	
=>	Horse*	Rel lvl:	
? Typ	Object	Relation	Seq Typ Referenced object
—	FIL Horse	Known by	10 FLD Horse code
—	FIL Horse	Has	20 FLD Horse name
—	FIL Horse	Has	30 FLD Horse gender
—	FIL Horse	Has	40 FLD Horse value
—	FIL Horse	Has	50 FLD Date of birth
—	FIL Horse	Refers to	60 FIL Horse
	For: Dam	Sharing:	*ALL
—	FIL Horse	Refers to	70 FIL Horse
	For: Sire	Sharing:	*ALL
—			
—			
—			
—			
—			
—			
Bottom			
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options			
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys			

Press Enter.

## Field Entries for HORSE

The entries on the HORSE file are automatically updated by CA 2E to include the two new Refers to relations. Examine the entries on the HORSE file by typing **E** in the Subfile selector.

EDIT DATABASE RELATIONS		My model	
=>	Horse*	Rel lvl:	
? Typ	Object	Relation	Seq Typ Referenced object
—	FIL Horse	Known by	10 FLD Horse code
—	FIL Horse	Has	20 FLD Horse name
—	FIL Horse	Has	30 FLD Horse gender
—	FIL Horse	Has	40 FLD Horse value
—	FIL Horse	Has	50 FLD Date of birth
<b>E</b>	FIL Horse	Refers to	60 FIL Horse
	For: Dam	Sharing:	*ALL
—	FIL Horse	Refers to	70 FIL Horse
	For: Sire	Sharing:	*ALL
—			
—			
—			
—			
—			
—			
Bottom			
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options			
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys			

Press Enter to display the Edit File Entries panel.

EDIT FILE ENTRIES						
My model						
File . . . . . : Horse						
? Field	Type	Ocr	Et	Ksq	GEN name	Length
— Horse code	CDE		K	1	ADCD	6
— Horse name	TXT		A		ADTX	25
— Horse gender	STS		A		ADST	1
— Horse value	VAL		A		ABVA	11.2
— Date of birth	DT#		A		ACDZ	18
— Dam Horse code	CDE REF		A		AFCD	6
— Sire Horse code	CDE REF		A		AGCD	6

SEL: Z-Details, R-Replace field, U-Usage, M-Mapped field parameters, L-Locks.  
F3=Exit

Note that the resolution of the two Refers to relations added two foreign key entries to the HORSE file, Dam Horse code and Sire Horse code. CA 2E based the definition of these fields on that of the Horse code field, and as a result, assigned the attribute CDE. REF following the attribute indicates that the field entry was resolved from a Refers to relation.

## Assigning Unique Names to Field Entries

Normally the name of a foreign key is the same as the name of the original key field; in this case Horse code. However, CA 2E files cannot contain duplicate field names. To assign unique names to the two new foreign key entries, CA 2E prefixed Horse code with the For text you entered. If you had not entered For text, CA 2E would have added a 5-digit number to Horse code to create a unique name within the HORSE file; for example, Horse code 25642.

Press F3 to return to the Edit Database Relations panel.

## Showing and Hiding Relation Extension Lines

You can use the F6 command key to toggle between showing and hiding relation extension lines. Note that the default extension lines are always hidden.

Press F6 to hide the relation extension lines for the two HORSE Refers to HORSE relations.

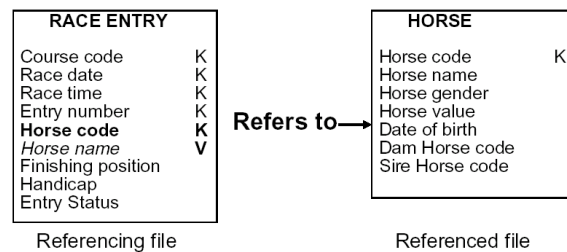
Press F6 again to show the relation extension lines.

## Specifying Virtual Fields

When a file is referenced by another file by means of a Refers to relation, entries are automatically created on the referencing file for the key fields of the referenced file. These are known as foreign keys and are shown in bold in the figure below.

To include a non-key field from the referenced file in the referencing file, you need to specify it as a virtual field. This makes the field available for use in the functions that operate upon the referencing file.

For example, to include the name of the horse when displaying the RACE ENTRY file, specify Horse name as a virtual field on the RACE ENTRY file. As a result, Horse name is physically present on the HORSE file, and is logically, but not physically, present on the RACE ENTRY file. This is shown in italics in the following figure.



Note that this same capability is available when two files are associated with each other by means of an Owned by relation. In other words, non-key fields from the owning file may be included in the owned file as virtual fields.

## Virtual Fields for Details of a Horse's Dam and Sire

The HORSE file now includes Dam Horse code and Sire Horse code as a result of the Refers to relations. In this step you will specify Horse name and Date of birth as virtual fields on the HORSE file so you can refer to the names and dates of birth for a horse's Dam and Sire.

This information is already present in the database. To retrieve this data onto the HORSE file, you need to specify the additional fields to be virtual fields on the appropriate Refers to relations.

## Sequence of Relations and Virtual Fields

With a self-referencing file such as HORSE, the sequence numbers you entered previously against the database relations are significant when specifying virtual fields.



CA 2E uses a single-pass algorithm to resolve entries on a file. When CA 2E looks for virtual fields, it resolves all the relations for the referenced file in the order that they appear on the Edit Database Relations panel. The following figure shows both the default relation order for the HORSE file and the order determined by the sequence numbers you entered earlier in this tutorial.

Default Relation Order (no sequence numbers)		User-defined Relation Order (with sequence numbers)	
Known by	Horse code	Known by	Horse code
Refers to	Horse For Dam	Has	Horse name
Refers to	Horse For Sire	Has	Horse gender
Has	Horse name	Has	Horse value
Has	Horse gender	Has	Date of birth
Has	Horse value	Refers to	Horse For Dam
Has	Date of birth	Refers to	Horse For Sire

Because the HORSE file refers to itself, any relations that occur after the Refers to HORSE relations will not have been resolved at the time CA 2E attempts to resolve the virtual fields for the Refers to relations. In other words, if you had not inserted sequence numbers for the HORSE file relations, the Has relations would not have been resolved at the time CA 2E needed to resolve the Refers to relations. As a result, CA 2E could not have created the virtual entries for the Horse name and Date of birth fields.

## Adding Virtual Fields

You can now add the Dam's name and date of birth to the list of fields for the HORSE file by specifying virtual fields. Type **V** next to the HORSE Refers to HORSE For Dam relation.

EDIT DATABASE RELATIONS		My model	
Rel	lvl:	Seq	Typ
?	Typ Object	Relation	Seq Typ Referenced object
FIL	Horse	Known by	10 FLD Horse code
FIL	Horse	Has	20 FLD Horse name
FIL	Horse	Has	30 FLD Horse gender
FIL	Horse	Has	40 FLD Horse value
FIL	Horse	Has	50 FLD Date of birth
<b>V</b> FIL	Horse	Refers to	60 FIL Horse
	For: Dam		Sharing: *ALL
FIL	Horse	Refers to	70 FIL Horse
	For: Sire		Sharing: *ALL

Bottom

Z(n)=Details   F=Functions   E(n)=Entries   S(n)=Select   F23=More options  
 F3=Exit   F5=Reload   F6=Hide/Show   F7=Fields   F9=Add/Change   F24=More keys

Press Enter to display the Edit Virtual Field Entries panel.

## Virtual Field Entries

The Edit Virtual Field Entries panel shows all the fields on the referenced file. Since the HORSE file refers to itself, these are the same as the fields on the referencing file. From this list, you can select any field as a virtual field.

## Selecting Virtual Fields for Dam

The Horse name and Date of birth fields will become virtual fields for the Refers to For Dam relation on the HORSE file. Type **+** against Horse name and Date of birth.

EDIT VIRTUAL FIELD ENTRIES									
My model									
File . . . . . : Horse									
? V	Field	Type	Ocr	Etp	Ksq	GEN name	Length	Renamed	
	Horse code	CDE		K	1	ADCD	6		
±	Horse name	TXT		A		ADTX	25		
-	Horse gender	STS		A		ADST	1		
-	Horse value	VAL		A		ABVA	11.2		
±	Date of birth	DT#		A		ACDZ	10		
█	Dam Horse code	REF		A		AFCD	6		
-	Sire Horse code	REF		A		AGCD	6		

SEL: '+' Add virtual field, '-' Remove virtual field.  
F3=Exit

Press Enter.

## Confirming Virtual Fields for Dam

The virtual fields are indicated by an asterisk (\*) in the V column of the panel confirming that CA 2E has created the virtual fields.

Press F3 to return to the Edit Database Relations panel.

The procedure for adding virtual fields for the horse's Sire is the same as for the horse's Dam. Type **V** against the HORSE Refers to HORSE For Sire relation.

```

EDIT DATABASE RELATIONS                               My model
=>   Horse*                                           Rel lvl:
? Typ Object                                          Relation      Seq Typ Referenced object
_ FIL Horse                                         known by     10 FLD Horse code
_ FIL Horse                                         Has         20 FLD Horse name
_ FIL Horse                                         Has         30 FLD Horse gender
_ FIL Horse                                         Has         40 FLD Horse value
_ FIL Horse                                         Has         50 FLD Date of birth
_ FIL Horse                                         Refers to    60 FIL Horse

For: Dam                                             Sharing: *ALL
V FIL Horse                                         Refers to    70 FIL Horse
For: Sire                                            Sharing: *ALL

_ 
_ 
_ 
_ 
_ 

Bottom
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options
F3=Exit F5=Rload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

```

## Fields for Sire

Chapter 3: Data Modeling 3-41

EDIT VIRTUAL FIELD ENTRIES

My model

File . . . . . : Horse

? V	Field	Type	Ocr	Etp	Ksq	GEN name	Length	Renamed
	Horse code	CDE	K	1	ADCD		6	
±	Horse name	TXT	A		ADTX		25	
-	Horse gender	STS	A		ADST		1	
-	Horse value	VAL	A		ABVA		11.2	
+	Date of birth	DT#	A		ACDZ		10	
■	Dam Horse code	REF	A		AACD		6	
-	Dam Horse name	REF	V		AFTX		25	Y
-	Dam Date of birth	REF	V		ADDZ		10	Y
-	Sire Horse code	REF	A		AGCD		6	

SEL: '+' Add virtual field, '-' Remove virtual field.

F3=Exit

Press Enter.

## Confirming Virtual Fields for Sire

CA 2E has confirmed your selection of virtual fields, as indicated by the asterisks.

EDIT VIRTUAL FIELD ENTRIES		My model							
File . . . . . : Horse									
?	V	Field	Type	Ocr	Etp	Ksq	GEN name	Length	Renamed
		Horse code	CDE	K	1	ADCD		6	
■	*	Horse name	TXT	A		ADTX		25	
—		Horse gender	STS	A		ADST		1	
—		Horse value	VAL	A		ABVA		11.2	
—	*	Date of birth	DT#	A		ACDZ		10	
—		Dam Horse code	REF	A		AACD		6	
—		Dam Horse name	REF	V		AFTX		25	V
—		Dam Date of birth	REF	V		ADDZ		10	V
—		Sire Horse code	REF	A		AGCD		6	

SEL: '+' Add virtual field, '-' Remove virtual field.  
F3=Exit

Press F3 to return to the Edit Database Relations panel.

## File Entries

Check the file entries for the HORSE file. To display the file entries, type **E** next to one of the HORSE file relations.

Press Enter to display the Edit File Entries panel.

Note that the virtual fields you just specified are included as indicated by a V in the Entry type (Et) column. CA 2E has defined them by reference to the original fields and has named them by prefixing the For text to the original name, for example, Dam Horse name.

You can change the names that CA 2E has given the virtual fields to make them more explicit or compact. For example, you could change Dam Horse name to Dam name, and Sire Horse name to Sire name.

EDIT FILE ENTRIES		My model				
File . . . . . : Horse						
? Field	Type	Ocr	Et	Ksq	GEN name	Length Renamed
— Horse code	CDE		K	1	ADCD	6
— Horse name	TXT		A		ADTX	25
— Horse gender	STS		A		ADST	1
— Horse value	VAL		A		ADVA	11.2
— Date of birth	DT#		A		ACDZ	18
— Dam Horse code	CDE REF		A		AFCO	6
Z Dam Horse name	TXT REF		V		AFTX	25 Y
— Dam Date of birth	DT# REF		V		ADDZ	18 Y
— Sire Horse code	CDE REF		A		AGCO	6
Z Sire Horse name	TXT REF		V		AGTX	25 Y
■ Sire Date of birth	DT# REF		V		AEDZ	18 Y

SEL: Z=Details, R=Replace field, U=Usage, M=Mapped field parameters, L=Locks.  
F3=Exit

Press Enter to display the Edit Field Details panel for the first field you selected.

## Field Details

The Edit Field Details panel lets you view details for the field, Dam Horse name. Some of the attributes of the field, such as the name, are initially protected. The F8 key unprotects the field name so it can be changed.

EDIT FIELD DETAILS		My model	
Field name . . . . .	Dam Horse name	Document'n seq. . .	1
Type . . . . .	REF Ref: TXT Horse name	Field usage:	ATR
Internal length. . .	25 Data type : A	GEN name:	ATX
	K'bd shift: Lowercase : Y		
Headings . . . . . :-		Old DDS name:	
Text . . . . .	Dam Horse name		
Left hand side text:	Dam Horse name		
Right hand side text:	Text		
Column headings. . .	Dam Horse name		
Control . . . . . :-			
Default condition :	*NONE		
Check condition . .	*NONE		
Valid system name. .	Mandatory fill . . .		

F3=Exit no update F8=Change name/type F9=Conditions F10=Appearance F20=Har

Press F8 to change the field name.

## Entering a New Field Name

The cursor will now be on the Field name entry on the Edit Field Details panel. Rename the field by typing the new name over the existing name. Type **Dam name** over Dam Horse name.

```

EDIT FIELD DETAILS                               My model
Field name . . . . : Dam name Document'n seq. . :
Type . . . . . : REF Ref: TXT Horse name Field usage: ATR
Internal length. . : 25 Data type : A GEN name: AFTX
                  K'bd shift: Lowercase : Y
Headings. . . . . :-
Text . . . . . : Dam Horse name Old DDS name:
Left hand side text. : Dam Horse name
Right hand side text : Text
Column headings. . . : Dam Horse name

Control . . . . . :-
Default condition : *NONE

Check condition . : *NONE
Valid system name. : Mandatory fill . . . :

F3=Exit no update F8=Change name/type F9=Conditions F10=Appearance F20=Har

```

Press Enter.

## Field Details After Renaming

The Edit Field Details panel will be displayed with the new field name inserted throughout. You can rename any field in this way, and the new name will be reflected throughout your design model.

```

EDIT FIELD DETAILS                               My model
Field name . . . . : Dam name Document'n seq. . :
Type . . . . . : REF Ref: TXT Horse name Field usage: ATR
Internal length. . : 25 Data type : A GEN name: AFTX
                  K'bd shift: Lowercase : Y
Headings. . . . . :-
Text . . . . . : Dam name Old DDS name:
Left hand side text. : Dam name
Right hand side text : Text
Column headings. . . : Dam name

Control . . . . . :-
Default condition : *NONE

Check condition . : *NONE
Valid system name. : Mandatory fill . . . :

F3=Exit no update F8=Change name/type F9=Conditions F10=Appearance F20=Har

```

## Renaming Other Fields

Press Enter to redisplay the Edit Field Details panel for the Sire horse name field. Recall that this was the second field you selected for renaming on the Edit File Entries panel.

## Exercise

Change Sire Horse name to **Sire name** following the same steps as shown previously.

## Displaying File Entries Again

When you have renamed the selected fields, press Enter to return to the Edit File Entries panel. The panel has been updated to show the new field names.

EDIT FILE ENTRIES		My model				
File . . . . . : Horse						
? Field	Type	Ocr	Et Ksq	GEN name	Length	Renamed
■ Horse code	CDE	K	L	ADCD	6	
- Horse name	TXT	A		ADTX	25	
- Horse gender	STS	A		ADST	1	
- Horse value	VAL	A		ABVA	11.2	
- Date of birth	DT#	A		ACDZ	10	
- Dam Horse code	CDE REF	A		AFCD	6	
- Dam name	TXT REF	V		AFIX	25	Y
- Dam Date of birth	DT# REF	V		ADDZ	10	Y
- Sire Horse code	CDE REF	A		AGCD	6	
- Sire name	TXT REF	V		AGTX	25	Y
- Sire Date of birth	DT# REF	V		AEDZ	10	Y

SEL: Z-Details, R-Replace field, U-Usage, M-Mapped field parameters, L-Locks.  
F3=Exit

Press F3 to return to the Edit Database Relations panel.

## Adding Virtual Fields for the Race and Race Entry Files

Having added virtual fields to the HORSE file, you may specify virtual fields on other file-to-file relations; for example, on the RACE and RACE ENTRY files. Type **Race\*** on the selection line.



```

EDIT DATABASE RELATIONS                               My model
=>      Race*                                         Rel lvl:
? Typ Object                                          Relation          Seq Typ Referenced object
_ FIL Horse     Known by         10 FLD Horse code
_ FIL Horse     Has              20 FLD Horse name
_ FIL Horse     Has              30 FLD Horse gender
_ FIL Horse     Has              40 FLD Horse value
_ FIL Horse     Has              50 FLD Date of birth
_ FIL Horse     Refers to        60 FIL Horse
    For: Dam                                     Sharing: *ALL
_ FIL Horse     Refers to        70 FIL Horse
    For: Sire                                    Sharing: *ALL
_____
_____
_____
_____
_____
_____
Bottom
Z(n)=Details   F=Functions   E(n)=Entries   S(n)=Select   F23=More options
F3=Exit       F5=Rload      F6=Hide>Show   F7=Fields      F9>Add/Change  F24=More keys

```

Press Enter to display relations for the RACE and RACE ENTRY files.

In this step you will define Course name as a virtual field on the RACE file and Horse name and Jockey name as virtual fields on the RACE ENTRY file. Type **V** against the three relations.

```

EDIT DATABASE RELATIONS                               My model
=>  Race*                                           Rel lvl:
?  Typ Object                                     Relation      Seq Typ Referenced object
Y  FIL Race                                       Owned by        FIL Course
FIL Race                                       Known by        FLD Race date
FIL Race                                       Known by        FLD Race time
FIL Race                                       Has             FLD Race name
FIL Race                                       Has            FLD Going conditions
FIL Race                                       Has            FLD Distance
FIL Race                                       Has            FLD Prize money
FIL Race Entry                               Owned by        FIL Race
FIL Race Entry                               Known by        FLD Entry number
Y  FIL Race Entry                               Refers to       FTL Horse
Y  FIL Race Entry                               Refers to       FTL Jockey
FIL Race Entry                               Has            FLD Finishing position
FIL Race Entry                               Has            FLD Handicap
FIL Race Entry                               Has            FLD Entry Status

```

Bottom

Z(n)=Details    F=Functions    E(n)=Entries    S(n)=Select    F23=More options  
F3=Exit    F5=Reload    F6=Hide/Show    F7=Fields    F9=Add/Change    F24=More keys

Press Enter.

## Adding More Virtual Entries

The allowed virtual field entries from the COURSE file are displayed. Type + against Course name to add it as a virtual entry to the RACE file.

```

EDIT VIRTUAL FIELD ENTRIES          My model
File . . . . . : Course
?  V  Field                        Type Dcr Etp Ksq GEN name  Length  Renamed
*      Course code                  CDE      K    1 ABCD           6
      Course name                   TXT      A    ABTX           25

SEL: '+' Add virtual field, '-' Remove virtual field.
F3=Exit

```

Press Enter. The Edit Virtual Field Entries panel redisplay to confirm the creation of the virtual field as indicated by the '\*' in the V column.

Press F3. CA 2E displays the Edit Virtual Field Entries panel for the HORSE file. Recall that you typed **V** against the RACE ENTRY Refers to HORSE relation on the Edit Database Relations panel.

```

EDIT VIRTUAL FIELD ENTRIES          My model
File . . . . . : Horse
?  V  Field                        Type Dcr Etp Ksq GEN name  Length  Renamed
*      Horse code                   CDE      K    1 ABCD           6
*      Horse name                   TXT      A    ADTX           25
-      Horse gender                 STS      A    ADST           1
-      Horse value                  VAL      A    ABVA          11.2
-      Date of birth                 DT#      A    ACBZ           10
-      Dam Horse code               REF      A    AFCD           6
-      Dam name                     REF      V    AFTX           25      V
-      Dam Date of birth            REF      V    ADDZ           10      V
-      Sire Horse code              REF      A    AGCD           6
-      Sire name                    REF      V    AGTX           25      V
-      Sire Date of birth           REF      V    AEDZ           10      V

SEL: '+' Add virtual field, '-' Remove virtual field.
F3=Exit

```

## Exercise

Add Horse name and Jockey name as virtual fields to the RACE ENTRY file. Use the same method you just used to add Course name as a virtual field to the RACE file. When you finish, press F3 to return to the Edit Database Relations panel.

## CA 2E Access Paths

This topic introduces CA 2E access paths and presents some of their features such as access path selection and access path relations. You are also shown how to generate source code for the database files to implement CA 2E access paths.

New terms introduced:

- access path
- retrieval (RTV) access path
- based-on file
- access path selection
- select/omit set
- access path relation
- referenced access path

New panels introduced:

- Edit File Details
- Edit Access Path Details
- Edit Access Path Select/Omit
- Edit Access Path Conditions
- Edit Access Path Relations
- Display File Access Paths
- Virtualize Access Path

## Objectives

In this topic you will:

- Accept the default access paths for the COURSE file.
- Add virtual fields to the default Retrieval access path for the HORSE file.
- Add two access paths to the HORSE file to select either female horses (Mares) or male horses (Stallions).
- Modify the default Retrieval access path for the HORSE file to ensure that Dams are female and Sires are male.

## Overview of Access Paths

The next stage in defining your design model is to define the required access paths. Access paths specify views of the files in the data model. The views define how data is to be presented to the functions. Functions specify the processes that operate on the data, see the chapter "Designing Functions" for more information.

Access paths control three different aspects of how data is presented to a function. An access path determines:

- Which fields will be available to the function
- Which records will be selected or omitted from the file (select/omit)
- The order in which records are presented (key sequence) to the function

## Default Access Paths

CA 2E uses the relations in the data model to create a default set of access paths for each file. In many cases these will be sufficient. You may override the defaults or define additional access paths.

Three default access paths are automatically created for every file:

- A physical (PHY) access path (un-keyed). It contains the address of all data stored physically within a file and stored in the order in which the data was written to the file.
- An update (UPD) access path (logical view). It is used to update the file and contains all fields defined for the file. It cannot be altered.
- A retrieval (RTV) access path (logical view). It specifies a view of the data that CA 2E generated programs use to retrieve records from a file. Each file has at least one.

## View Default Access Paths for COURSE

Type **Course\*** on the selection line to display only relations for the COURSE file.



EDIT FILE DETAILS		My model																									
File name . . . . .	: Course	Field reference file. :	*NONE																								
Attribute . . . . .	: REF	Source library. . . . .	: MYGEN																								
Documentation sequence. . .		Distributed . . . . .	: N (V,N)																								
GEN format prefix . . . . .	: AB																										
Assimilated physical. . . .																											
Record not found message. :	Course	NF Msgid. :	USR0001																								
Record exists message . . .	Course	EX Msgid. :	USR0002																								
<table border="1"> <thead> <tr> <th>? Typ</th> <th>Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>PHY</td> <td>Physical file</td> <td>MYABREP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>UPD</td> <td>Update index</td> <td>MYABREL0</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>RTV</td> <td>Retrieval index</td> <td>MYABREL1</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> </tbody> </table>				? Typ	Access path	Source mbr	Key	Index options	Auto add	PHY	Physical file	MYABREP	NONE		ATR ONLY	UPD	Update index	MYABREL0	UNIQUE IMMED		ATR ONLY	RTV	Retrieval index	MYABREL1	UNIQUE IMMED		ATR ONLY
? Typ	Access path	Source mbr	Key	Index options	Auto add																						
PHY	Physical file	MYABREP	NONE		ATR ONLY																						
UPD	Update index	MYABREL0	UNIQUE IMMED		ATR ONLY																						
RTV	Retrieval index	MYABREL1	UNIQUE IMMED		ATR ONLY																						
SEL: Z=Details, G/J=Generate, E=STRSEU, D=Delete, L=Locks, O=Overrides H=Hold/Release, T=Trim, V=Virtualize, U=Usage, F=Func refs., N=Narrative F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative																											

This panel shows the three default access paths that CA 2E automatically created for the COURSE file.

Press F3 to return to the Edit Database Relations panel.

## Access Paths for the HORSE File

Display the relations for the HORSE file from the Edit Database Relations panel by typing **Horse\*** on the selection line and pressing Enter. To display the access paths for the HORSE file, zoom into one of the relations for the file on the Edit Database Relations panel. Type **Z** against one of the relations.

EDIT DATABASE RELATIONS		My model	
=>	<b>Horse*</b>	Rel lvl:	
? Typ	Object	Relation	Seq Typ Referenced object
<b>Z</b>	FIL <b>Horse</b>	<b>Known by</b>	<b>10 FLD Horse code</b>
—	FIL <b>Horse</b>	<b>Has</b>	<b>20 FLD Horse name</b>
—	FIL <b>Horse</b>	<b>Has</b>	<b>30 FLD Horse gender</b>
—	FIL <b>Horse</b>	<b>Has</b>	<b>40 FLD Horse value</b>
—	FIL <b>Horse</b>	<b>Has</b>	<b>50 FLD Date of birth</b>
—	FIL <b>Horse</b>	<b>Refers to</b>	<b>60 FIL Horse</b>
	For: <b>Dam</b>		Sharing: *ALL
—	FIL <b>Horse</b>	<b>Refers to</b>	<b>70 FIL Horse</b>
	For: <b>Sire</b>		Sharing: *ALL
—			
—			
—			
—			
—			
Bottom Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys			

Press Enter to display the default access paths for the HORSE file.

## Adding Virtual Fields to the Retrieval Access Path

Each access path initially contains all of the relations for the file on which it is based but none of the virtuals. When you add a new relation to a file, the effect on the access path is controlled by the value in the Auto add column of the Edit File Details panel. The default is ATR ONLY, which automatically adds only attributes to the access path. As a result, if you want virtual fields included on an access path, you need to add them explicitly.

In this step you will add the virtual fields you just defined for the HORSE file to the default Retrieval access path.

**Note:** Adding virtual fields to an access path increases processing overhead. As a result, in a real-world model you would not add virtual fields to the default retrieval access path. Instead, create a new RTV access path and add the virtual fields to it. Assign a name to the new access path that indicates it contains virtual fields; for example, Retrieval with virtuals. You will learn how to add a new access path later in this topic.

Type **V** in the Subfile selector for the Retrieval index access path.

EDIT FILE DETAILS		My model	
File name . . . . .	Horse	Field reference file. :	*NONE
Attribute . . . . .	REF	Source library. . . . .	MYGEN
Documentation sequence. . .		Distributed . . . . .	H (Y,N)
GEN format prefix . . . .	AE		
Assimilated physical. . . .			
Record not found message. :	Horse	HF Msgid. :	USR0007
Record exists message . . :	Horse	EX Msgid. :	USR0008
? Typ Access path			
PHY Physical file	MYAERE0	NONE	ATR ONLY
UPD Update index	MYAERE0	UNIQUE IMMED	ATR ONLY
<b>V</b> RTV Retrieval index	MYAERE1	UNIQUE IMMED	ATR ONLY
SEL: Z=Details, G/J=Generate, E=STRSEU, D=Delete, L=Locks, O=Overrides			
H=Hold/Release, T=Trim, V=Virtualize, U=Usage, F=Func refs., N=Narrative			
F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative			

Press Enter to view virtual entries for the HORSE file.

VIRTUALIZE ACCESS PATH		My Model					
File name . . . . .	Horse					Attribute . . .	REF
Access path . . . . .	Retrieval index					Type . . . . .	RTV
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Dam name	TXT REF		V		AFTX	25	Y
Dam Date of birth	DT# REF		V		ADDZ	10	Y
Sire name	TXT REF		V		AGTX	25	Y
Sire Date of birth	DT# REF		V		AEDZ	10	Y
F3=Exit, no update ENTER=Validate							

Press Enter to validate the addition of the virtual fields to the Retrieval index access path for the HORSE file.

VIRTUALIZE ACCESS PATH		My model					
File name . . . . .	Horse					Attribute . . .	REF
Access path . . . . .	Retrieval index					Type . . . . .	RTV
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Dam name	TXT REF		V		AFTX	25	Y
Dam Date of birth	DT# REF		V		ADDZ	10	Y
Sire name	TXT REF		V		AGTX	25	Y
Sire Date of birth	DT# REF		V		AEDZ	10	Y
F3=Exit, no update ENTER=Validate							
CONFIRM: <input checked="" type="checkbox"/> (Y,N)							

Press Enter again to respond to the Confirm prompt in the lower right-hand corner of your screen and return to the Edit File Details panel.

To view the entries for the default Retrieval access path, type **Z** in the Subfile selector for the Retrieval index.



EDIT FILE DETAILS		My model																																																							
File name . . . . .	: Horse	Field reference file . .	: *NONE																																																						
Attribute . . . . .	: REF	Source library . . . . .	: MYGEN																																																						
Documentation sequence . .		Distributed . . . . .	: N (V,N)																																																						
GEN format prefix . . . . .	: AE																																																								
Assimilated physical . . .																																																									
Record not found message .	: Horse	NF Msgid . . . . .	: USR0007																																																						
Record exists message . . .	: Horse	EX Msgid . . . . .	: USR0008																																																						
<table border="1"> <thead> <tr> <th>? Typ</th> <th>Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>-</td> <td>PHY Physical file</td> <td>MYAREP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>-</td> <td>UPD Update index</td> <td>MYAREL0</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>Z</td> <td>RTV Retrieval index</td> <td>MYAREL1</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>█</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>				? Typ	Access path	Source mbr	Key	Index options	Auto add	-	PHY Physical file	MYAREP	NONE		ATR ONLY	-	UPD Update index	MYAREL0	UNIQUE IMMED		ATR ONLY	Z	RTV Retrieval index	MYAREL1	UNIQUE IMMED		ATR ONLY	█						-						-						-						-					
? Typ	Access path	Source mbr	Key	Index options	Auto add																																																				
-	PHY Physical file	MYAREP	NONE		ATR ONLY																																																				
-	UPD Update index	MYAREL0	UNIQUE IMMED		ATR ONLY																																																				
Z	RTV Retrieval index	MYAREL1	UNIQUE IMMED		ATR ONLY																																																				
█																																																									
-																																																									
-																																																									
-																																																									
-																																																									
SEL: Z=Details, G/J=Generate, E=STRSEU, D=Delete, L=Locks, O=Overrides H=Hold/Release, T=Trim, V=Virtualize, U=Usage, F=Func refs., N=Narrative F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative																																																									

Press Enter to display the Edit Access Path Details panel.

Type **Z** again as shown to view entries for the Retrieval index access path.

EDIT ACCESS PATH DETAILS		My model													
File name . . . . .	: Horse	Attribute . . . . .	: REF												
Access path name . . . . .	: Retrieval index	Type . . . . .	: RTV												
Unique or duplicate order .	: U (U-Unique, F-FIFO, L-LIFO, C-FCFO, ' '-Undefined)														
Index maintenance option .	: I (I-IMMED, D-DLV, R-REBLD)														
Alternate collating table .															
Allow select/omit . . . . .	: _ (S-Static, D-Dynamic, ' '-None)														
Generation mode . . . . .	: M (M-MDLVAL, D-DDS, S-SQL, X-UNX)														
Source member name . . .	: MYAREL1														
Source member text . . .	: Horse	Retrieval index													
<table border="1"> <thead> <tr> <th>Format</th> <th>GEN</th> <th>Format text</th> <th>Associated</th> </tr> </thead> <tbody> <tr> <td>? Seq name</td> <td>pfx</td> <td>(Based on file)</td> <td>Update access path</td> </tr> <tr> <td>Z 1</td> <td>MYAREAK</td> <td>AE Horse</td> <td>Update index</td> </tr> </tbody> </table>				Format	GEN	Format text	Associated	? Seq name	pfx	(Based on file)	Update access path	Z 1	MYAREAK	AE Horse	Update index
Format	GEN	Format text	Associated												
? Seq name	pfx	(Based on file)	Update access path												
Z 1	MYAREAK	AE Horse	Update index												
SEL: Z=Entries, R=Relations, S=Select/omit, A=Assoc.acps, T=Trim, V=Virtualize F3=Exit F8=Rename F20=Narrative															

Press Enter to display the Edit Access Path Format Entries panel.

EDIT ACCESS PATH FORMAT ENTRIES				My model			
File name	:	Horse		Attribute	:	REF	
Access path name	:	Retrieval index		Type	:	RTV	
Format text	:	Horse					
Based on	:	Horse		Format No	:	1	

? Field		GEN	Name	Type	Key	Altcol	Ref
					no.	Dsc	seq
Horse code	CDE	ADCD	K	1			1
Horse name	TXT	ADTX	A				1
Horse gender	STS	ADST	A				1
Horse value	VAL	ABVA	A				1
Date of birth	DT#	ACDZ	A				1
Dam Horse code	CDE REF	AFCO	A				1
Dam name	TXT REF	AFTX	V				1
Dam Date of birth	DT# REF	ADDZ	V				1
Sire Horse code	CDE REF	AGCO	A				1
Sire name	TXT REF	AGTX	V				1
Sire Date of birth	DT# REF	AEDZ	V				1

SEL: Z-Field details, L-Locks.  
F3=Exit F7=Relations

Verify that the virtual fields you specified are listed. Virtual fields are indicated by a **V** in the Type column. Press F13 to return to the Edit File Details panel.

## Adding New Access Paths for HORSE

In this step you will add some new access paths as well as update the default Retrieval (RTV) access path. This will demonstrate some of the more sophisticated capabilities of access paths.

For example, you can define an access path to select only certain relations from the file or to include selection criteria based on conditions specified for certain fields. In this tutorial, you will specify new access paths that select only mares or only stallions from the HORSE file. Call these new RTV access paths Mares and Stallions.

Type the names of the new RTV access paths, **Mares** and **Stallions**.

EDIT FILE DETAILS				My model			
File name	:	Horse		Field reference file	:	*NONE	
Attribute	:	REF		Source library	:	MYGEN	
Documentation sequence	:			Distributed	:	N (V,N)	
GEN format prefix	:	AE					
Assimilated physical	:						
Record not found message	:	Horse		NF Msgid	:	USR0007	
Record exists message	:	Horse		EX Msgid	:	USR0008	

? Typ	Access path	Source mbr	Key	Index	options	Auto add
PHY	Physical file	MYAERE	NONE			ATR ONLY
UPD	Update index	MYAERE0	UNIQUE	IMMED		ATR ONLY
RTV	Retrieval index	MYAERE1	UNIQUE	IMMED		ATR ONLY
RTV	Mares					
RTV	Stallions					

SEL: Z-Details, G/J-Generate, E-STRSEU, D-Delete, L-Locks, O-Overrides  
H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative  
F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative

Press Enter.

## Confirming Addition of Access Paths

You have now added the two new access paths. CA 2E immediately assigns a source member name and default properties to the new access paths.

## Access Path Details

The RTV type access path for Mares must select only female horses from the HORSE file. To specify this condition, type **Z** against the Mares Retrieval access path.

EDIT FILE DETAILS		My model																																					
File name . . . . .	Horse	Field reference file . .	*NONE																																				
Attribute . . . . .	REF	Source library . . . . .	MYGEN																																				
Documentation sequence . .		Distributed . . . . .	N (V,N)																																				
GEN format prefix . . . .	AE																																						
Assimilated physical . . .																																							
Record not found message .	Horse	HF Msgid . .	USR0007																																				
Record exists message . .	Horse	EX Msgid . .	USR0008																																				
<table border="1"> <thead> <tr> <th>? Typ</th> <th>Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>-</td> <td>PHY Physical file</td> <td>MYAREP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>-</td> <td>UPD Update index</td> <td>MYAREL0</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>-</td> <td>RTV Retrieval index</td> <td>MYAREL1</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>Z</td> <td>RTV Mares</td> <td>MYAREL2</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>█</td> <td>RTV Stallions</td> <td>MYAREL3</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> </tbody> </table>				? Typ	Access path	Source mbr	Key	Index options	Auto add	-	PHY Physical file	MYAREP	NONE		ATR ONLY	-	UPD Update index	MYAREL0	UNIQUE IMMED		ATR ONLY	-	RTV Retrieval index	MYAREL1	UNIQUE IMMED		ATR ONLY	Z	RTV Mares	MYAREL2	UNIQUE IMMED		ATR ONLY	█	RTV Stallions	MYAREL3	UNIQUE IMMED		ATR ONLY
? Typ	Access path	Source mbr	Key	Index options	Auto add																																		
-	PHY Physical file	MYAREP	NONE		ATR ONLY																																		
-	UPD Update index	MYAREL0	UNIQUE IMMED		ATR ONLY																																		
-	RTV Retrieval index	MYAREL1	UNIQUE IMMED		ATR ONLY																																		
Z	RTV Mares	MYAREL2	UNIQUE IMMED		ATR ONLY																																		
█	RTV Stallions	MYAREL3	UNIQUE IMMED		ATR ONLY																																		
SEL: Z=Details, G/J=Generate, E=STSEU, D=Delete, L=Locks, O=Overrides H=Hold/Release, I=Trim, V=Virtualize, U=Usage, F=Func refs., N=Narrative F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative																																							

Press Enter to display the Edit Access Path Details panel.

EDIT ACCESS PATH DETAILS		My model											
File name . . . . .	Horse	Attribute . .	REF										
Access path name . . . .	Mares	Type . . . .	RTV										
Unique or duplicate order .	U (U-Unique, F-FIFO, L-LIFO, C-FCFO, ' '-Undefined)												
Index maintenance option .	I (I-IMMED, D-DLY, R-REBLD)												
Alternate collating table .													
Allow select/omit . . . .	(S-Static, D-Dynamic, ' '-None)												
Generation mode . . . .	M (M-MDLVAL, D-DDS, S-SQL, X-UNX)												
Source member name . . .	MYAREL2												
Source member text . . .	Horse	Mares											
<table border="1"> <thead> <tr> <th>? Seq name</th> <th>Format</th> <th>GEN pfx</th> <th>Format text (Based on file)</th> <th>Associated Update access path</th> </tr> </thead> <tbody> <tr> <td>█ 1</td> <td>FAEREAS</td> <td>AE</td> <td>Horse</td> <td>Update index</td> </tr> </tbody> </table>				? Seq name	Format	GEN pfx	Format text (Based on file)	Associated Update access path	█ 1	FAEREAS	AE	Horse	Update index
? Seq name	Format	GEN pfx	Format text (Based on file)	Associated Update access path									
█ 1	FAEREAS	AE	Horse	Update index									
SEL: Z=Entries, R=Relations, S=Select/omit, A=Assoc.acps, T=Trim, V=Virtualize F3=Exit F8=Rename F20=Narrative													

This panel supplies the parameters that control the i OS options for that access path. In this tutorial, we will discuss only those options you need to define conditions; namely, Select/Omit Sets and Static and Dynamic Selection. See your IBM manuals for details about the other options.

## Select/Omit Sets

A *select/omit set* lets you select or omit records from a view based on data values in specified fields. Specifying a select/omit set for an access path is a two-level process.

You can specify none, one, or many select/omit sets for a given access path. Multiple select/omit sets are joined with a logical OR; in other words, once a record satisfies a select/omit set, other sets are not relevant.

Within each select/omit set, you define one or more conditions. Multiple conditions are joined with a logical AND; in other words, all conditions need to be true for the entire set to be true.

In this tutorial, there is only one select/omit set for each access path and only one condition attached to each.

## Static and Dynamic Selection

There are two kinds of access path selection available under i OS: static and dynamic.

Static selection is permanently built into the access path of the i OS logical file so that the logical file only contains the records that meet the selection criteria.

Dynamic selection is not stored in the access path, but is applied to each record as it is retrieved.

You will use dynamic selection in this tutorial.

## Selection Conditions for the Mares Access Path

To specify the condition that mares must be female for the Mares access path, create a select/omit set for the access path, and attach conditions to this set. Type **D** in the Allow select/omit field to select Dynamic i OS selection and type **S** in the Subfile selector to define the select/omit criteria.

```

EDIT ACCESS PATH DETAILS                               My model
File name . . . . . : Horse                               Attribute . : REF
Access path name. . . : Mares                             Type. . . . : RTV
Unique or duplicate order : U (U-Unique, F-FIFO, L-LIFO, C-CFQ, ' '-Undefined)
Index maintenance option : I (I-IMMED, D-DLV, R-REBLD)
Alternate collating table :
Allow select/omit . . . : D (S-Static, D-Dynamic, ' '-None)
Generation mode . . . : I (M-MDLVAL, D-DDS, S-SQL, X-UNIX)
Source member name . . : MYAEREL2
Source member text . . : Horse                               Mares

```

---

```

          Format      GEN  Format text      Associated
? Seq name   pfx      (Based on file)    Update access path
S  1 FAEREAR   AE      Horse              Update index

```

SEL: Z=Entries, R=Relations, S=Select/omit, A=Assoc.acps, T=Trim, V=Virtualize  
F3=Exit F8=Rename F20=Narrative

Press Enter to display the Edit Access Path Select/Omit panel.

### Select/Omit Set for Mares Access Path

To define the select/omit set, you will need to enter the following information on the Edit Access Path Select/Omit panel.

- **Z** against the line defining the select/omit set to zoom into another panel where you enter details (the conditions) for the set.
- **S** for select in the S/O column. Note that you could define a set to omit (**O**) male horses instead. The result would be the same.
- A text description for the set.

In this case, use **Female horses only** for the description.

Type **Z**, **S**, and the description of the select/omit set.

```

EDIT ACCESS PATH SELECT/OMIT      My model
File name . . . . . : Horse       Attribute . . : REF
Access path name. . . . . : Mares Type. . . . . : RTV
Format text . . . . . : Horse     Format No . . : 1
Based on. . . . . : Horse

? S/O Seq Text description
Z S _____ Female horses only
- - _____
- - _____
- - _____
- - _____
- - _____
- - _____
- - _____
- - _____
- - _____
- - _____
- - _____

```

+

SEL: Z=Conditions.  
F3=Exit F5=Rload F9=Entries



**Note:** If you type **?** in the Field column, CA 2E displays a list of the fields on the access path on which selection could be based. If you type **?** in the Condition column, CA 2E displays a list of possible condition values for the associated field.

Press Enter to define the condition. The process is now complete for the selection of Mares. Press F13 (Fast exit) to return to the Edit File Details panel.

## Selection Access Path for Stallions

You can now specify a selection access path for Stallions by repeating the process you just used to specify a selection access path for Mares. Type **Z** against the Stallions access path.

EDIT FILE DETAILS		My model	
File name . . . . .	Horse	Field reference file . .	»NONE
Attribute . . . . .	REF	Source library . . . .	MYGEN
Documentation sequence . .		Distributed . . . . .	N (V,N)
GEN format prefix . . . .	AE		
Assimilated physical . . .			
Record not found message .	Horse	NF Msgid .	USR0007
Record exists message . .	Horse	EX Msgid .	USR0008
? Typ Access path			
- PHY Physical file	MYAERE0	NONE	ATR ONLY
- UPD Update index	MYAERE0	UNIQUE IMMED	ATR ONLY
- RTV Mares	MYAERE2	UNIQUE IMMED DYNST	ATR ONLY
- RTV Retrieval index	MYAERE1	UNIQUE IMMED	ATR ONLY
<b>Z</b> RTV Stallions	MYAERE3	UNIQUE IMMED	ATR ONLY
SEL: Z=Details, G/J=Generate, E=STRSEU, D=Delete, L=Locks, O=Overrides H=Hold/Release, T=Trim, V=Virtualize, U=Usage, F=Func refs., M=Narrative F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative			

Press Enter.

## Access Path Details

To specify the condition that stallions must be male for the Stallions access path, create a select/omit set for the access path, and attach conditions to this set.

Type **D** in the Allow select/omit field to select Dynamic i OS selection and type **S** in the Subfile selector to define the select/omit criteria.

Press Enter.

Press Enter.



```

EDIT ACCESS PATH CONDITIONS      My model
File name      : Horse          Attribute : REF
Access path name : Stallions   Type      : RTV
Format text    : Horse
Based on       : Horse
Sel/Omit set   : S Male horses only

Format No. : 1

Seq  Field          Condition
----  -----
  1  Horse gender   Stallion
  2
  3
  4
  5
  6
  7
  8
  9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003

```

Press Enter to finish defining the Male horses only select/omit set. Press F13 (Fast exit) to return to the Edit File Details panel.

## Access Path Relations

The entries to be included on the record format for an access path are defined by the access path relations. Access path relations can include all the file relations, or be a subset of them. The key level relations, however, must be present on every access path.

## Modifying the Default RTV Access Path for HORSE

In this step you will modify the default Retrieval access path for the HORSE file to ensure that Dams are female and Sires are male.

First display the access path details by zooming into the default Retrieval index. Type **Z** against the default RTV access path.

<b>EDIT FILE DETAILS</b>		<b>My model</b>	
File name . . . . .	: Horse	Field reference file . .	: *NONE
Attribute . . . . .	: REF	Source library . . . . .	: MYGEN
Documentation sequence . .	:	Distributed . . . . .	: N (V,N)
GEN format prefix . . . .	: AE		
Assimilated physical . . .	:		
Record not found message .	: Horse	NF Msgid . . . . .	: USR0007
Record exists message . .	: Horse	EX Msgid . . . . .	: USR0008

? Typ Access path	Source mbr	Key	Index options	Auto add
PHY Physical file	MYAEREP	NONE		ATR ONLY
UPD Update index	MYAEREL0	UNIQUE IMMED		ATR ONLY
RTV Retrieval index	MYAEREL1	UNIQUE IMMED		ATR ONLY
RTV Mares	MYAEREL2	UNIQUE IMMED DYNSLT		ATR ONLY
RTV Stallion	MYAEREL3	UNIQUE IMMED DYNSLT		ATR ONLY

SEL: Z-Details, G/J-Generate, E-STRSEU, D-Delete, L-Locks, O-Overrides  
 H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative  
 F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative

Press Enter to display the Edit Access Path Details panel for the default retrieval access path.

## Displaying Access Path Relations

You are now ready to view the list of access path relations for the HORSE Retrieval index access path. Type **R** in the Subfile selector.

<b>EDIT ACCESS PATH DETAILS</b>		<b>My model</b>	
File name . . . . .	: Horse	Attribute . . . . .	: REF
Access path name . . . . .	: Retrieval index	Type . . . . .	: RTV
Unique or duplicate order .	: U (U-Unique, F-FIFO, L-LIFO, C-FCFO, ' '-Undefined)		
Index maintenance option .	: I (I-IMMED, D-DLV, R-REBLD)		
Alternate collating table .	:		
Allow select/omit . . . . .	: _ (S-Static, D-Dynamic, ' '-None)		
Generation mode . . . . .	: M (M-MDLVAL, D-DDS, S-SQL, X-UNX)		
Source member name . . .	: MYAEREL1		
Source member text . . .	: Horse	Retrieval index	

Format	GEN	Format text	Associated
? Seq name	pfx	(Based on file)	Update access path
<b>R 1</b> <u>MYAEREL1</u>	<b>AE</b>	<u>Horse</u>	<u>Update index</u>

SEL: Z-Entries, R-Relations, S-Select/omit, A-Assoc.acps, T-Trim, V-Virtualize  
 F3=Exit F8=Rename F20=Narrative

Press Enter to display the Edit Access Path Relations panel.

EDIT ACCESS PATH RELATIONS		My model
File name . . . . .	Horse	Attribute . . : REF
Access path name. . . . .	Retrieval index	Type. . . . : RTV
Format text . . . . .	Horse	
Based on. . . . .	Horse	Format No . . : 1
? D Verb	File/for	Access path/function
<input checked="" type="checkbox"/> * Known by	Horse code	
<input type="checkbox"/> * Has	Horse name	
<input type="checkbox"/> * Has	Horse gender	
<input type="checkbox"/> * Has	Horse value	
<input type="checkbox"/> * Has	Date of birth	
<input type="checkbox"/> * Refers to	Horse	Retrieval index
	Dam	
<input type="checkbox"/> * Refers to	Horse	Retrieval index
	Sire	
A-Ref Accpths, S-Select F4, T-Default F4, '+'/'-'-Add/Rmv relation, V-Virtual F3=Exit F7=Entries		

## Access Path Relations for the Retrieval Index

The default Retrieval index access path includes all the relations that define a file; in other words, it contains all the fields in the file. The included relations are indicated by \* in the D column. You can drop non-key relations in order to include a subset of fields in your logical view of the data. You can later reinstate a dropped relation.

Note that each of the two Refers to access path relations involves a reference to another access path. Any reference to another file by an access path relation is by means of an access path of the referenced file known as the *referenced access path*. By default this is the Retrieval index of the referenced file.

## Involution and Access Path Relations

Because the HORSE file Refers to itself in your model (involution), you need to change the default referenced access path for these relations. Otherwise the same access path will be called more than once in functions that use the file. Note that you need to change the default referenced access path only when involution takes place.

The following sections describe how to change the referenced access path for the two Refers to access path relations.

## Changing a Referenced Access Path

To change the referenced access path used by the HORSE Refers to HORSE For Dam relation, type **A** against the relation.

```

EDIT ACCESS PATH RELATIONS          My model
File name . . . . . : Horse          Attribute . : REF
Access path name. . . . : Retrieval index  Type. . . . : RTV
Format text . . . . . : Horse
Based on. . . . . : Horse          Format No . : 1
? D Verb      File/for      Access path/Function
_ * Known by   Horse code
_ * Has        Horse name
_ * Has        Horse gender
_ * Has        Horse value
_ * Has        Date of birth
@ * Refers to   Horse          Retrieval index
                        Dam
■ * Refers to   Horse          Retrieval index
                        Sire
A-Ref Accpths, S-Select F4, T-Default F4, '+'/'-'-Add/Rmv relation, U-Virtual
F3=Exit  F7=Entries

```

Press Enter to view access paths for the referenced file.

## Selecting an Access Path for Mares

Select the access path that is to be used in place of the default referenced access path. In this case, select the Mares access path by typing **X** against Mares.

```

DISPLAY FILE ACCESS PATHS          My model
File name . . . . . : Horse          Attribute . : REF

Access path
-----
? Typ Access path      <== Position display
X RTV Mares            MYAREL2
■ RTV Retrieval index  MYAREL1
_ RTV Stallions        MYAREL3

SEL: X-Select access path, H-Narrative.
F3=Exit

```

Press Enter to select the access path and return to the Edit Access Path Relations panel.

## Selecting an Access Path for Stallions

Repeat this process for the HORSE Refers to HORSE For Sire access path relation. In other words, change the referenced access path for HORSE Refers to HORSE For Sire from Retrieval index to Stallions.

Type **A** against the HORSE Refers to HORSE For Sire relation.

```

EDIT ACCESS PATH RELATIONS          My model
File name . . . . . : Horse          Attribute . : REF
Access path name. . . . : Retrieval index  Type. . . . : RTV
Format text . . . . . : Horse
Based on. . . . . : Horse          Format No . : 1
? D Verb      File/for      Access path/function
■ * Known by   Horse code
- * Has        Horse name
- * Has        Horse gender
- * Has        Horse value
- * Has        Date of birth
- * Refers to   Horse          Mares
                  Dam
A * Refers to   Horse          Retrieval index
                  Sire
A-Ref Accpths, S-Select F4, T-Default F4, '+'/'-'-Add/Rmv relation, V-Virtual
F3=Exit  F7=Entries

```

Press Enter to view the access paths available for selection from the referenced file.

Type **X** against the Stallions access path.

```

DISPLAY FILE ACCESS PATHS          My model
File name . . . . . : Horse          Attribute . : REF

Access path
■ _____ <== Position display
? Typ Access path      Source mbr
- RTV Mares            MYAEREL2
- RTV Retrieval index  MYAEREL1
X RTV Stallions        MYAEREL3

SEL: X-Select access path, M-Narrative.
F3=Exit

```

Press Enter.

Press F13 to return to the Edit File Details panel. Press F3 to return to the Edit Database Relations panel.



# Chapter 4: Designing Functions

---

This chapter introduces the following topics:

- Functions
- Device Designs
- Action Diagrams
- Function Options
- Linking Functions
- Function Parameters

## Introduction to Functions

This topic introduces you to CA 2E functions.

New terms introduced:

- Function
- External function
- Internal function
- Function type
- Edit File (EDTFIL) function type
- Select Record (SELRCD) function type

New panels introduced:

- Edit Functions
- Edit Function Details

## Objectives

You will design interactive panels for the four following functions:

- Edit the COURSE file
- Select records from the COURSE file
- Edit the HORSE file
- Select records from the HORSE file

You will also define two additional functions:

- Select Stallions
- Select Mares

## Overview of Functions

A *function* defines a list of processes that will operate on the files and fields in your database. CA 2E provides a variety of standard functions to which you can add your own unique processing. CA 2E uses the information contained in your data model to provide all of the default processing for the standard functions.

The set of standard functions consist of database functions, device functions, and user functions. Note that only the first two will be covered in this tutorial. The standard functions consist of a variety of *function types*, each of which performs a specific process. For example, the Edit File (EDTFIL) function type retrieves data from a file and lets the end user update that data using an interactive panel. The Print File (PRTFIL) function type prints a report of the data from a file. The Create Object (CRTOBJ) database function creates a record in your database.

A function can be implemented either as a separate program or as a subroutine within a program as follows:

- An *external function* is implemented as a separate high-level language (HLL) program. Each external function is attached to an access path. In general, the standard device functions are external functions; for example, the Edit File function. The only exception is the Print Object (PRTOBJ) function that is covered in a later chapter of this tutorial.
- An *internal function* is implemented as source code within that of the calling program; in other words, it is implemented as a subroutine. All the standard database functions are internal functions; for example, the Create Object function.

There need not be a one-to-one correspondence between a function and the program that implements it. You can link functions together to create larger processes that become the building blocks of your applications; namely, several internal functions may be implemented as a single program. In other words, a function is a unit of specification, not implementation.



The standard functions may have the following components. Note that not all function types have all four components.

- **Device design**—Specifies the layout of the panel or report used by the function
- **Action diagram**—Specifies the processing steps that make up the function (includes default processing and processing that you define)
- **Function options**—Specifies default features of a function
- **Function parameters**—Specifies fields to be passed into the function and returned from the function at run time

Each of these components is discussed in this chapter.

In addition to the standard functions, CA 2E provides message functions, function fields, and built-in functions, which perform low-level processing such as arithmetic operations. These will all be discussed in this and the following chapters.

## Default Functions

CA 2E automatically creates five default functions for each reference file (REF) you declare: two external functions (Edit File and Select Record), and three internal functions (Change Object, Create Object, and Delete Object). Note that for capture files (CPT), CA 2E automatically creates only the three internal functions.

## Default External Functions

The two external functions CA 2E automatically creates are:

- An *Edit File* (EDTFIL) function  
This type of function maintains multiple records on the file. An example in this tutorial will be the Edit Course function.
- A *Select Record* (SELRCO) function  
This type of function provides a display of the records on the file, and allows the selection of one record for return to a calling program. An example in this tutorial will be the Select Course function.

Both of these external functions are interactive device functions. Note that CA 2E does not automatically create these device functions for capture files (CPT).



EDIT FUNCTIONS		
File name. . . : Course		
My model		
** 1ST LEVEL **		
? Function	Function type	Access path
- Change Course	Change object	Update index
- Create Course	Create object	Update index
- Delete Course	Delete object	Update index
- Edit Course	Edit file	Retrieval index
- Select Course	Select record	Retrieval index
-		
-		
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, H-Harr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press **F3** to return to the Edit Database Relations panel.

Displaying Default Functions For HORSE

Type **Horse\*** on the selection line and press Enter to display only the relations defined for the HORSE file.

Display the functions for the HORSE file by typing **F** against any relation on the HORSE file.

EDIT DATABASE RELATIONS		
=> <u>Horse*</u>		
? Typ Object	Rel lvl:	
F FIL Horse	Known by	10 FLD Horse code
- FIL Horse	Has	20 FLD Horse name
- FIL Horse	Has	30 FLD Horse gender
- FIL Horse	Has	40 FLD Horse value
- FIL Horse	Has	50 FLD Date of birth
- FIL Horse	Refers to	60 FIL Horse
For: Dan		Sharing: *ALL
- FIL Horse	Refers to	70 FIL Horse
For: Sire		Sharing: *ALL
-		
-		
-		
-		
-		
Bottom		
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options		
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys		
Function 'Create Horse' is being created ...		

Press Enter to display the default functions created for the HORSE file.

CA 2E creates the default functions the first time you access the functions for a file. Messages to indicate this appear at the bottom of the Edit Database Relations panel. For example, "Function 'Edit Horse' is being created."

EDIT FUNCTIONS		
File name. . . : Course		
My model		
** 1ST LEVEL **		
? Function	Function type	Access path
■ Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
- Edit Horse	Edit file	Retrieval index
- Select Horse	Select record	Retrieval index
-		
-		
-		
-		
-		
-		
-		
-		
-		
Note...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, H-Marr, O-Open,		
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.		
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

## Adding Functions

You can add your own functions to the list supplied by CA 2E. To declare a function, specify a function name, a function type, and the name of the access path that defines how data from the file is presented to the function. You can view a list of possible function types by typing ? in the Function type column, and a list of the existing access paths based on the HORSE file by typing ? in the Access path column.

## Select Mares and Select Stallions Functions

In this step you will add two new Select record functions, Select Mares and Select Stallions. These functions will enable you to display a list of mares only or stallions only in response to an inquiry. These functions will use the Mares and Stallions access paths you previously defined.

Type the function names, function types, and access paths.

EDIT FUNCTIONS		
File name. . . : Horse	My model	** 1ST LEVEL **
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
- Edit Horse	Edit file	Retrieval index
- Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Narr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press Enter.

Creating New Functions

The two messages “Function ‘Select Mares’ is being created” and “Function ‘Select Stallions’ is being created” will appear briefly at the bottom of the panel. The functions will now be declared to your design model and are available for reference by other functions. At this stage no further definition is required.

**Note:** You will need to define additional aspects of the new functions later in this tutorial before they can be implemented.

Understanding Function Details

Each interactive external function will be implemented by means of:

- A High Level Language (HLL) program
- A DDS display file
- A file containing Help text

An appropriate source member is created to contain each type of generated source. CA 2E supplies default names for the source members. You can change the default source member names using the Edit Function Details panel.

The implementation language is controlled by the target HLL (higher level language). You may generate in RPG, COBOL, UNIX, or Client Server Applications depending on your version of CA 2E.

The default implementation language for new functions is controlled by the YHLLGEN model value.

Displaying Function Details

You will now display the details of the Edit Horse function. Type **Z** against the Edit Horse function.

EDIT FUNCTIONS			My model
File name. . . : Horse			** 1ST LEVEL **
? Function	Function type	Access path	
- Change Horse	Change object	Update index	
- Create Horse	Create object	Update index	
- Delete Horse	Delete object	Update index	
Z Edit Horse	Edit file	Retrieval index	
█ Select Horse	Select record	Retrieval index	
- Select Mares	Select record	Mares	
- Select Stallions	Select record	Stallions	
-			
-			
-			
-			
-			
-			
			More...
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, M-Harr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services			

Press Enter to view function details for the Edit Horse function. Note the program (RPG), display file (DDS), and the help text file (UIM).

EDIT FUNCTION DETAILS				My model
Function name . . : Edit Horse				Type : Edit file
Received by file. : Horse				Acpth: Retrieval index
Workstation . . . : NPT				
Source library. . : MYGEN				
Object	Source	Target	Text	
? Type	Name	HLL	Text	
█ PGM	MYAEEFR	RPG	Edit Horse	Edit file
-	DSP	MYAEEFRD	DDS	Edit Horse Edit file
-	HLP	MYAEEFRH	UIM	Edit Horse Edit file
SEL: E-STRSEU, O-Compiler Overrides.				
F3=Exit F7=Options F8=Change name F9=Scr/rpt layout F20=Narrative				

Device Designs

CA 2E automatically creates a default interactive panel design for each interactive device function. You can modify the default designs to meet the requirements of your application. This topic describes how to use the CA 2E Device Design Editor to modify the device design associated with your Edit Horse and Select Horse functions.

The Edit Horse function will show the use of an action bar (pull-down menu) as part of the panel design. The Select Horse function will show the use of a window.

New terms introduced:

- CA 2E device design
- Usage

New panel introduced:

- Edit Function Devices

## Overview of Device Designs

When you declare a function, CA 2E supplies a default panel design. You can modify it to meet your requirements using an interactive panel design tool.

A device design specifies the following:

- The selection of fields present on a device panel and the accompanying text for each field
- The display of both fields and text may be conditional.
- The order in which the fields are displayed on the device panel and how they are edited
- The display attributes for the fields
- The use of each field

All of the above are defaulted from the design model. You may override most of these default values.

## Displaying the Device Design for Edit Horse

The Edit Function Details panel for the Edit Horse function should be displayed on your screen. If it is not, follow the steps at the end of the preceding topic.

There is only one panel design for the Edit Horse function. You can view the default device design from the Edit Function Details panel by pressing F9.

EDIT FUNCTION DETAILS

My model

Function name . . . : Edit Horse

Received by file. . : Horse

Workstation . . . : NPT

Source library. . : MYGEN

Type : Edit file

Acpth: Retrieval index

Object	Source	Target	HLL	Text
? Type	Name			
<input checked="" type="checkbox"/> PGM	MYAEEFR	RPG	Edit Horse	Edit file
<input type="checkbox"/> DSP	MYAEEFRD	DDS	Edit Horse	Edit file
<input type="checkbox"/> HLP	MYAEEFRH	UIM	Edit Horse	Edit file

SEL: E-STRSEU, O-Compiler Overrides.

F3=Exit F7=Options F8=Change name F9=Scr/rpt layout F20=Narrative

The default panel design for the Edit Horse function displays.

FileFunctionSelectorHelp

\*PROGRAM \*PGMMOD

DD/MM/VV HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth	Dam Hors code
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			

F3=Exit F5=Reset F9=Open F10=Actions

Design exceeds device size limits for function Edit Horse

**Note:** Do not be concerned about the message at the bottom of the panel.

Default Device Design Formats

A panel design consists of header/footer formats and a variable format that depends on the function type. The panel design for the Edit Horse function, which is an Edit File function type, consists of the following formats.

- The uppermost format contains the header lines that give status information about the user, date, time, and mode (New/Open) of the panel.
- At the bottom of the panel, the footer lines give function key explanations. Any execution time messages are displayed on line 24.



- The middle section of the panel contains the formats that vary depending on the function type. In this case we are using an Edit File function type. It contains two formats,
  - A subfile control format.
  - A subfile record format. The subfile record is the format for a single record that is retrieved from the fields of the specified access path according to the function type.

## Design Standards and Subfile Selector Options

When the function type includes a subfile, descriptions for the Subfile selector options are included as part of the default device design. These can be

- Part of the footer along with standard function key descriptions
  - Placed above the subfile record as part of the subfile control format
- This is the CUA ENTRY design standard.
- Listed as actions in the Selector Choice menu that is part of the action bar at the top of the panel
- This is the CUA TEXT design standard.

The default design standard for CA 2E design models on the IBM i is CUA ENTRY. However, when you created your design model at the beginning of this tutorial using the Create Model Library (YCRTMDLLIB) command, you were told to specify \*CUATEXT as the design standard for your model.

When the design standard is set to CUA TEXT, CA 2E automatically creates all Edit File functions with an action bar on the panel design. Note the action bar at the top of the panel design for the Edit Horse function. An action bar provides a set of choices that give the end users access to various actions available on the panel. CA 2E automatically provides a default set of choices depending on the function type. Later in this tutorial you will define a new choice and action on the Edit Horse function's action bar.

## Subfile Control Default

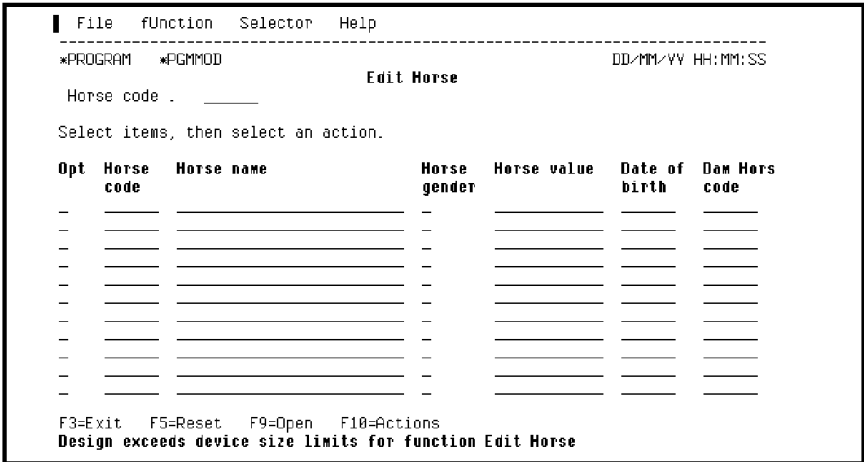
The key field of the HORSE file (Horse code) is placed on the subfile control format as well as on the subfile record. This field is a positioning field that causes the subfile record to begin at a particular record when a value is entered into this field.

Subfile Record Default

To determine the default panel design for an Edit File function, CA 2E places all the fields from the access path side-by-side on a subfile record, with the field names as column headings. If the fields total more than 80 characters, the design continues into a work area on the right. It is up to you, the designer, to adjust the design until the fields are all displayed on the panel. In this tutorial, the fields exceed the panel display. CA 2E alerts you to this fact and displays the message "Design exceeds device size limits for function Edit Horse" at the bottom of the panel.

Displaying the Rest of the Device Design

By default, only 80 characters of the panel design may be displayed at one time. You can move the panel window to display more of the panel design by pressing F4, which moves the window 40 columns to the right. Pressing F4 again moves the subfile record another 40 columns to the right. Pressing F1 moves the panel design 40 columns to the left.



Press F4 to move the device design 40 columns to the right.

Note the virtual fields, Dam name and Dam Date of birth that you added to the default Retrieval access path for the HORSE file. If you had not explicitly added these fields to the access path, they would not appear on this display.

Note also that the virtual fields are for output only and are represented by 0s or 6s.

-----  
DD/MM/YY HH:MM:SS

orse

|

Horse gender	Horse value	Date of birth	Dam Horse code	Dam name	Dam Date of birth
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66
-	_____	_____	_____	00000000000000000000000000000000	66/66/66

ns

Design exceeds device size limits for function Edit Horse

Press F4 again to move the device design 40 more columns to the right.

|

Dam name	Dam Date of birth	Sire Horse code	Sire name	\$
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6
00000000000000000000000000000000	66/66/66	_____	00000000000000000000000000000000	6

Design exceeds device size limits for function Edit Horse

Press F15 to return to the left margin of the device design; in other words, back to the first field in the subfile record.

Editing the Default Device Design

The Device Design Editor has a number of facilities to enable you to quickly design a more compact, understandable panel. Many of these facilities involve the use of function keys and are generally dependent on the position of the cursor when the function key is pressed. The device design is updated as you make each modification.

The following table shows some of the function keys available for editing your device design. Most of these will be illustrated in this and other topics in the tutorial. To learn about additional features of the Device Design Editor, refer to the Help text or the CA 2E manual *Building Applications*.

Function Key	Action
F1	Moves the device design 40 columns to the left
F2	Animates the device design using CA 2E Toolkit prototyping facilities
F4	Moves the device design 40 columns to the right
F5	Displays the Edit Screen Format Details for the format on which the cursor is positioned
F7	Displays the Edit Screen Format Relations panel when the cursor is positioned on the first subfile record
F8	Selects a field to be moved (cut) and moves the selected field after the field on which the cursor is positioned (paste)
F9	Moves the field on which the cursor is located and all fields to its right to the next line
F10	Adds a space before the field on which the cursor is positioned; in other words, moves text one column to the right
F15	Moves device design to the left margin
F16	Moves device design to the right margin
F17	Displays a list of device formats
F18	Displays the Edit Screen Field Attributes panel for the field on which the cursor is positioned
F19	Adds a function field to the device design
F20	Edits a function field on the device design
F22	Removes a space before the field on which the cursor is positioned; in other words, moves text one column to the left

## Reducing the Width of the Device Design Layout

You can reduce the width of the device design by folding the subfile records that make up the device design. This way the fields in each record are arranged over several lines rather than in one long line. You will use the F9 key to do this.

The following steps in this tutorial show how to split the information about each horse into three lines that give:

- Information about the horse itself

- Information about the horse’s Dam
- Information about the horse’s Sire

The change in position of the fields on the device design is determined by the position of the cursor when you press the function key. To begin the second line with *Dam Horse code*, you will simply place the cursor on this field and press F9. The subfile record format will split into two lines. The device design is updated to reflect the change.

Position cursor.

FilefUnctionSelectorHelp

\*PROGRAM

\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth	Dam Hors code
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			
-			-			

F3=Exit

F5=Reset

F9=Open

F10=Actions

Design exceeds device size limits for function Edit Horse

Press F9. The subfile record format now displays only two HORSE records.

Folding the Device Design Layout Again

Next, repeat the process to move the fields relevant to the Sire to a separate line from those for the Dam. Place the cursor on the Sire Horse code.

FilefUnctionSelectorHelp

\*PROGRAM

\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
Dam Horse code	Dam name	Dam Date of birth	Sire Horse code	Sire name	
	00000000000000000000000000000000	66/66/66		00000000000000000000000000000000	
Dam Horse code	Dam name	Dam Date of birth	Sire Horse code	Sire name	
	00000000000000000000000000000000	66/66/66		00000000000000000000000000000000	

F3=Exit

F5=Reset

F9=Open

F10=Actions

Design exceeds device size limits for function Edit Horse

Press F9. The subfile record format now displays only one HORSE record.

Panel Format Details

You can view a list of all the fields available on any particular format using the Edit Screen Format Details panel. To access this panel for the Subfile record format, place the cursor on any field of the first subfile record.

File   fUncion   Selector   Help

-----

\*PROGRAM   \*PGMMOD   DD/MM/YY HH:MM:SS

**Edit Horse**

Horse code .   \_\_\_\_\_

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
<b>Dam Horse code</b>	<b>Dam name</b>		<b>Dam Date of birth</b>		
█	00000000000000000000000000000000		66/66/66		
<b>Sire Horse code</b>	<b>Sire name</b>		<b>Sire Date of birth</b>		
_____	00000000000000000000000000000000		66/66/66		

F3=Exit   F5=Reset   F9=Open   F10=Actions

Press F5.

**EDIT SCREEN FORMAT DETAILS**   My model

Format . . . . . : Subfile record.   Type: RCD

Blank lines before fmt . . . . . : 1   or Fixed start line no . . . . . :

Blank lines after column headings:      Blank line between records . . . . . :

Subfile page . . . . . :

? Field	Func	Typ	Usg	Ovr	Length	GEN name	Etp	Rqd	LL
- *SFLSEL	ACT	STS	I	I	1	*SFLSEL	U		C
- Horse code	DTA	CDE	I	I	6	ADCD	K	Y	C
- Horse name	DTA	TXT	I	I	25	ADTX	A		C
- Horse gender	DTA	STS	I	I	1	ADST	A		C
- Horse value	DTA	VAL	I	I	11.2	ABVA	A		C
- Date of birth	DTA	DT#	I	I	6.0	ACDZ	A		C
- Dam Horse code	DTA	REF	I	I	6	AFCD	A		C
- Dam name	DTA	REF	O	O	25	AFTX	V		C
- Dam Date of birth	DTA	REF	O	O	6.0	ADDZ	V		C
- Sire Horse code	DTA	REF	I	I	6	AGCD	A		C +

SEL: Z=Details, A,B,C,D=Text position, I,O,H,'-'=Field usage.  
F3=Exit   F7=Fmt rel   F10=Sequence   F19=Add function field   F23=Add constant

Edit Screen Format Details

The Edit Screen Format Details panel shows all the fields in the Subfile record format. The details shown for each field on the format include the name and type of the field. This panel also allows you to control the positioning of the format with respect to other formats on the device design.

You can also use this panel to specify a field’s *usage*; in other words, whether the field is input capable (I), output only (O), or hidden (H). This applies to any field other than the Subfile selector field (\*SFLSEL). For example, you could hide a field by typing **H** against the field or you could change an input capable field to output only by typing **O** against the field.

**Note:** Some restrictions exist when changing an output only field to an input capable field.

Editing the Panel Format

In this step you will use the Edit Screen Format Details panel to make the following two changes to improve the appearance of the Edit Horse panel.

- 1. Separate the individual subfile records with a blank line by typing **Y** (yes) against the Blank line between records field.
- 2. Place field labels before the fields on the second and third lines of the record, rather than above them, to clarify the information. CA 2E has four options for positioning the label associated with a field.
  - Above (A) - on the line above a field
  - Before (B) - on the same line as the field
  - Column heading (C) - above a column of instances of the field; this value applies only to a subfile record format
  - Drop (D) - omit the label

To reposition the labels on the second and third lines, you will type **B** against each of the Dam and Sire fields. Note that later in this tutorial you will use the Edit Screen Entry Details panel to overwrite the wording of the field labels.

Specify **Y** for the Blank line between records field and type **B** against each of the Dam and Sire fields.

EDIT SCREEN FORMAT DETAILS

My model

Format : Subfile record. Type: RCD

Blank lines before fmt : 1 or Fixed start line no :

Blank lines after column headings: Blank line between records : Y

Subfile page :

? Field	Func	Typ	Usq	Ovr	Length	GEN name	Etp	Rqd	LL
*SFLSEL	ACT	STS	I	I	1	*SFLSEL	U		C
Horse code	DTA	CDE	I	I	6	ADCD	K	Y	C
Horse name	DTA	TXT	I	I	25	ADTX	A		C
Horse gender	DTA	STS	I	I	1	ADST	A		C
Horse value	DTA	VAL	I	I	11.2	ABVA	A		C
Date of birth	DTA	DT#	I	I	6.0	ACDZ	A		C
B Dam Horse code	DTA	REF	I	I	6	AFCO	A		C
B Dam name	DTA	REF	O	O	25	AFTX	V		C
B Dam Date of birth	DTA	REF	O	O	6.0	ADDZ	V		C
B Sire Horse code	DTA	REF	I	I	6	AGCO	A		C +

SEL: Z=Details, A,B,C,D=Text position, I,O,H,'-'=Field usage.  
F3=Exit F7=Fmt rel F10=Sequence F19=Add function field F23=Add constant

Press Roll Up to display more fields. Type **B** as shown to position the labels for the Sire name and Sire Date of birth before the field on the same line.

```

EDIT SCREEN FORMAT DETAILS                      My model
Format . . . . . : Subfile record.              Type: RCD

Blank lines before fmt . . . . . : 1 or Fixed start line no . . . . . :
Blank lines after column headings:  Blank line between records . . . : Y
Subfile page . . . . . :

? Field                      Func Typ Usq Ovr Length GEN name Etp Rqd LL
B Sire name                  DTA REF 0 0 25 AGTX V C
B Sire Date of birth        DTA REF 0 0 6.0 AEDZ V C

SEL: Z-Details, A,B,C,D-Text position, I,O,H,'-'-Field usage.
F3=Exit F7=Fmt rel F18=Sequence F19=Add function field F23=Add constant

```

Press Enter to confirm the changes. Note that the value in the LL (Label Location) column for each of the Dam and Sire fields has changed from **C** to **B**. Press Roll Down to view the preceding panel.

```

EDIT SCREEN FORMAT DETAILS                      My model
Format . . . . . : Subfile record.              Type: RCD

Blank lines before fmt . . . . . : 1 or Fixed start line no . . . . . :
Blank lines after column headings:  Blank line between records . . . : Y
Subfile page . . . . . :

? Field                      Func Typ Usq Ovr Length GEN name Etp Rqd LL
- *SFLSEL                    ACT STS I I 1 *SFLSEL U C
- Horse code                 DTA CDE I I 6 ADCD K Y C
- Horse name                 DTA TXT I I 25 ADTX A C
- Horse gender               DTA STS I I 1 ADST A C
- Horse value                DTA VAL I I 11.2 ABVA A C
- Date of birth              DTA DT# I I 6.0 ACDZ A C
- Dam Horse code            DTA REF I I 6 AFCD A B
- Dam name                  DTA REF 0 0 25 AFTX V B
- Dam Date of birth         DTA REF 0 0 6.0 ADDZ V B
- Sire Horse code           DTA REF I I 6 AGCD A B +

SEL: Z-Details, A,B,C,D-Text position, I,O,H,'-'-Field usage.
F3=Exit F7=Fmt rel F18=Sequence F19=Add function field F23=Add constant

```

Press F3 to return to the Device Design Editor.

## Effect of Screen Format Changes on Device Design

The updates you have made are displayed immediately when you press F3 on the Edit Screen Format Details panel. The new panel layout is clearer, with a blank line between each subfile record and the field labels shown before the fields. However, because the field labels are now on the same line as the fields, the second and third lines of the subfile record are too long to fit on the panel.



Shortening Field Labels

You can reduce the length of the lines on the panels by shortening the field labels. This is done by editing the individual field entry details using the Edit Screen Entry Details panel.

To display the field entry details for any field, place the cursor on that field or its label in the first subfile record and press the Enter key. To display the field entry details for the Dam Horse code field, place the cursor on the field.

FilefFunctionSelectorHelp

\*PROGRAM\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
	Dam Horse code	Dam name			Dam Date of birth
	Sire Horse code	Sire name			Sire Date of birth
	Dam Horse code	Dam name			Dam Date of birth
	Sire Horse code	Sire name			Sire Date of birth
	Dam Horse code	Dam name			Dam Date of birth
	Sire Horse code	Sire name			Sire Date of birth

F3=Exit F5=Reset F9=Open F10=Actions

Design exceeds device size limits for function Edit Horse

Press Enter to access the Edit Screen Entry Details panel for Dam Horse code.

EDIT SCREEN ENTRY DETAILS

My model

Field name : Dam Horse code

Display length : 6

GEN name : AFCD

Label location : 3 (Above,Before,Column,blank)

Label spacing :

Lines before : 1

Spaces before :

Screen text : (M, L, F)

Column Headings : Dam Horse

code

Left hand side text : Dam Horse code

Right hand side text : Code

Display RHS text :

RHS spaces : 1

Fill LHS text : Y

I/O Usage : I

Check condition : \*NONE

Allow blank :

Check numeric :

Field exit option : Y

F3=Exit, no update F7=Relations F18=Screen attributes

The Edit Screen Entry Details panel gives details of the label location, spacing, text, and position. Note the different versions of the text to be used for the different label positions (Above, Before).

## Shortening Field Label for Dam Horse Code

You can shorten the label before the field by changing the Left hand side text (*LHS* text). In this case, change the Left hand side text from Dam Horse code to Dam.

```

EDIT SCREEN ENTRY DETAILS                               My model
Field name . . . . . : Dam Horse code                   Display length . . . :    6
GEN name . . . . . : AFCD
Label location . . . : 3 (Above,Before,Column,blank)  Label spacing. : __
Lines before . . . . : 1
Spaces before . . . . :                               Screen text. . . . : E (M, L, F)
Column Headings. . . : Dam Horse
                        code
Left hand side text. : Dam
Right hand side text : Code
Display RHS text . . :   RHS spaces . . . . . : 1  Fill LHS text. . . . : Y
I/O Usage. . . . . : 1

Check condition . . : *NONE
Allow blank. . . . . :   Check numeric. . . : _  Field exit option. . . : Y

F3=Exit, no update  F7=Relations  F18=Screen attributes

```

Press Enter.

## Shortening Field Label for Sire Horse Code

The second line of the subfile record format now fits on the panel. The next step is to shorten the label for the Sire Horse code field using the same process and position the cursor.

```

File  fUnction  Selector  Help
-----
*PROGRAM  *PGMMOD                               DD/MM/YY HH:MM:SS
                                Edit Horse
Horse code .  _____
Select items, then select an action.

Opt  Horse  Horse name                Horse  Horse value  Date of
code  code  name                      gender  value        birth
-----
Dam  _____  Dam name 00000000000000000000000000000000  Dam Date of birth 66/66/66
Sire Horse code  _____  Sire name 00000000000000000000000000000000  Sire Date of birth

Dam  _____  Dam name 00000000000000000000000000000000  Dam Date of birth 66/66/66
Sire Horse code  _____  Sire name 00000000000000000000000000000000  Sire Date of birth

Dam  _____  Dam name 00000000000000000000000000000000  Dam Date of birth 66/66/66
Sire Horse code  _____  Sire name 00000000000000000000000000000000  Sire Date of birth

F3=Exit  F5=Reset  F9=Open  F10=Actions
Design exceeds device size limits for function Edit Horse

```

Press Enter.

Change the Left hand side text to Sire.

EDIT SCREEN ENTRY DETAILS

My model

Field name . . . . .

Sire Horse code

Display length . . .

6

GEN name . . . . .

AGCD

Label location . . . .

B (Above,Before,Column,blank)

Label spacing. : \_

Lines before . . . . .

1

Spaces before. . . . .

Screen text. . . . .

E (M, L, F)

Column Headings. . . .

Sire Horse

code

Left hand side text. :

Sire

Right hand side text :

Code

Display RHS text . . .

RHS spaces . . . . .

1

Fill LHS text. . . . .

Y

I/O Usage. . . . .

I

Check condition . . .

\*NONE

Allow blank. . . . .

Check numeric. . . \_

Field exit option. . .

Y

F3=Exit, no update

F7=Relations

F18=Screen attributes

Press Enter to return to the device design.

Removing Field Labels

You can compress the panel layout further by removing the labels Dam name and Sire name. To do so, first place the cursor on the Dam name field.

File fUnction Selector Help

\*PROGRAM \*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
Dam		Dam name 00000000000000000000000000000000		Dam Date of birth	66/66/66
Sire		Sire name 00000000000000000000000000000000		Sire Date of birth	66/66/66
Dam		Dam name 00000000000000000000000000000000		Dam Date of birth	66/66/66
Sire		Sire name 00000000000000000000000000000000		Sire Date of birth	66/66/66
Dam		Dam name 00000000000000000000000000000000		Dam Date of birth	66/66/66
Sire		Sire name 00000000000000000000000000000000		Sire Date of birth	66/66/66

F3=Exit

F5=Reset

F9=Open

F10=Actions

Press Enter.

CA 2E displays the Edit Screen Entry Details panel for the Dam name field.

Removing Field Label for Dam Name

Notice the value of B (Before) in the Label location field; this caused the LHS text to appear as a label before the Dam name field. To remove the label, change the Label location field from B to blank.

EDIT SCREEN ENTRY DETAILS		My model	
Field name . . . . .	Dam name	Display length . . .	25
GEN name . . . . .	AFTX	Override length. . .	
Label location . . . .	_ (Above,Before,Column,blank)		Label spacing. : <input type="checkbox"/>
Lines before . . . . .			
Spaces before. . . . .	2	Screen text. . . . .	E (M, L, F)
Column Headings. . . .	Dam name		
Left hand side text. :	Dam name		
Right hand side text :	Text		
Display RHS text . . .		RHS spaces . . . . .	1
I/O Usage. . . . .	0	Fill LHS text. . . . .	Y
Check condition . . .	*NONE	Field exit option. . .	_
Allow blank. . . . .			
F3=Exit, no update F7=Relations F18=Screen attributes			

Press Enter.

## Removing Field Label for Sire Name

Remove the field label for the Sire name field in the same way. Position the cursor on the Sire name field.

File		fUnction		Selector		Help	
*PROGRAM		*PGMMOD		Edit Horse		DD/MM/YY HH:MM:SS	
Horse code . . . . .							
Select items, then select an action.							
Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth		
Dam		00000000000000000000000000000000	Dam	Date of birth	66/66/66		
Sire		Sire name 00000000000000000000000000000000	Sire	Date of birth	66/66/66		
Dam		00000000000000000000000000000000	Dam	Date of birth	66/66/66		
Sire		Sire name 00000000000000000000000000000000	Sire	Date of birth	66/66/66		
Dam		00000000000000000000000000000000	Dam	Date of birth	66/66/66		
Sire		Sire name 00000000000000000000000000000000	Sire	Date of birth	66/66/66		
F3=Exit F5=Reset F9=Open F10=Actions							

Press Enter.

Notice the value of B (Before) in the Label location field; this caused the LHS text to appear as a label before the Sire name field. To remove the label, change the Label location field from B to blank.

EDIT SCREEN ENTRY DETAILS

My model

Field name : Sire name

Display length : 25

GEN name : AGTX

Override length:

Label location : (Above,Before,Column,blank)

Label spacing:

Lines before :

Spaces before : 2

Screen text. : E (M, L, F)

Column Headings. : Sire name

Left hand side text. : Sire name

Right hand side text : Text

Display RHS text. : RHS spaces : 1

Fill LHS text. : Y

I/O Usage. : 0

Check condition : \*NONE

Allow blank. : Field exit option. :

F3=Exit, no update F7=Relations F18=Screen attributes

Press Enter.

After removing field labels for the Dam name and Sire name fields, the resulting panel will look like the following screen.

File fUnction Selector Help

\*PROGRAM \*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
Dam		00000000000000000000000000000000	Dam Date of birth	66/66/66	
Sire		00000000000000000000000000000000	Sire Date of birth	66/66/66	
Dam		00000000000000000000000000000000	Dam Date of birth	66/66/66	
Sire		00000000000000000000000000000000	Sire Date of birth	66/66/66	
Dam		00000000000000000000000000000000	Dam Date of birth	66/66/66	
Sire		00000000000000000000000000000000	Sire Date of birth	66/66/66	

F3=Exit F5=Reset F9=Open F10=Actions

Centering a Field with Respect to its Label

The Label spacing field on the Edit Screen Entry Details panel lets you position a field with respect to its label. This applies to fields on the same line as their label as well as to fields positioned below their label. One use of the Label spacing field is to center a field whose length is shorter than the label length; for example, the Horse gender field.

In this step you will center the Horse gender field below its label. To do this, access the Edit Screen Entry Details panel for the Horse gender field. Position the cursor.

```

File  fUnction  Selector  Help
-----
*PROGRAM  *PGMMOD                      DD/MM/YY HH:MM:SS

                                Edit Horse

Horse code .  _____

Select items, then select an action.

Opt  Horse code  Horse name  Horse gender  Horse value  Date of birth
-----
Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66

Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66

Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66

F3=Exit  F5=Reset  F9=Open  F10=Actions

```

Press Enter.

Type **2** in the Label spacing field to move the field two positions to the right and press Field Exit.

```

EDIT SCREEN ENTRY DETAILS                      My model

Field name . . . . . : Horse gender              Display length . . : 1
GEN name . . . . . : ADST

Label location . . . : C (Above,Before,Column,blank)  Label spacing. : 2
Lines before . . . : 1
Spaces before . . . : 2                      Screen text. . . : E (M, L, F)
Column Headings. . . : Horse
                       gender

Left hand side text. : Horse gender
Right hand side text : value
Display RHS text . . : RHS spaces . . . : 1  Fill LHS text. . . : Y
I/O Usage. . . . . : I

Prompt Function      Condition Value Displayer(*CVD, #DDL)
Check condition . . : *All values
Allow blank. . . . . :                      Field exit option. . : _

F3=Exit, no update  F7=Relations  F18=Screen attributes

```

Press Enter to return to the Device Design Editor.

Note that the Horse gender field is now centered below its label.

## Moving Fields to Right and Left

Field positioning within a device design is relative. For example, each field is positioned with respect to the previous field in the format. You can adjust the position of a field by inserting or removing spaces before it using either function keys or the Edit Screen Entry Details panel.

Using Function Keys

For example, F10 inserts one space before the field label of the field where the cursor is positioned, causing the field and all fields to the right of it to be moved to the right. F22 removes one space before the field and moves the entire line to the left.

Position the cursor on the Dam field.

FilefUnctionSelectorHelp

\*PROGRAM\*PGHMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
Dam		00000000000000000000000000000000	Dam	Date of birth	66/66/66
Sire		00000000000000000000000000000000	Sire	Date of birth	66/66/66
Dam		00000000000000000000000000000000	Dam	Date of birth	66/66/66
Sire		00000000000000000000000000000000	Sire	Date of birth	66/66/66
Dam		00000000000000000000000000000000	Dam	Date of birth	66/66/66
Sire		00000000000000000000000000000000	Sire	Date of birth	66/66/66

F3=ExitF5=ResetF9=OpenF10=Actions

Press F10 five times to indent the line beginning with Dam by five spaces.

You have just moved the Dam field to the right so that the second row begins below the Horse code field.

Using the Edit Screen Entry Details Panel

You can also move fields by changing the value in the Spaces before field on the Edit Screen Entry Details panel. Position the cursor on the Sire field.

```

File  fFunction  Selector  Help
-----
*PROGRAM  *PGMMOD                                DD/MM/YY HH:MM:SS

                                Edit Horse

Horse code .  _____

Select items, then select an action.

Opt  Horse  Horse name                Horse  Horse value  Date of
code                                gender
-    Dam    _____              Dam Date of birth 66/66/66
Sire  █     000000000000000000000000  Sire Date of birth 66/66/66
-    Dam    _____              Dam Date of birth 66/66/66
Sire  _____ 000000000000000000000000  Sire Date of birth 66/66/66
-    Dam    _____              Dam Date of birth 66/66/66
Sire  _____ 000000000000000000000000  Sire Date of birth 66/66/66

F3=Exit  F5=Reset  F9=Open  F10=Actions

```

Press Enter to display the Edit Screen Entry Details panel. Change the Spaces before field to **6** as shown and press Field Exit.

```

EDIT SCREEN ENTRY DETAILS                                My model
Field name . . . . . : Sire Horse code                Display length . . : 6
GEN name . . . . . : AGCD
Label location . . . : B (Above,Before,Column,blank)  Label spacing. : _
Lines before . . . . : 1
Spaces before . . . . : 6                               Screen text. . . . : █ (M, L, F)
Column Headings. . . : Sire Horse
                        code
Left hand side text. : Sire
Right hand side text : Code
Display RHS text . . : RHS spaces . . . . : 1 Fill LHS text. . . : Y
I/O Usage. . . . . : I

Check condition . . : *NONE
Allow blank. . . . . : Check numeric. . . : _ Field exit option. . : Y

F3=Exit, no update  F7=Relations  F18=Screen attributes

```

Press Enter.

## Adjusting the Label Spacing of the Dam Field

The above steps will line up the field labels for Dam and Sire. The entry fields can now be aligned by placing the cursor on the Dam field.



FilefFunctionSelectorHelp

\*PROGRAM\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
-	Dam				
	Sire				
-	Dam				
	Sire				
-	Dam				
	Sire				

F3=ExitF5=ResetF9=OpenF10=Actions

Press Enter.

Change the Label spacing to 1 as shown and press Field Exit.

EDIT SCREEN ENTRY DETAILS

My model

Field name . . . . .

Dam Horse code

Display length . . .

6

GEN name . . . . .

AFCD

Label location . . . .

B (Above,Before,Column,blank)

Label spacing. :

1

Lines before . . . . .

1

Spaces before. . . . .

6

Screen text. . . . .

( M, L, F )

Column Headings. . . .

Dam Horse code

Left hand side text. :

Dam

Right hand side text :

Code

Display RHS text . . .

RHS spaces . . . . .

1

Fill LHS text. . . . .

Y

I/O Usage. . . . .

I

Check condition . . .

\*NONE

Allow blank. . . . .

Check numeric. . .

Field exit option. . .

Y

F3=Exit, no update

F7=Relations

F18=Screen attributes

Press Enter.

Modified Panel

Now both the labels and the entry fields are aligned.

```

File  fUnction  Selector  Help
-----
*PROGRAM  *PGMMOD                                DD/MM/YY HH:MM:SS

                                Edit Horse
Horse code .  _____

Select items, then select an action.

Opt  Horse  Horse name                Horse  Horse value  Date of
code  code                gender                birth
-----
-   Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
    Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66
-   Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
    Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66
-   Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
    Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66

F3=Exit  F5=Reset  F9=Open  F10=Actions

```

## Optional Exercise

Repeat this process to align the entry fields for Dam Date of birth and Sire Date of birth. To do so, place the cursor on Dam Date of birth, press Enter, and change the Label spacing field to 1. Press Enter to return to the device design. The result should look like this:

```

File  fUnction  Selector  Help
-----
*PROGRAM  *PGMMOD                                DD/MM/YY HH:MM:SS

                                Edit Horse
Horse code .  _____

Select items, then select an action.

Opt  Horse  Horse name                Horse  Horse value  Date of
code  code                gender                birth
-----
-   Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
    Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66
-   Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
    Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66
-   Dam  _____  000000000000000000000000  Dam Date of birth 66/66/66
    Sire  _____  000000000000000000000000  Sire Date of birth 66/66/66

F3=Exit  F5=Reset  F9=Open  F10=Actions

```

Panel Format Relations

When the Edit Horse function is implemented, the program automatically includes code to check the validity of each input capable field on the subfile record. By default, every relation must be satisfied. This will cause the resulting application to check the HORSE file when the end user specifies a Dam Code. The Dam code must be that of a valid female horse. This presents a potential problem. When the end user adds the first horse to the application, it will be rejected. This is because there are no horses existing on the file at that time that are available as parents (Dam or Sire).

To overcome this problem, you can override the default check values for the panel format relations and make the validation of particular relations optional. When an end user enters a value, it will be validated. However, the end user will not be required to enter a value. This is done using the Edit Screen Format Relations panel.

Place the cursor on any field of the first subfile record.

FileFunctionSelectorHelp

\*PROGRAM\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66

F3=ExitF5=ResetF9=OpenF10=Actions

Press F7 to display the Edit Screen Format Relations panel.

Editing Panel Format Relations

Change the validation of a relation to optional by typing **O** against the relevant relation. In this tutorial, the check condition for the two Refers to relations and the check condition for HORSE Has Horse value should be optional.

Type **O** against the first Refers to relation and the Has Horse value relation.

EDIT SCREEN FORMAT RELATIONS				My model
File name . . . . .	Horse	Attribute . . .	REF	
Access path name. . . . .	Retrieval index	Type. . . . .	RTV	
Format text . . . . .	Horse			
Based on. . . . .	Horse	Format No . . .	1	
? Verb	File/for	Access path/Function	Check	
Known by	Horse code		REQUIRED	
■ Has	Horse name		REQUIRED	
_ Has	Horse gender		REQUIRED	
0 Has	Horse value		REQUIRED	
_ Has	Date of birth		REQUIRED	
0 Refers to	Horse	Mares	REQUIRED	
	Dam			+
R-Required, 0-Optional, N-No error, U-User, S-Select F4, T-Default F4				
F3=Exit				

Press Roll Up to display the second HORSE Refers to HORSE relation and type **0** in the Subfile selector.

EDIT SCREEN FORMAT RELATIONS				My model
File name . . . . .	Horse	Attribute . . .	REF	
Access path name. . . . .	Retrieval index	Type. . . . .	RTV	
Format text . . . . .	Horse			
Based on. . . . .	Horse	Format No . . .	1	
? Verb	File/for	Access path/Function	Check	
0 Refers to	Horse	Stallions	REQUIRED	
	Sire			
R-Required, 0-Optional, N-No error, U-User, S-Select F4, T-Default F4				
F3=Exit				

Press Enter to confirm. Note that the value in the Check column for the selected relations changed from REQUIRED to OPTIONAL.

Press F3 to return to the device design.

## Completed Device Design

The Edit Horse device design is now complete and should look like this:

FilefunctionSelectorHelp

\*PROGRAM

\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66

F3=ExitF5=ResetF9=OpenF10=Actions

Press F3 to save and exit the device design. The Edit Function Devices panel displays.

Exiting the Device Design

From the Edit Function Devices panel you can change the panel title, display the action diagram for the function, display all open functions (CA 2E allows you to have multiple functions open), return to the device design, or exit and save the device design.

EDIT FUNCTION DEVICES

My model

Function name. : Edit Horse

Type : Edit file

Received by file : Horse

Acpth: Retrieval index

? Title

Screen title..... Edit Horse

SEL: 2=Scr/rpt design, N=Narrative, A=Animate

F3=ExitF5=Action diagramF15=Open Functions

Press F3 to exit and display the Exit Function Definition panel.

## Exit Function Definition

The Exit Function Definition panel lets you optionally save any changes to the device design. You can also change the name of the function, access path, or file. From this panel you can also print the function details, return to editing the device design, or submit the function for generation.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<u>Y</u>	Y=Yes, N=No
Function name . . . . .	<u>Edit Horse</u>	Name
Access path name. . . . .	<u>Retrieval index</u>	Name
File name . . . . .	<u>Horse</u>	Name
Function type . . . . .	Edit file	
Print function. . . . .	<u>N</u>	Y=Yes, N=No
Return to editing . . . . .	<u>N</u>	Y=Yes, N=No
Submit generation . . . . .	<u>N</u>	Y=Yes, N=No
F5=Refresh   F12=Cancel   F15=Open Functions		

Accept the defaults and press Enter to save the device design.

## Function Confirmation

When the save process is complete, CA 2E automatically returns to the Edit Function Details panel; the message "Function 'Edit Horse' has been saved," appears at the bottom of the panel. Press F3 to return to the Edit Functions panel.

## Window Device Design

In this step you will edit a function that has a window for its default display file. When the design standard is set to CUA TEXT, CA 2E automatically creates all Select Record functions with a window defined for the display file. Recall that you specified \*CUATEXT on the Create Model Library (YCRTMDLLIB) command when you created your model at the beginning of this tutorial.

The following steps in this tutorial illustrate how to use the Device Design Editor to modify the default window designs associated with the Select Record (SELRCR) function.

Modifying the default window design involves two basic steps. Both of these steps can be accomplished with the Device Design Editor.

- 1. Arrange the fields on the panel, hiding any that are unnecessary.
- 2. Adjust the window dimensions to surround the fields present on the device design.

Invoking the Device Design

You can access the default device design for a function directly from the Edit Functions panel. Compare this to the way you accessed the device design for the Edit Horse function in the tutorial topic you just completed. In that case you typed **Z** to display the Edit Function Details panel and then pressed F9 to access the device design.

Select the Select Horse function by typing an **S** in the ? column (Subfile selector).

EDIT FUNCTIONS		
File name. . . : Horse		
My model		
** 1ST LEVEL **		
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
- Edit Horse	Edit file	Retrieval index
S Select Horse	Select record	Retrieval index
█ Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
-		
More...		
SEL: Z=Details, P=Parms, F=Action diagram, S=Device design, H=Harr, O=Open, T=Structure, A=Access path, U=Usage, G/J=Generate, D=Delete, C=Copy, L=Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press Enter to display the default device design.

Default Device Design for Select Horse

A default window device design displays. The subfile control format shows only the key fields and the subfile record format shows all fields on the HORSE file. You can use any of the procedures shown in the previous section to modify the device design. In addition, in this step, you will hide all of the fields in the HORSE file other than Horse code, Horse name, Horse gender, and Date of birth in the subfile record format.

Position cursor on any field in the first subfile record.

```
.....Select Horse.....  
:  
: Horse  
: code  
: _____  
:  
: 1=Select  
:  
:  
: Opt   Horse name      Horse value    Date of     am  
:       code            gender                birth      od  
:  
: █     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: -     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: -     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: -     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: -     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: -     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: -     000000          000000000000000000000000  0        6566666666.66CR  66/66/66 : 00  
: F3=Exit   F4=Prompt  
:  
:  
.....
```

Design exceeds device size limits for function Select Horse

Press F5 to display a list of all available fields for the Select Horse device design window.

## Hiding Fields in the Subfile Record Format

The Edit Screen Format Details panel contains a list of all fields associated with this function. You may not want to show all of the fields on your panel. You can hide fields you do not want displayed by typing **H** in the Subfile selector of the Edit Screen Format Details panel.

In this step you will hide the following non-key fields: Horse value, Dam Horse code, Dam name, Dam Date of birth, Sire Horse code, Sire name, and Sire Date of birth. Type **H** next to these fields.

```

EDIT SCREEN FORMAT DETAILS                               My model
Format . . . . . : Subfile record.                     Type: RCD

Blank lines before fmt . . . . . : 1 or Fixed start line no . . . . . : __
Blank lines after column headings: __ Blank line between records . . . . : __
Subfile page . . . . . : __

? Field                               Func Typ Usg Ovr Length GEN name Etp Rqd LL
- *SFLSEL                             ACT STS I I 1 *SFLSEL U C
- Horse code                          DTA CDE 0 0 6 ACD0 K Y C
- Horse name                          DTA TXT 0 0 25 ADTX A C
- Horse gender                        DTA STS 0 0 1 A0ST A C
H Horse value                         DTA VAL 0 0 11.2 ABVA A C
- Date of birth                       DTA DT# 0 0 6.0 ACD2 A C
H Dam Horse code                      DTA REF 0 0 6 AFCD A C
H Dam name                            DTA REF 0 0 25 AFTX V C
H Dam Date of birth                   DTA REF 0 0 6.0 ADD2 V C
H Sire Horse code                     DTA REF 0 0 6 AGCD A C +

SEL: Z-Details, A,B,C,D-Text position, I,O,H,''-Field usage.
F3=Exit F7=Fmt rel F10=Sequence F19=Add function field F23=Add constant

```

Press Roll Up to display more fields. Type **H** next to Sire Name and Sire Date of Birth.



EDIT SCREEN FORMAT DETAILS

My model

Format : Subfile record. Type: RCD

Blank lines before fmt : 1 or Fixed start line no :

Blank lines after column headings: Blank line between records :

Subfile page :

?	Field	Func	Typ	Usq	Ovr	Length	GEN name	Etp	Rqd	LL
H	Sire name	DTA	REF	0	0	25	AGTX	V		C
H	Sire Date of birth	DTA	REF	0	0	6.0	AED2	V		C

SEL: Z=Details, A,B,C,D=Text position, I,O,H,'-'=Field usage.  
F3=Exit F7=Fmt rel F10=Sequence F19=Add function field F23=Add constant

Press Enter. Note that the value of the Ovr (Override) column for the selected fields changed from O to H.

Press F3 to return to the Device Design Editor.

Modified Subfile Control Format

Note that the hidden fields are no longer shown on the subfile record format of the device design for the Select Horse function.

Select Horse

: Horse

: code

:

: 1=Select

:

Opt	Horse code	Horse name	Horse gender	Date of birth
■	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66
-	000000	00000000000000000000000000000000	0	65/65/66

F3=Exit F4=Prompt

Exercise

Complete the following exercise to modify the device design for the Select Horse function. Use function keys as you did for the Edit Horse function previously in this tutorial.

Modify the device design for the Select Horse function so that it reflects the design that follows.

```

.....
:                                     Select Horse
:                                     :
: Horse                             :
: code                             :
:                      :
:                                     :
: 1=Select                         :
:                                     :
: Opt   Horse   Horse               Horse   Date of
: code  code    name                 gender  birth
: -----
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
:                                     :
: F3=Exit   F4=Prompt
:                                     :
: .....

```

## Window Options Editor

In this step you will reduce the dimensions of the Select Horse window so that it just surrounds all remaining fields as part of the device design. You can access the Window Options Editor from the function title. Position the cursor on the function title.

```

.....
:                                     Select  Horse
:                                     :
: Horse                             :
: code                             :
:                      :
:                                     :
: 1=Select                         :
:                                     :
: Opt   Horse   Horse               Horse   Date of
: code  code    name                 gender  birth
: -----
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
: - 000000 0000000000000000000000 0 66/66/66
:                                     :
: F3=Exit   F4=Prompt
:                                     :
: .....

```

Press Enter.

The Edit Function's Window Options panel contains the parameters that control the size and location of a window on a panel. You can change the default values and value limits for the window depth, width, and position as needed. In this tutorial, you will modify the depth and width of the window.

### Changing Window Dimensions

Change the width and depth of the window to include just the fields on the device design. You can feel free to experiment by trying several combinations until you achieve a panel layout that meets your needs. For this tutorial, change the window so that the depth is 17 and the width is 62.

EDIT FUNCTION'S WINDOW OPTIONS

My model

File Name . . . : Horse

Function name . : Select Horse

Size. . . . :-

Depth. . . . . 17 5-22

Width. . . . . 62 5-76

Location. . . . . A A=\*Auto, U=\*User

Position corner at:-

Row. . . . . 1 1-21

Column . . . . . 1 1-74

Corner to be positioned. . . . . TL TL, TR, BL, BR

F3=Exit

Press Enter and press F3 to return to the Device Design Editor.

### Completed Device Design

When you return to the Device Design Editor, the window's borders should surround all of the fields. The device design for the Select Horse window is now complete.

Select Horse

: Horse

: code

:

: i=Select

:

: Opt Horse Horse Horse Date of

: code name gender birth

: 000000 000000000000000000000000 0 66/66/66

: 000000 000000000000000000000000 0 66/66/66

: 000000 000000000000000000000000 0 66/66/66

: F3=Exit F4=Prompt

:

:

To exit the Device Design Editor, press F3.

EDIT FUNCTION DEVICES		My model
Function name. . . :	Select Horse	Type : Select record
Received by file :	Horse	Acpth: Retrieval index
? Title		
Screen title.....	■ <u>Select Horse</u>	
SEL: Z=Scr/rpt design, N=Narrative, A-Animate F3=Exit F5=Action diagram F15=Open Functions		

## Exiting the Device Design

Exit this function the same way as you exited from the Edit Horse function. Press F3 to return to the Exit Function Definition panel.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . .	<u>Y</u>	Y=Yes, N=No
Function name . . . . .	<u>Select Horse</u>	Name
Access path name. . . . .	<u>Retrieval index</u>	Name
File name . . . . .	<u>Horse</u>	Name
Function type . . . . .	Select record	
Print function. . . . .	<u>N</u>	Y=Yes, N=No
Return to editing . . . . .	<u>N</u>	Y=Yes, N=No
Submit generation . . . . .	<u>N</u>	Y=Yes, N=No
F5=Refresh F12=Cancel F15=Open Functions		

To save the function, accept the default of Y and press Enter. After your device design is saved, you will be returned to the Edit Function panel.

## Exercises

Before continuing with the rest of the tutorial, complete the following exercises.

1. Create the default functions for the RACE and JOCKEY files as you did for the COURSE and HORSE files. You will create the default functions for RACE ENTRY later in this chapter.

**Note:** Remember that CA 2E automatically creates the default functions for a file the first time you type **F** against any relation for the file on the Edit Database Relations panel.

2. Check the device design for each of the following external functions to improve its appearance and to ensure that the device design does not exceed the size of the panel: Edit Course, Select Course, Edit Race, Select Race, Edit Jockey, Select Jockey, Select Stallions, and Select Mares. Note that a device design that exceeds the size of the panel will cause errors when you generate the functions in the *Generating, Compiling, and Executing* chapter.

**Note:** You access the device design for a function by typing **S** against it on the Edit Functions panel or by pressing F9 from the Edit Function Details panel. To edit the device design, use the function keys you just used to edit the device designs for the Edit Horse and Select Horse functions. Press Help or refer to the table earlier in this topic for a list of function keys.

## Action Diagrams

This topic introduces action diagrams and the CA 2E Action Diagram Editor.

New terms introduced:

- CA 2E action diagram
- Action diagram action
- Sequential construct
- Iterative construct
- Conditional construct
- Hidden construct
- User point
- Context
- CA 2E message function
- CA 2E function parameter

New panels introduced:

- Edit Action Diagram
- Edit Action Condition
- Edit Action

- Edit Message Function
- Edit Function Parameters
- Edit Message Function Details

## Overview of Action Diagrams

An *action diagram* contains the processing steps that make up a function. Each action diagram consists of a list of *actions*, where each action may be either a call to another function or one of a number of low level built-in functions; for example, \*ADD.

The sequence in which actions are executed is controlled by three simple constructs: sequential, conditional, and iterative. The constructs specify a list of actions, and optionally, when and how to execute those actions. Constructs are the basic building blocks of an action diagram. They are always executed from top to bottom. In addition, constructs may be nested; in other words, an action within a construct may be another construct.

Following are brief descriptions of each of the three constructs and the way in which each is shown on the action diagram. Refer to the diagram following these descriptions as you read.

- **Sequential construct**—A sequential construct is the simplest construct. It is a list of actions or other constructs to be executed in the order in which they appear in the action diagram. It is shown on the action diagram enclosed by a bracket of dots (:).
- **Conditional construct**—A conditional construct specifies a condition and a series of actions to be taken if the condition is true. It is equivalent to an IF THEN ELSE logic statement or a SELECT set. This construct appears on the action diagram between CASE and ENDCASE statements and is enclosed by a bracket of broken vertical bars (|). You can specify several mutually exclusive conditions in a single conditional construct.
- **Iterative construct**—An iterative construct includes a list of actions that are to be executed while an initial condition is true. It is equivalent to a DO WHILE logic statement. An iterative construct appears on the action diagram between REPEAT WHILE and ENDWHILE statements. It is indented and enclosed by a bracket of solid vertical bars (|). The controlling condition is specified at the beginning of the bracket.

The following diagram shows the general structure of an CA 2E action diagram and the three constructs.

...		
: action 1		⇐ begin Sequential construct
: -CASE		⇐ begin Conditional construct
:   -controlling condition 1		
:   action 2		
:   -controlling condition 2		
:   =REPEAT WHILE		⇐ begin Iterative construct
:     -controlling condition 3		
:     action 3		
:     action 4		
:   -ENDWHILE		⇐ end Iterative construct
: -ENDCASE		⇐ end Conditional construct
...		⇐ end Sequential construct

Default Action Diagram

CA 2E supplies a default action diagram for each function. In this step you will view and edit the default action diagram for the Edit Horse function.

From the Edit Function panel type **F** in the Subfile selector for the Edit Horse function.

EDIT FUNCTIONS		My model
File name. . . : Horse		** 1ST LEVEL **
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
F Edit Horse	Edit file	Retrieval index
■ Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
		More...
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, H-Harr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press Enter to view the action diagram for the Edit Horse function.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

█ > Edit Horse
---
---  ...Initialize <--
---  ...REPEAT WHILE
---  ...-ALWAYS
---  ...Load first subfile page <--
---  PGM.*Reload subfile = CND.*NO
---  > Conduct screen conversation
---  ...REPEAT WHILE
---  ...-PGM.*Reload subfile is *NO
---  ...Display screen
---  ...Process response <--
---  ...-ENDWHILE
---  ...-ENDWHILE
---  ...Closedown <--
---  ...
---

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

The Action Diagram Editor's Subfile selector provides options for editing an action diagram, including options for inserting and manipulating user-defined actions. This topic demonstrates many of these options. You can type ? in the Subfile selector to view a list of all allowed values.

## Hidden Constructs and User Points

The action diagram is initially shown at a summary level, with many *hidden constructs*. This allows you to see the entire action diagram in outline form on a single panel. The names of hidden constructs are preceded by three dots (for example, ...Initialize). The name of a complete construct is preceded by an angle bracket (for example, >Edit Horse).

You can alter or add to the default processing by editing the action diagram. Only certain parts of the action diagram may be altered. The areas that you can modify are called *user points* and are indicated by an angle bracket (<□) in the right margin of the panel. The rest of the default action diagram is protected to ensure that you do not accidentally change essential function processing.

You can display a selection list of all the user points for a function by pressing F5. You will use this feature later in this tutorial.

Within a user point, each user-defined action is indicated by three chevrons (<<<) in the right margin of the panel. In addition, if you press F5 to view the list of all user points, user points containing user-defined actions are indicated by three chevrons in the right margin.



## Edit Horse Action Diagram

Apart from initialization, the action diagram for the Edit Horse function consists essentially of an iteration of two steps. The first step displays a panel to the user. The second processes the response that is entered. We will examine the second step in more detail. It includes validation and update processing.

The basic validation the default Edit Horse function performs is derived from the definition of the HORSE file relations, fields, and conditions you specified in the data model. In this tutorial, you will add extra validation routines to check that a horse is younger than its parents. To do this, you will need to find the relevant part of the action diagram to edit.

## Displaying Hidden Constructs in a Action Diagram

You can view the contents of a hidden construct by typing **S** (show) against any line indicated by three dots (...) and pressing Enter. This expands the selected hidden construct. Note that when you expand a hidden construct the three dots change to **>** to indicate that the entire construct is displayed. You can hide the construct again with a Subfile selector value of **H** (hide).

When hidden constructs are displayed, the whole action diagram occupies more than one panel. To simplify editing an action diagram, you can display hidden constructs individually by typing **Z** (zoom) in the Subfile selector. You will use this in the next step.

As you will see in the following steps you can zoom into hidden constructs at continually deeper levels of the action diagram. At any point you can reverse the effect of the last **Z** you typed by typing **U** (unzoom) in the Subfile selector. To return to the topmost level, type **T** (top) in the Subfile selector.

## Adding Extra Validation to Edit Horse

To add extra validation conditions to the Edit Horse function, you must access the part of the action diagram that processes the data keyed into the displayed subfile page. This is the hidden ...Process response construct. Note the arrow in the right margin (<—). This indicates that this construct contains one or more user points where you can add user-defined validation.

To display this construct, type **Z** against ...Process response.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

___ > Edit Horse
___ .---
___ . ...Initialize <--
___ . =REPEAT WHILE
___ .   - *ALWAYS <--
___ .   ...Load first subfile page
___ .   PGM.*Reload subfile = CND.*NO
___ .   > Conduct screen conversation
___ .   =REPEAT WHILE
___ .   | -PGM.*Reload subfile is *NO
___ .   |   Display screen
___ .   |   ...Process response <--
___ .   | -ENDWHILE
___ .   -ENDWHILE
___ .   ...Closedown <--
___ . ---

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

## The Process Response Construct

The Process response construct is now displayed. It is made up of four mutually exclusive actions based on which function key was pressed. For example, the Exit key exits the program, the Next page key loads another subfile page, and the Reset key reloads the subfile page to process the input data. In this case, you will zoom into the hidden ...Process screen construct.

Type **Z** against ...Process screen.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

___ > Process response
___ .-CASE
___ . | -CTL.*CMD key is *Exit
___ . |   ...Exit program
___ . | -CTL.*CMD key is *Next page
___ . |   ...Load next subfile page ==>
___ . | -CTL.*CMD key is *Reset
___ . |   PGM.*Reload subfile = CND.*YES
___ . | -CTL.*CMD key is *Help
___ . |   ...Process help request
___ . | - *OTHERWISE
___ . |   ...Process screen <--
___ . -ENDCASE

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

The Process Screen Construct

The device design for the Edit Horse function is made up of two parts: a subfile control format and a repeating subfile record format. Default processing for this function involves validation of the subfile control and subfile record before updating the database file.

Zoom into the ...Validate subfile construct by typing **Z** against it.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                   Edit Horse

--- > Process screen
--- .--
--- . ...Validate subfile control <--
--- . <--QUIT if errors
Z  . ...Validate subfile <--
--- . <--QUIT if errors
--- . Display confirm prompt
--- . -CASE
--- .   -PGM.*CONFIRM is Do not confirm
--- .   <--QUIT
--- . -ENDCASE
--- . ...Update DBF from subfile <--
--- . <--QUIT if errors
--- . Request subfile reload if necessary
--- . ...Process function keys <--
--- .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

Press Enter.

The Validate Subfile Record Construct

Note the hidden ...Validate subfile record construct that results when you expanded the ...Validate subfile construct. Validation of subfile record input is carried out on any subfile records that have been changed by the end user. Subfile record fields and relations are automatically validated according to the specifications declared in the design process. This is summarized as follows:

- Fields must satisfy their data types; for example,  
Date of birth must be a valid date.  
Horse value must be numeric. Note that zero is valid because it is an optional relation.  
Horse name must be present and lower case is valid.
- Fields must satisfy any check conditions; for example,  
Horse gender must be male (M) or female (F).

- Any file-to-file relationships must be satisfied; for example,  
Dam code, if specified, must be the code of a valid horse. In addition, the horse must be female because the HORSE Refers to HORSE For Dam relation uses the Mares access path that selects female horses only.  
Sire code, if specified, must be the code of a valid horse. The horse must be male because the HORSE Refers to HORSE For Sire relation uses the Stallions access path that selects male horses only.

Type **Z** against ...Validate subfile record.

```
EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

___ > Validate subfile
___ .--
___ . Read first changed subfile record
___ . =REPEAT WHILE
___ .   -Changed subfile record found
___ Z .   ...Validate subfile record          <--
___ .   <--QUIT if errors
___ .   Update subfile record
___ .   Read next changed subfile record
___ . -ENDWHILE
___ .--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys
```

Press Enter.

## Adding a Validation Procedure

In this step you will use the action diagram editor to add your own validation procedure. Add the procedure to the USER: Validate subfile record relations user point.

Type **Z** against ...USER: Validate subfile record relations.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                    Edit Horse

___ > Validate subfile record
___ .--
___ . Check fields
___ . <--QUIT if errors
___ . ...USER: Validate subfile record fields      <--
___ . <--QUIT if errors
___ . Check relations
___ . <--QUIT if errors
___ . ...CALC: Subfile record function fields      <--
21 . ...USER: Validate subfile record relations  <--
___ . --

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys
```

Press Enter.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                    Edit Horse

___ > USER: Validate subfile record relations      <<<
___ .--
___ . --

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys
```

Subfile Record Relations

The three chevrons (<<<) in the right margin of the panel indicate that you may add your own processing at this point.

Editing the Action Diagram

This topic describes some of the Subfile selector values available for editing an action diagram. You can type ? in the Subfile selector to view a selection list of all values. Note that these Subfile selector values are sometimes referred to as *commands*.

Most of the Subfile selector values shown here will be demonstrated in this and the following topics. To learn about additional features of the Action Diagram Editor, refer to the online Help or the CA 2E guide, *Building Applications*.

**Showing and hiding hidden constructs**—These Subfile selector values were covered previously in this topic; this is just a review. To display a hidden construct individually use **Z** (zoom). To reverse the effect of the last **Z** you typed, use **U** (unzoom); to return to the topmost level, use **T** (top).

To hide a low level construct, use **H** (hide). The value **H** is generally used to simplify an action diagram display so it is easier to follow. To view the contents of a hidden construct, use the value **S** (show).

**Inserting new constructs**—Use the following Subfile selector values to add new constructs to an action diagram:

Subfile Selector Value	Inserts
IA	An action
IS	A sequence of actions (Sequence construct)
II	An iteration loop (Iterative construct)
IC	A case condition (Case construct)
IX	A new condition within a Case construct

The new construct starts on the line following the one on which the insertion was specified. Inserted constructs are initially defined in general terms. For example:

```
.-CASE
| -!!! New condition
|   !!! Undetermined action
| -ENDCASE
```

This is a generic definition of a conditional construct; where, **!!!** New condition indicates a conditional statement and **!!!** Undertermined action indicates the action to be performed if the condition is true. You can use these general structures to check the program logic. After you are sure the logic is correct, you can specify the details of the conditions and actions.

**Fast paths for inserting actions**—The following Subfile selector values provide fast paths that you can use to enter various actions.

Subfile Selector Value	Inserts
I*	A comment
I+	An *ADD built-in function
I-	A *SUB built-in function
I=	A *MOVE built-in function
IM	A Message function

**Entering construct details**—You can take an additional fast path to defining the details for any of the fast path constructs/actions by appending **F** to the Subfile selector value (for example, **IAF**). Including the **F** causes CA 2E to display the appropriate panel where you enter details for the construct or action.

**Moving and copying constructs**—To move or copy a construct, type **M** or **C** in the Subfile selector of the selected construct. Type **A** (after) on the line above the location in the action diagram to which you want to move or copy the construct. The construct will be placed After this line. Alternatively, you can type **B** (before) on the line below the target location. The construct will be placed Before this line.

You can also move or copy a block of constructs using the **MM** (move block) or **CC** (copy block) commands. These commands are used in pairs; for example, type **MM** on the first line in the block to be moved and type another **MM** on the last line of the block to be moved. The entire block must be at the same construct level as indicated by the action diagram indentations. Use **A** (after) or **B** (before) as for the **M** and **C** commands.

**Note:** CA 2E provides a Notepad utility that lets you copy constructs from one action diagram to another. The Notepad is a temporary action diagram that serves as a work area where you can copy and append constructs from the current action diagram. Press F18 to toggle between the action diagram and the Notepad.

**Deleting constructs**—To delete a construct, type **D** against the first line of the construct. You can also use a pair of **DD**'s to delete a block of constructs at the same construct level.

## Editing the Edit Horse Action Diagram

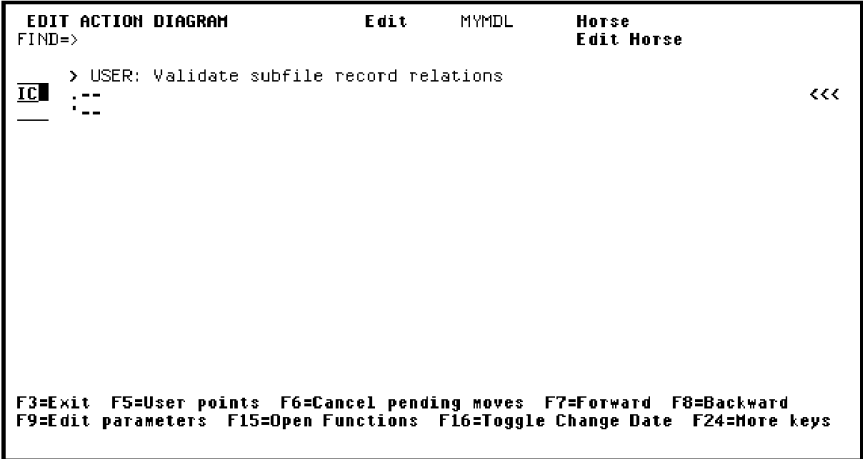
In this tutorial, you will change the action diagram for the Edit Horse function as follows. Your objective is to insert logic to validate that a horse's Date of birth is later than that of both its Dam and Sire; otherwise, display an error message.

To do this you will add:

- Two case conditions
- Two actions

## Inserting a Condition

Insert the first condition by typing **IC** on the line showing the three chevrons (<<<) in the right margin.



The screenshot shows a window titled "EDIT ACTION DIAGRAM" with a menu bar containing "Edit", "MYMDL", and "Horse". Below the menu bar, the text "Edit Horse" is visible. The main area of the window contains the following text:

```
FIND=>  
> USER: Validate subfile record relations  
IC :--<<<  
--
```

At the bottom of the window, a status bar displays the following text:

```
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward  
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

Press Enter.

## Inserting a Single Action

An empty CASE construct has been inserted in the action diagram. The CASE construct by itself only specifies the condition and not the action that is to be taken if the condition is satisfied. You will now add this action.

In this case, you only require a single action. As a result, type **IA** (Insert action).



```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                    Edit Horse

___ > USER: Validate subfile record relations
___ .--
IA . .-CASE
   . .-!!! New condition
   . .-ENDCASE
___ .--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys
```

Press Enter to insert the new action.

**Note:** If you wanted more than one action to result from the condition, you could add more actions using the **IA** option repeatedly or by inserting a sequence construct using the **IS** option.

The logic of the action diagram for the first validation is now complete.

Adding Another Condition by Copying

In this step you will add a second condition to validate a horse’s date of birth with respect to that of its Sire. You can use the same process you used in the previous step or you can copy the first construct. In this case you will copy the first construct.

Type **A** on the line above the location where you want the construct copied; type **C** next to the first line of the construct you want to copy.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                    Edit Horse

A ___ > USER: Validate subfile record relations
C ___ .--
   . .-CASE
   . .-!!! New condition
   . .-!!! Undetermined action
   . .-ENDCASE
___ .--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys
```

Press Enter.

## Specifying Details of Conditions

Now that both case conditions have been added, you are ready to specify the details of the conditions and actions. In this step you will specify the details of the condition: Is the dam's date of birth greater than or equal to that of the horse. You will specify the action to send an error message later.

First you will specify the controlling condition for the first !!! Undetermined action. Type **F** against the first condition as shown to edit that line of the action diagram.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

___ > USER: Validate subfile record relations
___ .--
___ .-CASE
F  .-!!! New condition
___ .-!!! Undetermined action
___ .-ENDCASE
___ .-CASE
___ .-!!! New condition
___ .-!!! Undetermined action
___ .-ENDCASE
___ .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter to display the Edit Action - Condition window.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

___ > USER: Validate
___ .--
___ .-CASE
F  .-!!! New condi
___ .-!!! Undetermi
___ .-ENDCASE
___ .-CASE
___ .-!!! New condi
___ .-!!! Undetermi
___ .-ENDCASE
___ .--

EDIT ACTION - CONDITION
Title : !!! New condition
Context.Field . . . : 2 ?
Condition . . . . : ?
OR
Comparison . . . . :
Context.Field . . . :
F3=Exit F7=Edit Compound Condition

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

## The Edit Action Condition Window

The Edit Action - Condition window lets you enter the terms of the condition that you require. Type the name of the field to be evaluated and either the condition it must satisfy or the name of another field to compare it to.

If you type ? in any field on this panel, other than the Comparison field, CA 2E displays a list of possible values.

## Contexts

Within a function there are several possible sources for a field's value; for example, a field may be found on the panel display, in a database file, or as an entry parameter. An CA 2E *context* specifies which source of a particular field to use. Different contexts are available at different points in a function's action diagram.

A context is identified by a three-character mnemonic; for example, CTL - subfile control, RCD - subfile record, CON - constant data, and PAR - entry parameter. This mnemonic is prefixed to each field used in an action diagram to show the source of the field. For example, RCD.Date of birth means that the value of the Date of birth field displayed in the subfile record is to be used in the action diagram.

When you enter ? to view a list of possible contexts, the most sensible context appears at the top of the selection list.

## Adding Details of the First Condition

In this step, you will test that a horse is younger than its dam; in other words, you will test whether the Dam Date of birth is greater than or equal to the horse's Date of birth. Since dates of field type DT# are stored as an alphanumeric, YYYY-MM-DD, you can do a direct comparison of these dates.

To express the condition, you will use the GE (greater than or equal) comparison operator. The basic question that needs to be asked is:

Is Dam Date of birth GE Date of birth?

If this condition is true, display an error message; if the condition is false, continue to the next action.

Note that when you add this condition to the action diagram for Edit Horse, the context for both fields in the comparison will be RCD since they are both from the subfile record.

Type the details of the comparison as shown; namely, type **RCD** and **Dam Date of birth** for the first context and field of the condition, blank the Condition field, and type **RCD** and **Date of birth** for the second context and field.

EDIT ACTION DIAGRAM	Edit	MYMDL	Horse Edit Horse
FIND=>			
> USER: Validate	EDIT ACTION - CONDITION		
---	Title. : !!! New condition		
F	-CASE	Context.Field . . . .	RCD Dam Date of birth
---	!!! Undetermi	Condition . . . . .	
---	-ENDCASE	OR	
---	-CASE	Comparison. . . . .	GE
---	!!! Undetermi	Context.Field . . . .	RCD Date of birth
---	-ENDCASE	F3=Exit F7=Edit Compound Condition	
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys			

Press Enter to return to the action diagram.

## Adding the Second Condition

Notice that the test for the condition has been updated. You are now ready to edit the second condition. To edit the second condition, type **F** against **!!! New condition**.

EDIT ACTION DIAGRAM	Edit	MYMDL	Horse Edit Horse
FIND=>			
> USER: Validate subfile record relations			
---	---		<<<
---	-CASE		<<<
---	-RCD.Dam Date of birth GE RCD.Date of birth		<<<
---	!!! Undetermined action		<<<
---	-ENDCASE		<<<
---	-CASE		<<<
F	!!! New condition		<<<
---	!!! Undetermined action		<<<
---	-ENDCASE		<<<
---	---		<<<
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys			

Press Enter.

### Adding Details of the Second Condition

The second condition is similar to the first. In this case, you need to compare Sire Date of birth to Date of birth. Again, both fields have the RCD context since they are from the subfile record.

Type the comparison details as shown; namely, type **RCD** and **Sire Date of birth** for the first context and field of the condition, blank the Condition field, and type **RCD** and **Date of birth** for the second context and field.

EDIT ACTION DIAGRAM	Edit	MYMDL	Horse
FIND=>			Edit Horse
-----			
> USER: Validate	EDIT ACTION - CONDITION		
----	-----		
..-CASE	Title. : !!! New condition		
..-RCD.Dam Date	Context.Field . . . : RCD Sire Date of birth		
..!!! Undetermi	Condition . . . . .		
..-ENDCASE	OR		
..-CASE	Comparison. . . . . GE		
..-!!! New condi	Context.Field . . . : RCD Date of birth		
..-!!! Undetermi	-----		
..-ENDCASE	F3=Exit F7=Edit Compound Condition		
----	-----		

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward  
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

Press Enter.

### Adding Actions

In the following steps you will add two actions to display an error message at the bottom of the panel when either of the two conditions you just defined is true. This requires that you:

1. Define two message functions
2. Define parameters for the message functions
3. Change the text of the error messages to include the dates that caused the error using substitution variables
4. Select the message functions as the actions within the action diagram

### Exiting the Action Diagram

There are two ways to define a message function; you will use both in this tutorial. The first requires that you exit the action diagram and return to the Edit Database Relations panel.

Press F13 to Exit the Action Diagram Editor and display the Exit Function Definition panel. Accept the defaults on this panel to save all changes you made to the action diagram.

**Note:** The F13 function key (Fast exit) provides a quick way to exit the action diagram. You can use F3, but it will take you back through the action diagram one level at a time.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<u>Y</u>	Y=Yes, N=No
Function name . . . . .	<u>Edit Horse</u>	Name
Access path name. . . . .	<u>Retrieval index</u>	Name
File name . . . . .	<u>Horse</u>	Name
Function type . . . . .	Edit file	
Print function. . . . .	<u>N</u>	Y=Yes, N=No
Return to editing . . . . .	<u>N</u>	Y=Yes, N=No
Submit generation . . . . .	<u>N</u>	Y=Yes, N=No
F5=Refresh F12=Cancel F15=Open Functions		

Press Enter to return to the Edit Functions panel.

The message "Function 'Edit Horse has been saved" displays at the bottom of the panel.

EDIT FUNCTIONS		My model
File name. . . . : Horse		** 1ST LEVEL **
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
█ Edit Horse	Edit file	Retrieval index
- Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Harr, O-Open,		
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.		
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press F3 to return to the Edit Database Relations panel.

EDIT DATABASE RELATIONS

My model

=> Horse\*

? Typ Object

FIL Horse

FIL Horse

FIL Horse

FIL Horse

FIL Horse

FIL Horse

For: Dam

FIL Horse

For: Sire

Rel lvl:

Relation

Seq

Typ

Referenced object

known by

10

FLD

Horse code

Has

20

FLD

Horse name

Has

30

FLD

Horse gender

Has

40

FLD

Horse value

Has

50

FLD

Date of birth

Refers to

60

FIL

Horse

Refers to

70

FIL

Horse

Sharing: \*ALL

Sharing: \*ALL

More...

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options

F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

Message Functions

All message functions for CA 2E programs are stored in a system file called \*Messages. Begin by positioning the Edit Database Relations panel to this file. To do so, type \*M on the selection line.

EDIT DATABASE RELATIONS

My model

=> \*M

? Typ Object

FIL Horse

FIL Horse

FIL Horse

FIL Horse

FIL Horse

FIL Horse

For: Dam

FIL Horse

For: Sire

Rel lvl:

Relation

Seq

Typ

Referenced object

known by

10

FLD

Horse code

Has

20

FLD

Horse name

Has

30

FLD

Horse gender

Has

40

FLD

Horse value

Has

50

FLD

Date of birth

Refers to

60

FIL

Horse

Refers to

70

FIL

Horse

Sharing: \*ALL

Sharing: \*ALL

More...

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options

F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

Press Enter.

Type F against one of the relations for the \*Messages file to display the existing message functions attached to this file.

EDIT DATABASE RELATIONS				My model		
=>	*M			Rel lvl:		
? Typ	Object			Relation	Seq	Typ Referenced object
F FIL	*Messages			Has		FLD *MSGID
— FIL	*Messages			Has		FLD *MSGDTA
— FIL	*Messages			Has		FLD *MSG
— FIL	*Program data			Has	1	FLD *Program mode
— FIL	*Program data			Has	2	FLD *Return code
— FIL	*Program data			Has	3	FLD *Record selected
— FIL	*Program data			Has	4	FLD *Reload subfile
— FIL	*Program data			Has	5	FLD *Scan limit
— FIL	*Program data			Has	6	FLD *Defer confirm
— FIL	*Program data			Has	7	FLD *Print format
— FIL	*Program data			Has	8	FLD *Continue transaction
— FIL	*Program data			Has	9	FLD *Next RDB
— FIL	*Program data			Has	10	FLD *Re-read Subfile Record
— FIL	*Program data			Has	11	FLD *Cursor field
— FIL	*Program data			Has	12	FLD *Cursor row

More...

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options  
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

Press Enter.

## Displaying the Message Functions

The Edit Message Functions panel displays a list of the message functions available on the file \*Messages. A number of general purpose messages are supplied with CA 2E. The types of messages available include, ERR (Error message), INF (Information message), EXC (Execution message), and STS (Status message). You can scroll through the messages by pressing Roll Up and Roll Down.

You can select one of the messages displayed or you can create your own. To send an explicit message about a horse, you will need to create a new message.

EDIT MESSAGE FUNCTIONS				My model	
File . . . : *Messages				Default msg file. . : QUSRMMSG	
				Generation library. : MYGEN	
Message				<== Position	
? Message	Type	Msgid	Ovr Msgf		
— *Accelerator key error	ERR	Y2U0029	Y2USRMMSG		
— *Action Bar API error	ERR	Y2U0027	Y2USRMMSG		
— *Action Bar not found	ERR	Y2U0028	Y2USRMMSG		
— *Array full-cannot add...	ERR	Y2U0036	Y2USRMMSG		
— *Arrays	EX	Y2U0035	Y2USRMMSG		
— *Arrays	NF	Y2U0034	Y2USRMMSG		
— *CHECK(MF) validation msg	ERR	Y2U0020	Y2USRMMSG		
— *Clearing status message	STS	Y2U0041	Y2USRMMSG		
— *Configuration Table	EX	Y2U0048	Y2USRMMSG		
— *Configuration Table	NF	Y2U0047	Y2USRMMSG		
— *Confirm Connect to Nxt	INF	Y2U0025	Y2USRMMSG		
— *Confirm Connect to Prv	INF	Y2U0049	Y2USRMMSG		
— *Connect failed	INF	Y2U0040	Y2USRMMSG		

+

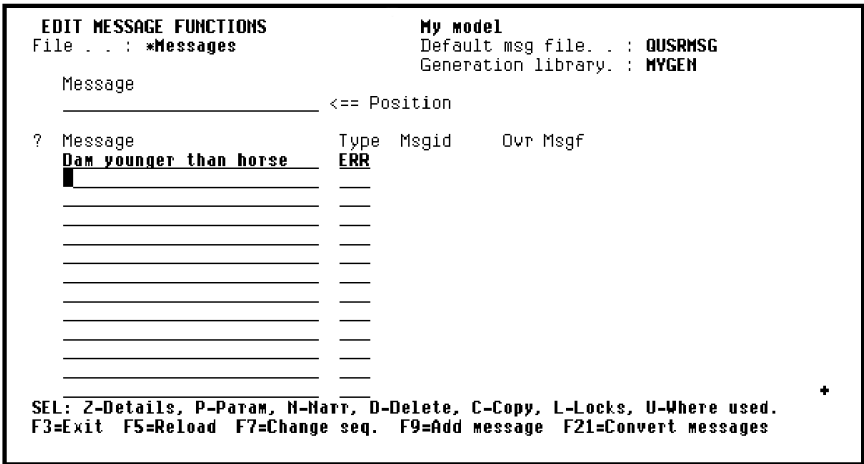
SEL: Z-Details, P-Param, N-Name, D-Delete, C-Copy, L-Locks, U-Where used.  
F3=Exit F5=Reload F7=Change seq. F9=Add message F21=Convert messages



### Adding a New Message Function

Press F9 to add a new message function that will be executed if Dam Date of birth is greater than or equal to Date of birth.

Name the message function Dam younger than horse. This is to be an error message so the type will be ERR. Type the message name.



Press Enter. The new message function is added to the \*Messages file.

**Note:** You could also add the message function to be executed when Sire Date of birth is greater than or equal to Date of birth now. Instead you will use another method for creating message functions to add the second message function later in this tutorial.

### Defining Parameters

Parameters define the values to be communicated between functions at execution time. How a parameter is used in the called function will depend on the type of the called function, the processing specified for that function, or both.

The next step is to define the parameters that will be passed to the message function by the calling program, Edit Horse. The objective is to include the dates that caused the error in the text of the error message.

Specifying a field as a parameter to a message does two things.

First, it makes the field available for use as a substitution variable in the message. In this case, the error message text will include the current values for Dam Date of birth and Date of birth.

Second, for input-capable fields, the field will be highlighted in reverse image on the panel if the error occurs. In this case, the Date of birth field would be highlighted.

Type **P** against the message function as shown to define its parameters.

[illegible]

Press Enter to display the Edit Function Parameters panel.

## Specifying Parameters for the First Message Function

You are now ready to specify the message parameters. There are two ways of specifying function parameters. The simplest is to name the individual fields that are to be passed as parameters (\*FIELD). In this case, you will define two fields as parameters: Date of birth and Dam Date of birth.

**Note:** The fields on this panel will be discussed in more detail in the Function Parameters topic later in this chapter.

Type the message function parameters.

EDIT FUNCTION PARAMETERS

My model

Function name. . : Dam younger than horse

Type : Send error message

Received by file : \*Messages

Acpth:

?	File/*FIELD	Access path/Field	Passed as	Seq
-	*FIELD	Date of birth	FLD	
-	*FIELD	Dam Date of birth	FLD	
-				
-				
-				
-				
-				
-				
-				
-				

Values

FLD: One parameter per field

RCD: One parameter for all fields

KEY: One parameter for key fields only

SEL: Z=Details (field selection).

F3=Exit F5=Reload

Press Enter. Your selection of input function parameters is now confirmed.

**Note:** You can view the details of parameter usage by typing **Z** against a parameter and pressing Enter. By default, the parameter usage will be input. If you display the details, be sure to press F3 to return to the Edit Function Parameters panel before you continue with the tutorial.

Press F3 to return to the Edit Message Functions panel.

Message Function Details

Once the parameters have been defined, you can display the message function details. Type **Z** next to the message function.

EDIT MESSAGE FUNCTIONS

My Model

File . . : \*Messages

Default msg file. . : QUSRMSG

Generation library. : MYGEN

Message

<== Position

?	Message	Type	Msgid	Ovr Msgf
Z	Dam younger than horse	ERR	USR0011	

SEL: Z=Details, P=Param, H=Narr, D=Delete, C=Copy, L=Locks, U=Where used.

F3=Exit F5=Reload F7=Change seq. F9=Add message F21=Convert messages

Press Enter to display the Edit Message Function Details panel. This panel displays details for the function, including the parameters you just defined. At this point you may change the message text, which has been defaulted to the name of the message function.

## Adding Text to the First Message

You will use the two parameters you just defined as substitution variables within the message text. Substitution variables are indicated by the symbol "&n", where 'n' is the number of the parameter. Note the numbers assigned to the two parameters in the No. column.

Add the text "&1 before &2)" to indicate that the date of birth of the horse is prior or equal to the date of birth of the dam.

When the message appears on the panel, the value contained in that field will be substituted in the message text. For example, the resulting message might read: "Dam younger than horse (10185 before 11294)." Substitution variables help the end user understand which fields are in error.

Type the new text containing the substitution variables, "&1 before &2)", after the default message text.

EDIT MESSAGE FUNCTION DETAILS				My Model
File name . . . : *Messages				
Message . . . : Dam younger than horse				
Message type. . : ERR (INF,ERR,STS,CMP)				
Message id. . . : USR0011		Override message file. :		*LIBL
		Default message file. :		QUSRMSG MYGEN
Severity. . . : 20				
Message text: Dam younger than horse (&1 before &2)				
Parameters . . :				
No.	Field	Type	Length	
&1	Date of birth	DT#	6.0	
&2	Dam Date of birth	REF	6.0	
F3=Exit F7=Second level text F8=Change name				

Press Enter to return to the Edit Message Functions panel.

## Returning to the Edit Horse Action Diagram

In this step you will insert the action you just defined into the Edit Horse action diagram and define another message function as the action when the horse's date of birth is earlier than its sire's date of birth. This time you will define the message function from within the action diagram.

Press F3 to return to the Edit Database Relations panel. When the Edit Database Relations panel displays, type **Horse\*** on the selection line and press Enter to display relations for the HORSE file.

EDIT DATABASE RELATIONS

My Model

=> Horse\*

Rel lvl:

Relation Seq Typ Referenced object

FIL Horse Known by 10 FLD Horse code

FIL Horse Has 20 FLD Horse name

FIL Horse Has 30 FLD Horse gender

FIL Horse Has 40 FLD Horse value

FIL Horse Has 50 FLD Date of birth

FIL Horse Refers to 60 FIL Horse

For: Dam Sharing: \*ALL

FIL Horse Refers to 70 FIL Horse

For: Sire Sharing: \*ALL

Bottom

2(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options

F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

Type **F** against one of the relations and press Enter to display functions for the HORSE file. Type **F** against the Edit Horse function.

EDIT FUNCTIONS

My model

File name: : Horse

\*\* 1ST LEVEL \*\*

? Function

Function type

Access path

Change Horse Change object Update index

Create Horse Create object Update index

Delete Horse Delete object Update index

F Edit Horse Edit file Retrieval index

Select Horse Select record Retrieval index

Select Hares Select record Hares

Select Stallions Select record Stallions

More...

SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, M-Harr, O-Open,

T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.

F3=Exit F5=Reload F7=File details F9=Add functions F17=Services

Press Enter to display the Edit Horse action diagram.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

<--
> Edit Horse
<--
...Initialize
<--
=REPEAT WHILE
  <--
  *ALWAYS
  <--
  ...Load first subfile page
  PGM.*Reload subfile = CND.*NO
  <--
  > Conduct screen conversation
  =REPEAT WHILE
    <--
    -PGM.*Reload subfile is *NO
    Display screen
    ...Process response
    <--
  -ENDWHILE
  ...Closedown
  <--
  <--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

## Displaying User Points for Edit Horse

Recall that you added the two conditions and actions at the user point USER: Validate subfile record relations. To return to that user point quickly, press F5 to display a list of all user points for the Edit Horse function. If you wish, you can press Roll Up to view the additional user points.

Type **X** (or **Z**) against USER: Validate subfile record relations.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

<--
> Edit      ACTION DIAGRAM EXIT POINTS      F3=Exit  SEL:X,Z=Select.
<--
...I      USER: Initialize program
<--
=RE      USER: Initialize subfile header
<--
  *A      USER: Initialize subfile record (existing record)
  <--
  ..      USER: Initialize subfile record (new record)
  <--
  PG      CALC: Subfile control function fields
  <--
  >      USER: Validate subfile control
  <--
  .      USER: Validate subfile record fields
  <--
  .      CALC: Subfile record function fields
  <--
  -      X USER: Validate subfile record relations      + <<<
  <--
  ...Process response
  <--
  -ENDWHILE
  <--
  -ENDWHILE
  <--
  ...Closedown
  <--
  <--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Add Action for First Condition

Type **F** against the first !!! Undetermined action as shown.

EDIT ACTION DIAGRAM

EditMYMDL

HorseEdit Horse

FIND=>

> USER: Validate subfile record relations

---

..-CASE

..-RCD.Dam Date of birth GE RCD.Date of birth

F..!!! Undetermined action

..-ENDCASE

..-CASE

..-RCD.Sire Date of birth GE RCD.Date of birth

..!!! Undetermined action

..-ENDCASE

---

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward

F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

Press Enter to display the Edit Action - Function Name window.

The Edit Action Function Name Panel

The Edit Action - Function Name window lets you specify the function to be called if the first CASE condition is satisfied.

Because all functions are associated with a file, when you specify a function you must give both a file and a function name. Functions can be divided into three groups:

- Functions attached to user-defined files. For example the Edit Horse function that you are currently editing.
- Functions attached to CA 2E shipped system files. For example, CA 2E user-defined message functions are attached to a system file called \*Messages.
- Built-in shipped functions, which are also attached to CA 2E shipped system files. Such functions provide commonly required low level actions; for example, to manipulate fields. Some examples are, \*ADD, \*SUB, and \*MOVE. If you do not specify a file name, CA 2E assumes you are referring to a built-in function.

Message Functions

Because you want to send the user-defined message you just defined, type **\*Messages** for the Function file as shown and leave the ? in the Function field.

EDIT ACTION DIAGRAM		Edit	MYMDL	Horse								
FIND=>				Edit Horse								
<pre> __ &gt; USER: Validate subfile re __ .-- __ .-CASE __ .-RCD.Dam Date of birth G __ .-!!! Undetermined action F .-ENDCASE __ .-CASE __ .-RCD.Sire Date of birth __ .-!!! Undetermined action __ .-ENDCASE __ .-- </pre>		<table border="1"> <thead> <tr> <th colspan="2">EDIT ACTION - FUNCTION NAME</th> </tr> </thead> <tbody> <tr> <td>Function file</td> <td>*Messages</td> </tr> <tr> <td>Function</td> <td>?</td> </tr> <tr> <td>Comment</td> <td></td> </tr> </tbody> </table>			EDIT ACTION - FUNCTION NAME		Function file	*Messages	Function	?	Comment	
EDIT ACTION - FUNCTION NAME												
Function file	*Messages											
Function	?											
Comment												
		F3=Exit										
<<<												
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys												

Press Enter.

## Displaying the Message Functions

The Edit Message Functions panel displays a list of the message functions available on the file \*Messages. Type **D** in the positioner field at the top of the panel as shown to position to the Dam younger than horse message function you just defined.

EDIT MESSAGE FUNCTIONS		My model	
File . . : *Messages		Default msg file. . : QUSRMSG	
		Generation library. : MYGEN	
Message	<== Position		
D			
? Message	Type	Msgid	Ovr Msgf
- *Accelerator key error	ERR	Y2U0029	Y2USRMSG
- *Action Bar API error	ERR	Y2U0027	Y2USRMSG
- *Action Bar not found	ERR	Y2U0028	Y2USRMSG
- *Array full-cannot add...	ERR	Y2U0036	Y2USRMSG
- *Arrays	EX	Y2U0035	Y2USRMSG
- *Arrays	NF	Y2U0034	Y2USRMSG
- *CHECK(MF) validation msg	ERR	Y2U0020	Y2USRMSG
- *Clearing status message	STS	Y2U0041	Y2USRMSG
- *Configuration Table	EX	Y2U0048	Y2USRMSG
- *Configuration Table	NF	Y2U0047	Y2USRMSG
- *Confirm Connect to Hxt	INF	Y2U0025	Y2USRMSG
- *Confirm Connect to Prv	INF	Y2U0049	Y2USRMSG
- *Connect failed	INF	Y2U0040	Y2USRMSG
+			
SEL: Z=Details, P=Param, N=Harr, D=Delete, C=Copy, L=Locks, U=Usage, X=Select.			
F3=Exit F5=Reload F7=Change seq. F9=Add message F21=Convert messages			

Press Enter.

## Selecting a Message Function

Type **X** against the message, Dam younger than horse, as shown to select it as the action in the action diagram.



Press Enter.

The Edit Action - Function Name window redisplay with the Function field containing the name of the Dam younger than horse message function.

Press Enter.

The details of the function, including the parameters which are to be passed to the message function, are displayed. Check that the information shown is correct.

[illegible]

Press Enter to accept the information shown and to return to the action diagram.

## Action Diagram with the First Action Defined

The action diagram now includes a call to the Dam younger than horse message function. In other words, "Send error message - 'Dam younger than horse'" has replaced the first !!! Undetermined action.

## Defining the Second Action

In this step you will define the message function to be called when the sire is younger than the horse. The entire process will take place within the action diagram. Note that you could have defined both message functions in this way.

Type **F** against the second !!! Undetermined action.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                   Edit Horse

___ > USER: Validate subfile record relations
___ .--
___ .-CASE
___ .-RCD.Dam Date of birth GE RCD.Date of birth
___ .-Send error message - 'Dam younger than horse'
___ .-ENDCASE
___ .-CASE
___ .-RCD.Sire Date of birth GE RCD.Date of birth
F1 .-!!! Undetermined action
___ .-ENDCASE
___ .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

Press Enter. A blank Edit Action - Function Name window displays as before. Type **\*Messages** for the Function file field as shown. Leave the ? in the Function name field.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                   Edit Horse

___ > USER: Validate subfile re
___ .--
___ .-CASE
___ .-RCD.Dam Date of birth G
___ .-Send error message - 'D
___ .-ENDCASE
___ .-CASE
___ .-RCD.Sire Date of birth
F1 .-!!! Undetermined action
___ .-ENDCASE
___ .--

EDIT ACTION - FUNCTION NAME
Function file : *Messages
Function. . . : ?
Comment . . . :
F3=Exit F22=File Locks

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

Press Enter.

Adding the Second Message Function

All message functions attached to the \*Messages file are displayed. You will now create a new message function to be executed if Sire Date of birth is greater than or equal to Date of birth as you did for the first message function.

```

EDIT MESSAGE FUNCTIONS
File . . . : *Messages
My model
Default msg file. . . : QUSRMSG
Generation library. : MYGEN

Message
█ _____ <== Position

? Message                                     Type   Msgid      Ovr Msgf
- *Accelerator key error                     ERR   Y2U0029    Y2USRMSG
- *Action Bar API error                     ERR   Y2U0027    Y2USRMSG
- *Action Bar not found                     ERR   Y2U0028    Y2USRMSG
- *Array full-cannot add...                 ERR   Y2U0036    Y2USRMSG
- *Arrays                                   EX    ERR   Y2U0035    Y2USRMSG
- *Arrays                                   HF    ERR   Y2U0034    Y2USRMSG
- *CHECK(HF) validation msg                 ERR   Y2U0020    Y2USRMSG
- *Clearing status message                  STS   Y2U0041    Y2USRMSG
- *Configuration Table EX                   ERR   Y2U0048    Y2USRMSG
- *Configuration Table HF                   ERR   Y2U0047    Y2USRMSG
- *Confirm Connect to Nxt                   INF   Y2U0025    Y2USRMSG
- *Confirm Connect to Prv                   INF   Y2U0049    Y2USRMSG
- *Connect failed                           INF   Y2U0040    Y2USRMSG
+

SEL: Z=Details, P=Param, N=Attr, D=Delete, C=Copy, L=Locks, U=Usage, X=Select.
F3=Exit F5=Reload F7=Change seq. F9=Add message F21=Convert messages

```

Press F9 to add the new message.

## Details of the Second Message Function

To define the second message function you need to specify a function name and a function type as you did for the first message function. Name the message function **Sire younger than horse**. The type will be **ERR**. Type the message name and type.

```

EDIT MESSAGE FUNCTIONS                                My model
File . . : *Messages                               Default msg file. . : QUSRMSG
                                                    Generation library. : MYGEN

Message _____ <== Position

? Message                      Type   Msgid      Ovr Msgf
Sire younger than horse       ERR
_____  

_____  

_____  

_____  

_____  

_____  

_____  

_____  

_____  

_____  

_____  

_____  

_____
SEL: Z=Details, P=Param, N=Narr, D>Delete, C=Copy, L=Locks, U=Usage, X>Select.  

F3=Exit F5=Reload F7=Change seq. F9>Add message F21=Convert messages

```

Press Enter.

The second message function has now been added to the \*Messages file.



[illegible]

Press Enter.

## Adding Message Function Text

Type the new text containing the substitution variables, "&1 before &2", after the default message text as shown.

```

EDIT MESSAGE DETAILS                               My model
File name . . . : *Messages

Message . . . . : Sire younger than horse
Message type. . : ERR (INF,ERR,STS,CMP)
Message id. . . : USR001Z      Override message file. : *LIBL
                                Default message file.  : QUSRMGS  HYGEN

Severity. . . . : 20
Message text: Sire younger than horse (&1 before &2)

```

---

```

Parameters . :
No.  Field                               Type  Length
&1   Date of birth                       DT#    6.0
&2   Sire Date of birth                  REF    6.0

```

F3=Exit F7=Second level text F8=Change name

Press Enter to return to the Edit Message Functions panel. Both required message functions are now completely defined.

## Selecting the Second Message Function

You can now select the message function to be executed when the second condition is satisfied. Type **X** to select the message function for the action diagram.

EDIT MESSAGE FUNCTIONS  
File . . . : \*Messages

My model  
Default msg file. . . : QUSRMSG  
Generation library. : MYGEN

Message  
\_\_\_\_\_ <== Position

? Message

X Sire younger than horse

Type

ERR

Msgid

USR0012

Ovr Msgf

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

SEL: Z=Details, P=Param, H=Harr, D=Delete, C=Copy, L=Locks, U=Where used.

F3=Exit F5=Reload F7=Change seq. F9=Add message F21=Convert messages

\*

Returning to the Action Diagram

- You will press Enter three times to return to the Action Diagram Editor as follows:
- 1. Press Enter the first time to return to the Edit Action - Function Name window showing the name of the new message function.
  - 2. Press Enter a second time to return to the Edit Action - Function Details window showing the function parameters. A message at the bottom of the window indicates that some parameters have been defaulted. Note that the default context for the parameters is RCD (subfile record).
  - 3. Press Enter the third time to accept the defaulted parameters and return to the Action Diagram Editor.

Completed Action Diagram

The USER: Validate subfile record relations construct is now complete.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse

___ > USER: Validate subfile record relations
___
___ .--
___ .-RCD.Dam Date of birth GE RCD.Date of birth      <<<
___ .-Send error message - 'Dam younger than horse'   <<<
___ .-ENDCASE                                         <<<
___ .-CASE                                           <<<
___ .-RCD.Sire Date of birth GE RCD.Date of birth     <<<
___ .-Send error message - 'Sire younger than horse'  <<<
___ .-ENDCASE                                         <<<
___
___

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

## Exiting the Action Diagram

Press F13 (Fast exit) to Exit the Action Diagram Editor and display the Exit Function Definition panel. Accept the defaults on this panel to save all changes you made to the action diagram.

```

EXIT FUNCTION DEFINITION          My model

Type choices, press Enter.

Change/create function. . . . Y      Y=Yes, N=No

Function name . . . . . Edit Horse      Name
Access path name. . . . . Retrieval index  Name
File name . . . . . Horse              Name
Function type . . . . . Edit file

Print function. . . . . N      Y=Yes, N=No

Return to editing . . . . . N      Y=Yes, N=No

Submit generation . . . . . N      Y=Yes, N=No

F5=Refresh  F12=Cancel  F15=Open Functions

```

Press Enter to return to the Edit Functions panel.

The message "Function 'Edit Horse has been saved" displays at the bottom of the panel.



EDIT FUNCTIONS		
File name. . . : Horse		
My model		
** 1ST LEVEL **		
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
█ Edit Horse	Edit file	Retrieval index
- Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, M-Harr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		
Function 'Edit Horse' has been saved.		

Function Options

This topic discusses how to change the function options for a function.

New term introduced:

- Function Option

New panel introduced:

- Edit Function Options

Objectives

In this topic you will remove the capability to delete records from the Edit Horse function and change the default confirm prompt to Y.

Accessing the Function Options Panel

From the Edit Functions panel, type **Z** against the Edit Horse function.

EDIT FUNCTIONS		My model	** 1ST LEVEL **
File name. . . : Horse			
? Function	Function type	Access path	
- Change Horse	Change object	Update index	
- Create Horse	Create object	Update index	
- Delete Horse	Delete object	Update index	
Z Edit Horse	Edit file	Retrieval index	
█ Select Horse	Select record	Retrieval index	
- Select Mares	Select record	Mares	
- Select Stallions	Select record	Stallions	
-			
-			
-			
-			
-			
-			
More...			
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, M-Marr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.			
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services			
Function 'Edit Horse' has been saved.			

Press Enter to display the Edit Function Details panel.

EDIT FUNCTION DETAILS		My model
Function name . . : Edit Horse		Type : Edit file
Received by file. : Horse		Acpth: Retrieval index
Workstation . . . : NPT		
Source library. . : MYGEN		
Object	Source	Target
? Type	Name	HLL Text
█ PGM	MYAHEFR	RPG Edit Horse Edit file
- DSP	MYAHEFRD	DDS Edit Horse Edit file
- HLP	MYAHEFRH	UIM Edit Horse Edit file
SEL: E-STRSEU, O-Compiler Overrides.		
F3=Exit F7=Options F8=Change name F9=Scr/rpt layout F20=Narrative		

## Default Function Options

To view the function options for Edit Horse, press F7. The Edit Function Options panel gives you a simple means of modifying certain aspects of the behavior of a function. The actual function options available and their defaults depend on the function type. In this step you will change the two following function options for the Edit Horse function.

### Delete record

This option specifies whether you can delete database records. The default is Y, which means you can delete database records. If you change this value to N, the function code will not include logic to delete database records.

**Confirm prompt**

The Confirm prompt option specifies whether the Edit Horse function prompts for confirmation before updating the database files. The default is Y.

If the Confirm prompt option is Y, a related option, Initially "Yes", specifies whether the default for the confirm prompt will be Y or N.

**Note:** When the default value for a function option is M, it means that the function option's default value depends on the setting of a model value. The current setting of the model value is shown highlighted or underlined.

Changing Function Options

From the Edit Function Options panel for the Edit Horse function, change the value for the Delete record option to N. Change the initial value of the confirm prompt by typing **Y** for Initially 'Yes'.

EDIT FUNCTION OPTIONS

My model

Function name . . . : Edit HorseType : Edit file  
Received by file. : HorseAcpth: Retrieval index  
Header/Footer . . . : \*STD CUA ACTION BAR<-Implicitly set by mdl default

OPTION	SEL	VALID VALUES
Create record . . . . .	Y	( Y-Yes, N-No )
Change record . . . . .	Y	( Y-Yes, N-No )
Delete record . . . . .	N	( Y-Yes, N-No )
Dynamic program mode. . . . .	N	( Y-Yes, N-No )
Subfile selection . . . . .	Y	( Y-Yes, N-No )
Confirm Prompt. . . . .	Y	( Y-Yes, N-No )
Initially "Yes" . . . . .	Y	( M-MDLVAL, Y-Yes, N-No )
Commit control. . . . .	N	( M-Master, S-Slave, N-None )
If action bar, what type? . . .	M	( M-MDLVAL, A-Action bar, D-DDS-menubar )
Generation mode . . . . .	A	( M-MDLVAL, D-DDS, S-SQL, A-ACPVAL )

More...

F3=Exit F5=Select header/footer F10=All options

Press Enter to return to the Edit Function Details panel.

EDIT FUNCTION DETAILS				My model	
Function name . . . : <b>Edit Horse</b>				Type :	<b>Edit file</b>
Received by file. : <b>Horse</b>				Acpth:	<b>Retrieval index</b>
Workstation . . . : <b>NPT</b>					
Source library. . . : <b>MYGEN</b>					
Object	Source	Target			
? Type	Name	HLL	Text		
■ PGM	<b>MYAHEFR</b>	<b>RPG</b>	<b>Edit Horse</b>		<b>Edit file</b>
— DSP	<b>MYAHEFRD</b>	<b>DDS</b>	<b>Edit Horse</b>		<b>Edit file</b>
— HLP	<b>MYAHEFRH</b>	<b>UIM</b>	<b>Edit Horse</b>		<b>Edit file</b>
SEL: E-STRSEU, O-Compiler Overrides.					
F3=Exit F7=Options F8=Change name F9=Scr/rpt layout F20=Narrative					

Press F3 to return to the Edit Functions panel.

## Linking Functions

This topic shows how to use CA 2E to link two functions together to build a larger function. In the course of doing this, you will define the second function and an access path upon which it is based.

New terms introduced:

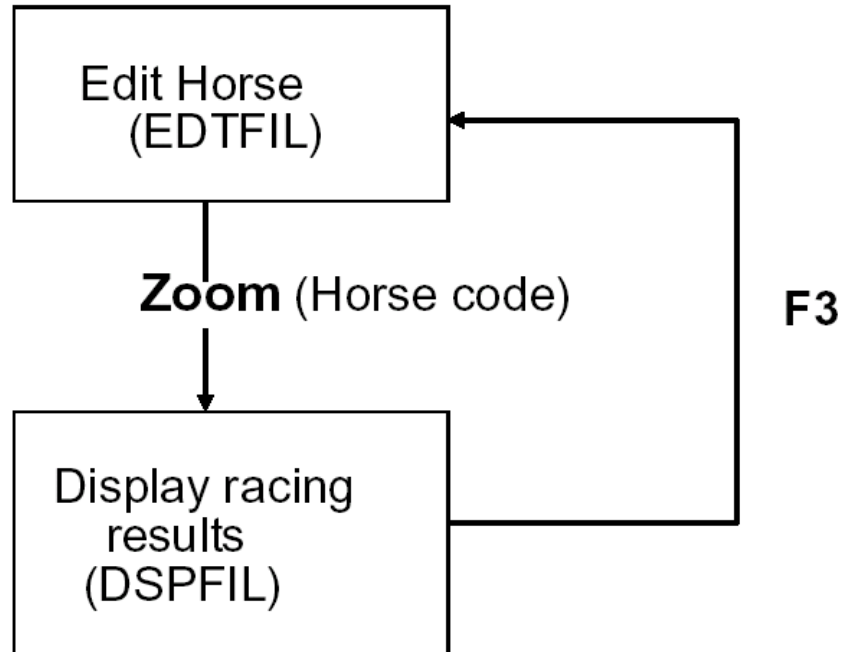
- Built-in function
- \*MOVE function

New panels introduced:

- Action Diagram User Points
- Edit Access Path Format Entries

## Objectives

Modify the Edit Horse function to give it a Subfile selector option. The new facility should let the user zoom against the detail line for a particular horse on the Edit Horse panel in order to view a subsidiary display of the races in which the horse has raced.



To achieve this, you will define a new function, Display Racing results. The new function will display information from the RACE ENTRY file for the horse selected. You also need to modify the action diagram of the existing Edit Horse function to call the new function. Note that CA 2E lets you define the new function while you are in the process of modifying the action diagram of the existing function.

The two functions will be implemented as separate programs. The Horse code of the selected line will be passed as a parameter between the two programs in order for the Display Racing results program to display race entries for a particular horse.

## Steps Required to Link Functions

In this topic, you will define a Display Racing results function and link it to the Edit Horse function. You will also update the Edit Horse function. The process is similar to that of adding a message function. To achieve the objectives for this topic you will need to:

1. Modify the action diagram of the Edit Horse function to specify the condition under which the Display Racing results function is to be called.
2. Define the Display Racing results function by giving it a name and type and selecting an appropriate access path based on the RACE ENTRY file.
3. Specify the parameters of the Display Racing results function; namely, Horse code.

4. Select the Display Racing results function for inclusion in the action diagram of the Edit Horse function.
5. Modify the action diagram of the Edit Horse function to defer updating the HORSE file if a line selection is made.
6. Modify the Selector Choice menu of the action bar of the Edit Horse function's device design to show the availability of the Display Racing results function.

## Modifying the Edit Horse Action Diagram

The first step is to modify the action diagram of the Edit Horse function so that the Display Racing results function is called when the end user types / in the Subfile selector field and selects the Display Racing results action from the Selector Choice menu of the Edit Horse function's action bar.

You will access the Action Diagram Editor for the Edit Horse function and locate the USER: Validate subfile record relations user point. Recall that user points are the areas in an action diagram that you can modify.

From the Edit Functions panel, type **F** next to the Edit Horse function to enter the action diagram.

EDIT FUNCTIONS		My model	** 1ST LEVEL **
File name. . . : Horse			
? Function	Function type	Access path	
- Change Horse	Change object	Update index	
- Create Horse	Create object	Update index	
- Delete Horse	Delete object	Update index	
F Edit Horse	Edit file	Retrieval index	
■ Select Horse	Select record	Retrieval index	
- Select Hares	Select record	Hares	
- Select Stallions	Select record	Stallions	
-			
-			
-			
-			
-			
-			
-			
		More...	
		SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Marr, O-Open,	
		T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.	
		F3=Exit F5=Reload F7=File details F9=Add functions F17=Services	

Press Enter.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>
█ > Edit Horse
---
--- ...Initialize <--
--- .=REPEAT WHILE
--- .-ALWAYS
--- . ...Load first subfile page <--
--- . PGM.*Reload subfile = CND.*NO
--- . > Conduct screen conversation
--- . .=REPEAT WHILE
--- . .-PGM.*Reload subfile is *NO
--- . .Display screen
--- . . ...Process response <--
--- . .-ENDWHILE
--- . .-ENDWHILE
--- . ...Closedown <--
---
---

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

Obtaining the Action Diagram User Points

Press F5 to view the list of user points for the Edit Horse function.

Type **X** against USER: Validate subfile record relations.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>
█ > Edit
---
--- .=REPEAT WHILE
--- .-ALWAYS
--- . ...Load first subfile page
--- . PGM.*Reload subfile = CND.*NO
--- . > Conduct screen conversation
--- . .=REPEAT WHILE
--- . .-PGM.*Reload subfile is *NO
--- . .Display screen
--- . . ...Process response <--
--- . .-ENDWHILE
--- . .-ENDWHILE
--- . ...Closedown <--
---
---

ACTION DIAGRAM EXIT POINTS      F3=EXIT      SEL:X,Z-Select.
█ USER: Initialize program
--- USER: Initialize subfile header <--
--- USER: Initialize subfile record(existing record)
--- USER: Initialize subfile record(new record)
--- .-CALC: Subfile control function fields <--
--- .-PGM: Validate subfile control
--- .-PGM: Validate subfile record fields
--- .-CALC: Subfile record function fields
--- .-X USER: Validate subfile record relations
---
---
--- . ...Process response <--
--- .-ENDWHILE
--- .-ENDWHILE
--- . ...Closedown <--
---
---

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

Press Enter.

Adding a New CASE Construct

Insert a condition in the action diagram that defines when the Display Racing results function is to be called. You can insert the new condition after the validation conditions you added in the previous topic.

## Inserting a Condition

Type **ICF** to insert a new condition.

```

EDIT ACTION DIAGRAM          Edit    MYMDL    Horse
FIND=>                        Edit Horse

___ > USER: Validate subfile record relations
___ .--
___ .-CASE                      <<<
___ .-RCD.Dam Date of birth GE RCD.Date of birth <<<
___ .-Send error message - 'Dam younger than horse' <<<
___ .-ENDCASE                  <<<
___ .-CASE                      <<<
___ .-RCD.Sire Date of birth GE RCD.Date of birth <<<
___ .-Send error message - 'Sire younger than horse' <<<
ICF .-ENDCASE                  <<<
___ .--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter to display the Edit Action - Condition window.

## Entering the Condition

In this step you will add the \*Zoom#1 condition for the Subfile selector field, \*SFLSEL, to indicate when the racing results for a given horse should be displayed. The \*SFLSEL field is automatically provided on function types that have subfile record formats unless you explicitly suppress it using the function options.

The \*Zoom#1 condition will automatically be added to the Selector Choice menu on the action bar of the Edit Horse device design as an action because it is used here in the Edit Horse action diagram.

## Entering the Condition Details

The \*SFLSEL field is in the subfile record; as a result, its context is RCD. You will specify the context and the field name and leave the ? in the Condition field to display a list of conditions available for the field.

Type the context **RCD**, field name **\*SFLSEL**, and **?** for the condition.



EDIT ACTION DIAGRAM  
FIND=>

EditMYMDLHorse  
Edit Horse

> USER: Validate  
-CASE  
-RCD.Dam Date  
Send error me  
-ENDCASE  
-CASE  
-RCD.Sire Date  
Send error me  
ICF  
-ENDCASE

EDIT ACTION - CONDITION  
Title. : !!! New condition  
Context.Field . . . : RCD \*SFLSEL  
Condition . . . . . : ?  
OR  
Comparison. . . . . :  
Context.Field . . . :  
F3=Exit F7=Edit Compound Condition

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward  
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

Press Enter.

Selecting the Zoom#1 Condition

The Edit Field Conditions panel shows the existing conditions for the \*SFLSEL field. You may select an existing condition or create a new one.

Type **X** against the \*Zoom#1 condition.

EDIT FIELD CONDITIONS

My model

Field name. . . . . : \*SFLSELAttr. : STS  
Enter condition . . . :and type to add new condition.  
type . . . : (Type: LST, VAL)

? Condition	Type	Op	File/From value	Display/To value	MN
- *ALL values	LST	**			
- *Delete	LST				
- *Select	LST				
- *Selection character	LST				
- *Zoom	LST				
- *Delete#1	VAL	4		4	
- *Delete#2	VAL	4		4	
- *Select#1	VAL	1		1	
- *Select#2	VAL	1		1	
- *Selection char value	VAL	/		/	
- *Selection char value #2	VAL	/		/	
X *Zoom#1	VAL	5		5	
*Zoom#2	VAL	5		5	

SEL: Z-Details, D-Delete, X-Select, U-Where used, N-Narrative.  
F3=Exit

Press Enter to return to the Edit Action - Condition and note that Zoom#1 has been inserted in the Condition field. Press Enter again to return to the Action Diagram Editor.

## Specifying a Function as the Action

You have now finished defining the condition. Next you will specify a function to be called as the action when the condition is true.

Type **IAF**.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                   Edit Horse

___ > USER: Validate subfile record relations
___ .--
___ .-CASE
___ .-RCD.Dam Date of birth GE RCD.Date of birth
___ .-Send error message - 'Dam younger than horse'
___ .-ENDCASE
___ .-CASE
___ .-RCD.Sire Date of birth GE RCD.Date of birth
___ .-Send error message - 'Sire younger than horse'
___ .-ENDCASE
___ .-CASE
IAF .-RCD.*SFLSEL is *Zoom#1
___ .-ENDCASE
___ .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

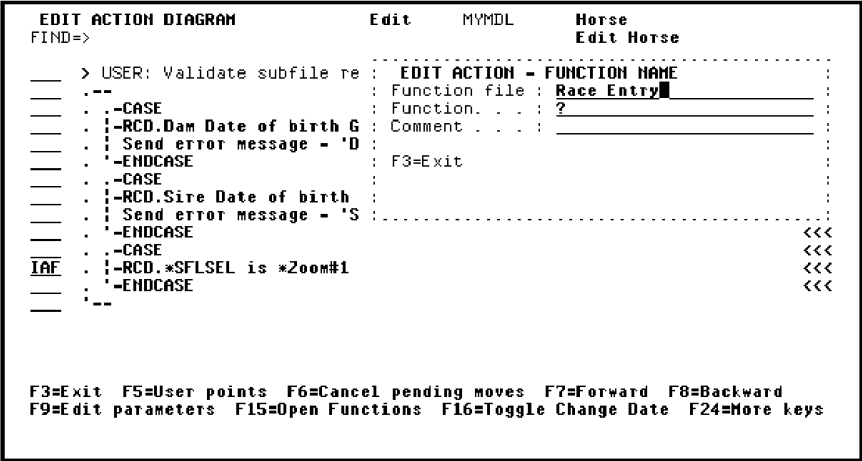
```

Press Enter to display the Edit Action - Function Name window.

## Naming the Function

To specify a reference to a function, type the name of the file to which the function is attached and the function's name. If you enter a file name and leave the **?** for the Function name field, CA 2E will display a list of functions that already exist for the file. You can select one or define a new one. In this case, the function to display race results does not yet exist. Note that the new function will be attached to the RACE ENTRY file, not the HORSE file.

Type **Race Entry** and leave the **?**.



Press Enter to go to the Edit Functions panel for the RACE ENTRY file.

Selecting a Function

The Edit Functions panel shows all the existing functions for the RACE ENTRY file. Note that if this is the first time you are viewing functions for RACE ENTRY, the default functions (Change Race Entry, Create Race Entry, and Delete Race Entry) are created before the Edit Functions panel displays.

Creating the New Function and Access Path

You are going to define a new function by typing the name of the new function, Display Racing results, and the function type, DSPFIL (Display File). You also need to specify an access path for the function that will display racing results in horse order.

Type **Display Racing results** for the Function name, **DSPFIL** for the Function type, and **?** to display the existing access paths.



EDIT FILE DETAILS

My model

File name : Race Entry  
Attribute : CPT  
Documentation sequence :  
GEN format prefix : AD  
Assimilated physical :  
Record not found message : Race Entry  
Record exists message : Race Entry

Field reference file : \*NONE  
Source library : MYGEN  
Distributed : N (Y,N)  
HF Msgid : USR0005  
EX Msgid : USR0006

? Typ Access path

PHY Physical file

UPD Update index

RTV Retrieval index

**Z RSQ Races for a Horse**

Source mbr

Key

Index options

Auto add

MYADCPP

NONE

ATR ONLY

MYADCPLB

UNIQUE

IMMED

ATR ONLY

MYADCPL1

UNIQUE

IMMED

ATR ONLY

SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock  
H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., M-Narrative  
F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative

Press Enter.

The Edit Access Path Details panel is displayed. Type **Z** against the format.

EDIT ACCESS PATH DETAILS

My model

File name : Race Entry  
Access path name : Races for a Horse  
Unique or duplicate order : F (U-Unique,F-FIFO,L-LIFO,C-FCFO,''-Undefined)  
Index maintenance option : I (I-IMMED, D-DLY, R-REBLD)  
Alternate collating table :  
Allow select/omit : (S-Static, D-Dynamic, ''-None)  
Generation mode : M (M-MDLVAL, D-DDS, S-SQL, X-UNX)  
Source member name : MYADCPL2  
Source member text : Race Entry

Attribute : CPT  
Type : RSQ

Format

GEN

Format text

Associated

? Seq name

pfx

(Based on file)

Retrieval access path

**Z 1**

**ADCPA2**

**AD**

**Race Entry**

**Retrieval index**

SEL: Z-Entries, R-Relations, S-Select/omit, A-Assoc.acps, T-Trim, V-Virtualize  
F3=Exit F8=Rename F20=Narrative

Press Enter.

Specifying the Access Path Details

On the Edit Access Path Format Entries panel you will specify the new key order for the RSQ access path. Initially, the keys default to be the same as for the RTV type access path. Remember, you cannot change the keys for a RTV access path.

## Defining the Access Path Key

The objective for the resulting application is to give the end user the capability to view race results for a selected horse. The recommended order by which to retrieve the RACE ENTRY file is by Horse code, Race date, and Race time, with the most recent races retrieved first. Specify a key of Horse code followed by Race date (descending order) and Race time (descending order). Before you change the order, clear the numbers from the original key order.

Type the details as shown. In other words, blank the Key no. field for Course code and Entry number; type **2**, **3**, **1** in the Key no. field for Race date, Race time, and Horse code, respectively; and type **D** (descending order) in the Dsc column for Race date and Race time.

EDIT ACCESS PATH FORMAT ENTRIES				My model			
File name . . . . .	:	Race Entry		Attribute . . .	:	CPT	
Access path name. . . . .	:	Races for a Horse		Type. . . . .	:	RSQ	
Format text . . . . .	:	Race Entry					
Based on. . . . .	:	Race Entry		Format No . . .	:	1	
				GEN	Key	Altcol	Ref
? Field			Type	no.	Dsc	seq	cnt
- Course code	CDE	ABCD	K				1
- Race date	DT#	ABDZ	K	2	D		1
- Race time	TM#	ABT2	K	3	D		1
- Entry number	CDE	ACCD	K				1
- Horse code	CDE	ADCD	A	1			1
- Jockey code	CDE	AECD	A				1
- Finishing position	NBR	ABNB	A				1
- Handicap	QTY	ABQT	A				1
- Entry Status	STS	ACST	A				1
SEL: Z-Field details, L-Locks.							
F3=Exit F7=Relations							

Press Enter to confirm the keys. Press F13 to return to the Edit File Details panel.

## Selecting the Access Path

Now select the new access path for use in the Display Racing results function. Type **X** to select the Races for a Horse RSQ access path.

EDIT FILE DETAILS

File name . . . . . : Race Entry

Attribute . . . . . : CPT

Documentation sequence . . . :

GEN format prefix . . . . . : AD

Assimilated physical . . . . :

Record not found message . . : Race Entry

Record exists message . . . : Race Entry

My model

Field reference file . . . : \*NONE

Source library . . . . . : MYGEN

Distributed . . . . . : N (Y,N)

NF Msgid . . : USR0005

EX Msgid . . : USR0006

? Typ Access path

☒ PHY Physical file

MYADCPP

NONE

ATR ONLY

☐ UPD Update index

MYADCPL0

UNIQUE

IMMED

ATR ONLY

☐ RTV Retrieval index

MYADCPL1

UNIQUE

IMMED

ATR ONLY

☒ RSQ Races for a Horse

MYADCPL2

FIFO

IMMED

ATR ONLY

☐

☐

☐

☐

SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock

H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative

F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative

Function 'Display Racing results' is being created ...

Press Enter. The message “Function ‘Display Racing results’ is being created” appears at the bottom of the panel. When the function is created, CA 2E returns to the Edit Functions panel, which now includes the Display Racing results function.

## Function Parameters

*Function parameters* specify which fields can be passed between the calling and the called functions. Each call can pass different values in these fields.

New terms introduced:

- Function parameter
- Parameter usage
- Parameter role
- Restrictor parameter

New panels introduced:

- Edit Function Parameter Details
- Work with Choices
- Work with Actions of a Choice
- Edit Action
- Work with Actions of a Choice

## Objectives

In this topic you will specify the parameters that will be passed from the Edit Horse function to the Display Racing results function in order to view races for a particular horse.

## Understanding Parameter Usage and Role

You need to assign a usage and role for each parameter to direct the calling and called functions to use the parameter in a particular way.

- **Parameter Usage**—A parameter's usage determines how the parameter is to be received from or returned to the calling function. The four possible usages are Input only, Output only, Both, and Neither. In this tutorial you will use only the first two usages:
  - **Input only**—A value for the parameter is passed to the called function. The called function returns the value to the calling function without changing it.
  - **Output only**—The called function returns a value for the parameter to the calling function when the called function ends.
- **Parameter Role**—The role of a function parameter specifies how the parameter will be used in the called function. The four possible roles are Map, Restrictor, Positioner, and Vary. In this tutorial you will use only the Restrictor role.

The Restrictor role is used to restrict which records from a database file will be displayed, changed, or printed by the called function. A Restrictor parameter must be a key field on the access path to which the called function attaches. If there are multiple keys, the parameter sequence must match the key sequence.

You will specify the Restrictor role for the Horse code parameter to restrict the Display Racing results function to show only race entries for a particular horse. In other words, the function can only process database records whose keys match the Horse code parameter.

## Specifying Function Parameters

In this step you will specify the parameters that will be passed to the Display Racing results function.

Type **P** to edit the parameters.



Press Enter to display the Edit Function Parameters panel.

## Defining Function Parameters

- You can specify individual fields as parameters by typing the keyword **\*FIELD** under File/\*FIELD and typing the field name under Access path/Field. In the Passed as column, type **FLD** to indicate that each field is passed as an individual parameter.
- You can specify a set of fields from a file by typing a file name under File/\*FIELD and the name of an access path based on the file. In the Passed as column, type one of the following.
  - **RCD** to indicate that all the fields from the access path are to be passed as a single parameter.

- **KEY** to indicate that only the key fields from the access path are to be passed as a single parameter.

In this case, you can zoom to display the Edit Function Parameter Details panel to select fields from the specified access path.

You used the \*FIELD method to specify parameters earlier in the tutorial when you defined error message functions for the Edit Horse function. In this topic you will use the access path method.

## Specifying Parameters Using an Access Path

Type the details as shown: type **Race Entry** as the file name, type **Races for a Horse** as the access path, type **KEY** as the Passed as value, and type **Z** in the Subfile selector to select the key fields on the access path to be used as parameters.

EDIT FUNCTION PARAMETERS				My model	
Function name: : Display Racing results			Type : Display file		
Received by file : Race Entry			Acpth: Races for a Horse		
?	File/*FIELD	Access path/Field	Passed as	Seq	
Z	Race Entry	Races for a Horse	KEY		
-					
-					
-					
-					
-					
-					
-					
-					
-					

Values

FLD: One parameter per field  
 RCD: One parameter for all fields  
 KEY: One parameter for key fields only

SEL: Z-Details (field selection).  
 F3=Exit F5=Reload

Press Enter to display the Edit Function Parameter Details panel.

## Defining Parameter Details

From the Edit Function Parameter Details panel, you can define parameter details.

The Horse code from the selected subfile record is to be passed as an input parameter to the Display Racing results function and will be restricted to show only race entries for this horse. This is where you specify that Horse code is an input only, restrictor parameter.

Type **R** against Horse code.

```
EDIT FUNCTION PARAMETER DETAILS      My model
Function name. . : Display Racing results  Type : Display file
Received by file : Race Entry            Acpth: Races for a Horse
Parameter (file) : Race Entry            Passed as: KEY
? Field                               Usage  Role  Flag error
R Horse code                          I      RST
■ Race date
- Race time

SEL: Usage: I-Input, O-Output, B-Both, N-Neither, D-Drop.
      Role: R-Restrict, M-Map, V-Vary length, P-Position. Error: E-Flag Error.
F3=Exit
```

Press Enter.

Note that the Usage for Horse code defaulted to I (Input only) and the parameter role changed to RST (restrictor).

```
EDIT FUNCTION PARAMETER DETAILS      My model
Function name. . : Display Racing results  Type : Display file
Received by file : Race Entry            Acpth: Races for a Horse
Parameter (file) : Race Entry            Passed as: KEY
? Field                               Usage  Role  Flag error
■ Horse code                          I      RST
- Race date
- Race time

SEL: Usage: I-Input, O-Output, B-Both, N-Neither, D-Drop.
      Role: R-Restrict, M-Map, V-Vary length, P-Position. Error: E-Flag Error.
F3=Exit
```

Completing the Parameter Details

You have now finished specifying the parameter details. Fast exit to the Edit Functions panel by pressing F13.

Selecting the New Function

The Display Racing results function is now defined. Remember that you were defining an action in the Edit Horse action diagram. Select the function you have just created as the action you are defining.

Type **X** to select the Display Racing results function.

[illegible]

Press Enter to return to the Action Diagram Editor. Note that the name of the Display Racing results function has been inserted in the Edit Action - Function Name window.

```

EDIT ACTION DIAGRAM                                Edit      MYMDL      Horse
FIND=>                                              Edit Horse
-----
> USER: Validate subfile re:
-----
- CASE
- RCD.Dam Date of birth G
- Send error message - 'D
- ENDCASE
- CASE
- RCD.Sire Date of birth
- Send error message - 'S
- ENDCASE
- CASE
- RCD.*SFLSEL is *Zoom#1
- ENDCASE
-----
IAF
-----

```

```

EDIT ACTION - FUNCTION NAME
Function file : Race Entry
Function. . . : Display Racing results
Comment_ _ _ :
F3=Exit

```

```

-----
<<<
<<<
<<<
<<<
<<<

```

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward  
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

Press Enter to display the Edit Action - Function Details window.

## Returning to the Edit Action - Function Details Window

This window confirms that you have selected Display Racing results as the function for your action. CA 2E has defaulted the value of Horse code from the subfile record to pass as the parameter to the Display Racing results function. Note the RCD context.

```
EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                   Edit Horse
-----
> USER: Validate subfile re : EDIT ACTION - FUNCTION NAME
-----
EDIT ACTION - FUNCTION DETAILS
Function file : Race Entry
Function. . . : Display Racing results
-----
IOB Parameter      Use Typ Ctx Object Name
I Horse code      RST FLD 3CD Horse code
-----
IAF
-----
F3=Exit F9=Edit parameters F10=Default parms F12=Previous
Some parameters have been defaulted. Press ENTER to accept
-----
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys
```

You can override these defaults; however, in this case, you will accept the defaults. Press Enter to accept the defaults.

The action diagram shows that the Display Racing results function is called when the end user selects the Zoom#1 option.

Suppressing the Confirm Prompt and File Update

In this step you will insert an action in the Edit Horse function to suppress both the confirm prompt and the update of the HORSE file when the user zooms to display racing results. As a result, the Edit Horse panel will remain open for input on return from the Display Racing results function. This step is not strictly necessary, but will make the function easier to use.

CA 2E supplies a set of system fields that control the execution of functions. \*Defer confirm is the system field that controls whether the confirm prompt is displayed or suppressed. \*Defer confirm is a status field (STS) that has two condition values: Proceed to confirm (the default) and Defer confirm. Note that the name of the field and the name of one of its values are the same, namely, Defer confirm.

The action you will insert will override the default for the \*Defer confirm field using the \*MOVE built-in function and the CND context.

- The \*MOVE built-in function lets you move the value of one field to another field. It has two parameters: an input parameter, which is the field to be moved, and an output parameter, which is the field into which the input field is moved.
- The CND context lets you specify a condition value and gives you the capability of specifying a condition value as a field value or a function parameter.

You will use the \*MOVE function to move the condition value *Defer confirm* (CND context) to the system field \**Defer confirm*, thus overriding the default value of the \**Defer confirm* system field.

## Inserting \*MOVE as an Action

CA 2E provides the I= command as a shortcut for inserting a \*MOVE built-in function as a new action. The I= command performs three steps in one: IA (inserts an action), F (edits action details), and specifies the \*MOVE built-in function as the action.

Type **I=** next to the function to insert a \*MOVE built-in function.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      Horse
FIND=>                   Edit Horse

___ > USER: Validate subfile record relations
___ .--
___ .-CASE                                     <<<
___ .-RCD.Dam Date of birth GE RCD.Date of birth <<<
___ .-Send error message - 'Dam younger than horse' <<<
___ .-ENDCASE                                   <<<
___ .-CASE                                     <<<
___ .-RCD.Sire Date of birth GE RCD.Date of birth <<<
___ .-Send error message - 'Sire younger than horse' <<<
___ .-ENDCASE                                   <<<
___ .-CASE                                     <<<
___ .-RCD.*SFLSEL is *Zoom#1                   <<<
___ I= Display Racing results - Race Entry *    <<<
___ .-ENDCASE                                   <<<
___ .--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Specifying Parameters for the \*MOVE Function

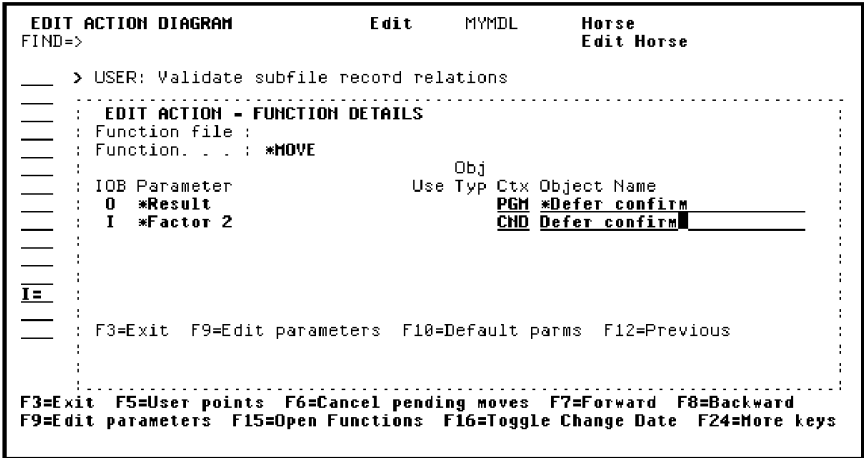
From the Edit Action - Function Details window, you can specify the parameters for the \*MOVE function.

The \*MOVE built-in function moves the value of one field to another field and thus has two parameters. On the Edit Action - Function Details window the output parameter is indicated by \**Result* and the input parameter is indicated by \**Factor 2*.

Recall that you can type **?** in any of the fields on this panel to display a selection list of possible values for the field.

Specifying Defer Confirm

Type parameters for the \*MOVE built-in function. In this case, you will type the system field **\*Defer confirm** as the output field. Because it is a system field, the appropriate context is **PGM**. The input field is the Defer confirm condition. Type **CND** for the context. This lets you use a condition value as if it were a field value and gives you the capability of specifying a condition value as a function parameter.



Press Enter.

Your action diagram now shows the action that will suppress the confirm prompt.

Reload Subfile

To activate the action bar and call the Display Racing results program, the end user types / in the Subfile selector of the Edit Horse program. As the Edit Horse program is now designed, the / remains in the Subfile selector when control is returned to Edit Horse. To prevent this, you will add processing in the action diagram to refresh the Edit Horse panel on return from the Display Racing results function. To reload and redisplay this panel, you need to override the default value of the \*Reload subfile system field. \*Reload subfile is a status field with two conditions: \*NO and \*YES and has the PGM context.

To accomplish this task, use the method you just used to override the default value of the \*Defer confirm system field. In other words, you will insert another \*MOVE function to move a value of \*YES to the \*Reload subfile system field.

Type **I=** as shown to insert a \*MOVE built-in function as an action in the action diagram.

```

EDIT ACTION DIAGRAM                                Edit      MYMDL      Horse
FIND=>                                              Edit Horse

___ > USER: Validate subfile record relations
___ .--
___ .-CASE
___ .-RCD.Dam Date of birth GE RCD.Date of birth
___ . Send error message - 'Dam younger than horse'
___ .-ENDCASE
___ .-CASE
___ .-RCD.Sire Date of birth GE RCD.Date of birth
___ . Send error message - 'Sire younger than horse'
___ .-ENDCASE
___ .-CASE
___ .-RCD.*SFLSEL is *Zoom#1
___ . Display Racing results - Race Entry *
___ I= PGH.*Defer confirm = CND.Defer confirm
___ .-ENDCASE
___ .--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Specifying Reload Subfile

Type the details as shown. In other words, type **PGM** for the output parameter context, type **\*Reload subfile** for the parameter name, type **CND** for the input parameter context, and type **\*Yes** for the condition to be moved.

```

EDIT ACTION DIAGRAM          Edit      MYMDL        Horse
FIND=>                        Edit Horse

___ > USER: Validate subfile record relations
___ .....
___ : EDIT ACTION - FUNCTION DETAILS
___ : Function file :
___ : Function. . . : *MOVE
___ :
___ : IOB Parameter           Obj
___ :                      Use Type Ctx Object Name
___ : 0 *Result              PGM *Reload subfile
___ : 1 *Factor 2            CND *Yes_____
___ :
___ :
___ :
___ :
___ : F3=Exit   F9>Edit parameters   F10=Default parms   F12=Previous
___ :
___ :
___ :
___ :
F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9>Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter to return to the action diagram.

## The Completed Action Diagram

The action diagram now shows the action to reload the subfile.



EDIT ACTION DIAGRAMEditMYMDLHorseEdit Horse

FIND=>

> USER: Validate subfile record relations

-CASE

-RCD.Dam Date of birth GE RCD.Date of birth

Send error message - 'Dam younger than horse'

-ENDCASE

-CASE

-RCD.Sire Date of birth GE RCD.Date of birth

Send error message - 'Sire younger than horse'

-ENDCASE

-CASE

-RCD.\*SFLSEL is \*Zoom#1

Display Racing results - Race Entry \*

PGH.\*Defer confirm = CHD.Defer confirm

PGH.\*Reload subfile = CHD.\*YES

-ENDCASE

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward

F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

Press F13 to fast exit to the Exit Function Definition panel.

Saving the Action Diagram

You have now finished specifying the interface to call the Display Racing results function from the Edit Horse function.

EXIT FUNCTION DEFINITIONMy model

Type choices, press Enter.

Change/create function. . . . .Y

Function name . . . . .Edit HorseName

Access path name. . . . .Retrieval indexName

File name . . . . .HorseName

Function type . . . . .Edit file

Print function. . . . .N

Return to editing . . . . .N

Submit generation . . . . .N

F5=Refresh F12=Cancel F15=Open Functions

Accept the default of Y (Change/create function) by pressing Enter. You have now completed the update to the action diagram for Edit Horse.

Updating the Edit Horse Function's Action Bar

In this step you will change the text in the Selector Choice menu on the action bar at the top of the Edit Horse function device design to show that the Display Racing results option is available. You will use the Device Design Editor to do this.

Type **S** against the Edit Horse function as shown to access the Device Design Editor.

EDIT FUNCTIONS		
File name. . . : Horse		
My model		
** 1ST LEVEL **		
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
S Edit Horse	Edit file	Retrieval index
█ Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Harr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press Enter.

Updating the Edit Horse Panel's Zoom Action Text

CA 2E has already updated the Edit Horse panel and added the Zoom#1 action to the action bar on the Edit Horse panel. Change the text from Zoom#1 to Display Racing results to provide a better explanation to the end user.

Move the cursor onto the action bar.

File█ fFunction Selector Help								
*PROGRAM *PGMMOD			DD/MM/YY HH:MM:SS					
Edit Horse								
Horse code . _____								
Select items, then select an action.								
Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth			
-	Dam	_____	_____	Dam	Date of birth 66/66/66			
	Sire	_____	_____	Sire	Date of birth 66/66/66			
-	Dam	_____	_____	Dam	Date of birth 66/66/66			
	Sire	_____	_____	Sire	Date of birth 66/66/66			
-	Dam	_____	_____	Dam	Date of birth 66/66/66			
	Sire	_____	_____	Sire	Date of birth 66/66/66			
F3=Exit F5=Reset F9=Open F10=Actions								

Press Enter to access the Work with Choices panel.

Working with the Selector Choice

From the Work with Choices panel, you can access the Work with Actions panel for a particular choice. In this case, you will work with the actions listed in the Selector Choice menu.

Type **A** next to Selector.

WORK WITH CHOICES

My model

File Name . . . : Horse  
Function name . : Edit Horse

Opt	Sequence	Choice	Usage	Mnemonic	CUA Model
-	1	File		F	A
-	4	fUnction		U	A
A	5	Selector		S	A
█	99	Help		H	A

SEL: Z-Details D-Delete A-Actions H/S-Usage N-Narrative  
F3=Exit F6=Add A Choice F7=All Actions

Press Enter.

Modifying an Action

The Work with Actions of a Choice panel lists all the actions available for a particular choice. You will zoom into the Zoom#1 action to view the details for that action.

Type **Z** in the Opt field against the Zoom#1 action.

WORK WITH ACTIONS OF A CHOICE

My Model

File Name . . . : Horse  
Function name . : Edit Horse  
Choice name . . : Selector

Opt	Sequence	Action	Usage	CUA Model
Z	1	Zoom#1		A

SEL: Z-Details, H,S-Usage, N-Narrative  
F3=Exit

Press Enter.

## Changing the Action Text

Update the text for the Zoom#1 action to read "Display Racing results."

Type **Display Racing results** in the Action text field.

EDIT ACTION		My model	
File Name . . . .	Horse		
Function name . .	Edit Horse		
Action . . . . .	<b>Display Racing results</b>	Text	Msgid
Sequence . . . . .	<b>1</b>	1..99	<b>MAAAAAA</b>
Choice . . . . .	<b>Selector</b>	Choice text	
Usage. . . . .		H, blank	
CUA Model. . . . .	<b>A</b>	A, T, G	
CUA Text :			
Accelerator. . . .		*CMDkey val cnd	
CUA Graphic :			
Accelerator. . . .		*PWSkey val cnd	<b>MAAAAC</b>
Mnemonic . . . . .	<b>Z</b>	Character	<b>MAAAAB</b>
Pushbutton sequence.	<b>0</b>	0..99	
Pushbutton default .	<b>H</b>	Y, N	
<b>F3=Exit F20=Narrative</b>			

Press Enter to confirm. Press F3 to exit.

## Showing the New Action Text

The Work With Actions of a Choice panel now shows the new action text for the Selector Choice menu.

WORK WITH ACTIONS OF A CHOICE		My Model	
File Name . . . .	Horse		
Function name . .	Edit Horse		
Choice name . . .	Selector		
Opt	Sequence	Action	Usage CUA Model
<b>1</b>	<b>1</b>	<b>Display Racing results</b>	<b>A</b>
SEL: Z-Details, H,S-Usage, N-Narrative			
F3=Exit			

Press F3 to return to the Work With Choices panel.

WORK WITH CHOICES

My model

File Name . . . : Horse  
Function name . . : Edit Horse

Opt	Sequence	Choice	Usage	Mnemonic	CUA	Model
■	1	File		F		A
-	4	fUnction		U		A
-	5	Selector		S		A
-	99	Help		H		A

SEL: Z=Details D=Delete A=Actions H/S=Usage N=Narrative  
F3=Exit F6=Add A Choice F7=All Actions

Exiting the Action Bar Editor

Press F3 to exit the Work with Choices panel and return to the device design.

■ File fUnction Selector Help

\*PROGRAM \*PGMMOD

DD/MM/YY HH:MM:SS

Edit Horse

Horse code .

Select items, then select an action.

Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66
-	Dam	00000000000000000000000000000000	Dam	Date of birth	66/66/66
	Sire	00000000000000000000000000000000	Sire	Date of birth	66/66/66

F3=Exit F5=Reset F9=Open F10=Actions

Exiting the Modified Panel

Press F3 to exit the modified device design.

Saving the Modified Panel

After exiting the device design, CA 2E displays the Edit Function Devices panel.

**Note:** From this panel, you can use Subfile selector option **Z** to return to the device design or you can press F5 to edit the action diagram for the Edit Horse function.

EDIT FUNCTION DEVICES		My model
Function name. . .	<b>Edit Horse</b>	Type : <b>Edit file</b>
Received by file :	<b>Horse</b>	Acpth: <b>Retrieval index</b>
? Title		
Screen title.....	<b>Edit Horse</b>	
SEL: Z-Scr/rpt design, N-Narrative, A-Animate		
F3=Exit F5=Action diagram F15=Open Functions		

Press F3 to return to the Exit Function Definition panel.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . .	<b>Y</b>	Y=Yes, N=No
Function name . . . . .	<b>Edit Horse</b>	Name
Access path name. . . . .	<b>Retrieval index</b>	Name
File name . . . . .	<b>Horse</b>	Name
Function type . . . . .	<b>Edit file</b>	
Print function. . . . .	<b>N</b>	Y=Yes, N=No
Return to editing . . . . .	<b>N</b>	Y=Yes, N=No
Submit generation . . . . .	<b>N</b>	Y=Yes, N=No
F5=Refresh F12=Cancel F15=Open Functions		

Press Enter to save the device design and return to the Edit Functions panel.

## Edit Database Relations Panel

Press F3 to return to the Edit Database Relations panel.

```

EDIT DATABASE RELATIONS
=> Horse*
?  Typ Object
  FIL Horse
  FIL Horse
  FIL Horse
  FIL Horse
  FIL Horse
  FIL Horse
  FIL Horse
  For: Dam
  FIL Horse
    For: Sire

My model
Rel lvl:
Relation      Seq Typ Referenced object
known by    10 FLD Horse code
Has        20 FLD Horse name
Has        30 FLD Horse gender
Has        40 FLD Horse value
Has        50 FLD Date of birth
Refers to  60 FIL Horse
                        Sharing: *ALL
Refers to  70 FIL Horse
                        Sharing: *ALL

Bottom
Z(n)=Details  F=Functions  E(n)=Entries  S(n)=Select  F23=More options
F3=Exit       F5=Reload    F6=Hide/Show  F7=Fields      F9=Add/Change F24=More keys

```

## Exercise

Check the device design for the Display Racing results function to improve its appearance and to ensure that the device design does not exceed the size of the panel. The latter prevents errors when you generate the functions in the *Generating, Compiling, and Executing* chapter.

**Note:** You can access the device design for a function by typing **S** against it on the Edit Functions panel or by pressing F9 from the Edit Function Details panel.

To edit the device design, use the function keys you used to edit the device designs for the Edit Horse and Select Horse functions in the Device Designs topic earlier in this chapter. Press Help or refer to the table in the Device Designs topic for a list of function keys.





# Chapter 5: Generating, Compiling, and Executing

---

This chapter introduces the following topics:

- Implementing Access Paths and Functions
- Executing and Testing Compiled Programs

## Implementing Access Paths and Functions

This topic discusses how to generate source to implement the access paths and functions you have created. You will then compile the source to produce executable i OS objects: files and programs.

New terms introduced:

- Job list
- Creation

New displays introduced:

- Display Services Menu
- Submit Model Creates
- Submit Model Generations & Creates
- Convert Condition Values

## Objectives

Submit the batch generation of all the access paths and functions you have designed. After generating the source code, each access path and function will be compiled. You will then execute the Edit Horse program.

## Overview of Implementation

Once your application design is complete, you are ready to generate the source members for the database files, programs, display files, and help text that will implement the access paths and functions. Once generated, the source will need compiling.

You can generate source either interactively or in batch. CA 2E automatically keeps a list of the members to be generated and compiled in batch or to be generated interactively and not yet compiled. In this tutorial, you will generate source code and compile executable objects for all of your access paths and functions as one batch job. To do this, you will use options available on the Display Services Menu.

## Source Generation and Compilation of Access Paths and Functions

You must generate and compile the source for your access paths and functions before you can test your application programs. The first step is to indicate which access paths and functions you want generated and compiled. The Display Services Menu includes options to facilitate this process.

From the Edit Database Relations panel, press F17 to go to the Display Services Menu.

EDIT DATABASE RELATIONS				My model		
=>			Rel lvl:			
? Typ	Object	Relation	Seq	Typ	Referenced object	
<input checked="" type="checkbox"/> FIL	Course	Known by		FLD	Course code	
<input type="checkbox"/> FIL	Course	Has		FLD	Course name	
<input type="checkbox"/> FIL	Horse	Known by	10	FLD	Horse code	
<input type="checkbox"/> FIL	Horse	Has	20	FLD	Horse name	
<input type="checkbox"/> FIL	Horse	Has	30	FLD	Horse gender	
<input type="checkbox"/> FIL	Horse	Has	40	FLD	Horse value	
<input type="checkbox"/> FIL	Horse	Has	50	FLD	Date of birth	
<input type="checkbox"/> FIL	Horse	Refers to	60	FIL	Horse	
	For: Dam			Sharing:	*ALL	
<input type="checkbox"/> FIL	Horse	Refers to	70	FIL	Horse	
	For: Sire			Sharing:	*ALL	
<input type="checkbox"/> FIL	Jockey	Known by		FLD	Jockey code	
<input type="checkbox"/> FIL	Jockey	Has		FLD	Jockey name	
<input type="checkbox"/> FIL	Jockey	Has		FLD	Jockey gender	
<input type="checkbox"/> FIL	Race	Owned by		FIL	Course	
More...						
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options						
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys						

## Display all Access Paths for Selection

The Display Services Menu contains an option to display a list of all access paths designed in the design model.

Select the Display all access paths option.

DISPLAY SERVICES MENU		My model
<b>Generation</b>	1. Submit model create request (YSBMDLCRT)	
	2. Convert model data menu	
	3. Job list menu	
<b>Documentation</b>	6. Documentation menu	
	7. Convert model panel designs (YCVTMDLPNL)	
<b>Model</b>	8. Display all access paths	
	9. Display all functions	
	10. Display model values (YDSPMDLVAL)	
	11. Edit model profile (YEDTMDLPRF)	
	12. Work with model lists (YWRKMDLLST)	
	13. Edit model list (YEDTMDLLST *SESSION)	
	14. Impact analysis menu	
<b>Change Control</b>	21. Go to 2ECM menu	
Option: <b>8</b> (press F4 to prompt commands)		
F3=Exit F6=Messages F8=Submitted jobs F9=Command line F10=Display job log		

Press Enter.

## Selecting All Access Paths for Batch Generation and Compilation

The Display All Access Paths panel presents a list of all access paths in the design model. It lets you perform many design, control, and generation tasks for one or more access paths from one panel.

Select all of the existing access path designs for batch generation and compilation by typing a **J** in the Subfile selector next to each access path in the list.

DISPLAY ALL ACCESS PATHS		My model	
Application area. : <b>1</b>		Source library. : <b>MYGEN</b>	
? File	Access path		Prefix
<b>J</b> Course	Physical file	PHY MYABREP	AB
<b>J</b> Course	Retrieval index	RTV MYABREL1	AB
<b>J</b> Course	Update index	UPD MYABREL0	AB
<b>J</b> Horse	Mares	RTV MYAEREL2	AE
<b>J</b> Horse	Physical file	PHY MYAEREP	AE
<b>J</b> Horse	Retrieval index	RTV MYAEREL1	AE
<b>J</b> Horse	Stallions	RTV MYAEREL3	AE
<b>J</b> Horse	Update index	UPD MYAEREL0	AE
<b>J</b> Jockey	Physical file	PHY MYAFREP	AF
<b>J</b> Jockey	Retrieval index	RTV MYAFREL1	AF
<b>J</b> Jockey	Update index	UPD MYAFREL0	AF
<b>J</b> Race	Physical file	PHY MYACREP	AC
<b>J</b> Race	Retrieval index	RTV MYACREL1	AC
<b>J</b> Race	Update index	UPD MYACREL0	AC
<b>J</b> Race Entry	Physical file	PHY MYADCPP	AD +
SEL: Z-Details, G/J-Generate, E-STRSEU, H-Narrative, D-Delete, U-Usage, F-Function refs., L-Locks.			
F3=Exit F5=Reload			

Press Roll Up to display the next screen of access paths. Select all remaining access paths with a **J**.

DISPLAY ALL ACCESS PATHS		My model	
Application area. . . :	<input type="checkbox"/>	Source library. . . :	MYGEN
? File	Access path	Prefix	
J Race Entry	Races for a Horse	RSQ MYADCPL2	AD
J Race Entry	Retrieval index	RTV MYADCPL1	AD
J Race Entry	Update index	UPD MYADCPL0	AD

SEL: Z-Details, G/J-Generate, E-STRSEU, N-Narrative,  
D-Delete, U-Usage, F-Function refs., L-Locks.  
F3=Exit F5=Reload

Press Enter.

## Completing the Request

When the process is complete, the Display All Access Paths panel will be redisplayed with messages displayed at the bottom of the panel. The messages will state that the source generation requests have been accepted.

DISPLAY ALL ACCESS PATHS		My model	
Application area. . . :	<input type="checkbox"/>	Source library. . . :	MYGEN
? File	Access path	Prefix	
- Race Entry	Races for a Horse	RSQ MYADCPL2	AD
- Race Entry	Retrieval index	RTV MYADCPL1	AD
- Race Entry	Update index	UPD MYADCPL0	AD

SEL: Z-Details, G/J-Generate, E-STRSEU, N-Narrative,  
D-Delete, U-Usage, F-Function refs., L-Locks.  
F3=Exit F5=Reload  
Source generation request for MYABREP accepted. +

Press F3 to return to the Display Services Menu.

## Display all Functions for Selection

In this step you will repeat this process for the functions in your design model. The Display Services Menu includes an option to display a list of the functions in the design model.

Select the Display all functions option.

DISPLAY SERVICES MENU		My model
<b>Generation</b>	1. Submit model create request (YSEBMDLCRT)	
	2. Convert model data menu	
	3. Job list menu	
<b>Documentation</b>	6. Documentation menu	
	7. Convert model panel designs (YCVTMDLPNL)	
<b>Model</b>	8. Display all access paths	
	9. Display all functions	
	10. Display model values (YDSPMDLVAL)	
	11. Edit model profile (YEDTMDLPRF)	
	12. Work with model lists (VWRKMDLLST)	
	13. Edit model list (YEDTMDLLST *SESSION)	
	14. Impact analysis menu	
<b>Change Control</b>	21. Go to 2ECM menu	
Option: <b>9</b> (press F4 to prompt commands)		
F3=Exit F6=Messages F8=Submitted jobs F9=Command line F10=Display job log		

Press Enter.

## Selecting All External Functions for Batch Generation and Compilation

The Display All Functions panel displays a list of all functions in the design model. It lets you perform many design, control, and generation tasks for one or more functions from one display.

Only external functions need to be generated and compiled. External functions can be identified by the source member name found in the GEN name column; internal functions show \*N/A in this column. Start by displaying just the external functions in your design model.

To do so, type **\*EXT** in the Type column on the selection line.

DISPLAY ALL FUNCTIONS		My model	Source library: MYGEN
Application area. : ____	Function	Type	GEN name
? File		*EXT	
- Course	Change Course	CHGOBJ	*N/A
- Course	Create Course	CRTOBJ	*N/A
- Course	Delete Course	DLTOBJ	*N/A
- Course	Edit Course	EDTFIL	HYACEFR
- Course	Select Course	SELRCO	HYABSRR
- Horse	Change Horse	CHGOBJ	*N/A
- Horse	Create Horse	CRTOBJ	*N/A
- Horse	Delete Horse	DLTOBJ	*N/A
- Horse	Edit Horse	EDTFIL	HYAEEFR
- Horse	Select Horse	SELRCO	HYAISRR
- Horse	Select Mares	SELRCO	HYAJSRR
- Horse	Select Stallions	SELRCO	HYAKSRR
- Jockey	Change Jockey	CHGOBJ	*N/A
- Jockey	Create Jockey	CRTOBJ	*N/A
- Jockey	Delete Jockey	DLTOBJ	*N/A
SEL: 2-Dtls, P-Parms, N-Harr., F-Action diagram, S-Device Design, T-Structure, A-Acp, G/J-Gen, E-STRSEU(pgm), L-Locks, D-Delete, U-Where used, 3-Doc.			
F3=Exit F5=Reload			

Press Enter to redisplay the Display All Functions panel showing only external functions.

**Note:** Write down the implementation names shown in the GEN name column for the following functions: Edit Course, Edit Horse, and Edit Jockey. You will need these later when you test the compiled programs.

Select all of the external functions for batch generation and compilation by typing a **J** in the Subfile selector next to each function on the list.

DISPLAY ALL FUNCTIONS		My model	Source library: MYGEN	
Application area. : <input type="checkbox"/>			Type	GEN name
? File	Function		*EXT	
<input checked="" type="checkbox"/> Course	Edit Course		EDTFIL	MYACEFR
<input checked="" type="checkbox"/> Course	Select Course		SELRCO	MYABSRR
<input checked="" type="checkbox"/> Horse	Edit Horse		EDTFIL	MYAEEFR
<input checked="" type="checkbox"/> Horse	Select Horse		SELRCO	MYAISRR
<input checked="" type="checkbox"/> Horse	Select Mares		SELRCO	MYAJSRR
<input checked="" type="checkbox"/> Horse	Select Stallions		SELRCO	MYAKSRR
<input checked="" type="checkbox"/> Jockey	Edit Jockey		EDTFIL	MYANEFR
<input checked="" type="checkbox"/> Jockey	Select Jockey		SELRCO	MYAMSRR
<input checked="" type="checkbox"/> Race	Edit Race		EDTFIL	MYAPEFR
<input checked="" type="checkbox"/> Race	Select Race		SELRCO	MYAOSRR
<input checked="" type="checkbox"/> Race Entry	Display Racing results		DSPFIL	MYALDFR

SEL: Z-Dtls, P-Parms, H-Harr., F-Action diagram, S-Device Design, T-Structure, A-Acp, G/J-Gen, E-STRSEU(pgm), L-Locks, D-Delete, U-Where used, 3-Doc.  
F3=Exit F5=Reload

Press Enter.

## Completing the Request

When the process is complete, the Display All Functions panel will be redisplayed with messages displayed at the bottom of the panel. These messages state that the source generation requests have been accepted.

DISPLAY ALL FUNCTIONS		My model	Source library: MYGEN	
Application area. : <input type="checkbox"/>			Type	GEN name
? File	Function		*EXT	
<input checked="" type="checkbox"/> Course	Edit Course		EDTFIL	MYACEFR
<input checked="" type="checkbox"/> Course	Select Course		SELRCO	MYABSRR
<input checked="" type="checkbox"/> Horse	Edit Horse		EDTFIL	MYAEEFR
<input checked="" type="checkbox"/> Horse	Select Horse		SELRCO	MYAISRR
<input checked="" type="checkbox"/> Horse	Select Mares		SELRCO	MYAJSRR
<input checked="" type="checkbox"/> Horse	Select Stallions		SELRCO	MYAKSRR
<input checked="" type="checkbox"/> Jockey	Edit Jockey		EDTFIL	MYANEFR
<input checked="" type="checkbox"/> Jockey	Select Jockey		SELRCO	MYAMSRR
<input checked="" type="checkbox"/> Race	Edit Race		EDTFIL	MYAPEFR
<input checked="" type="checkbox"/> Race	Select Race		SELRCO	MYAOSRR
<input checked="" type="checkbox"/> Race Entry	Display Racing results		DSPFIL	MYALDFR

SEL: Z-Dtls, P-Parms, H-Harr., F-Action diagram, S-Device Design, T-Structure, A-Acp, G/J-Gen, E-STRSEU(pgm), L-Locks, D-Delete, U-Where used, 3-Doc.  
F3=Exit F5=Reload  
Source generation request for MYACEFR accepted. +

Press F3 to return to the Display Services Menu.

## Submit Batch Generation and Creation

In this step you will use the Display Services Menu to Submit generations and compilations of all the access paths and external functions you have selected.

## Generating and Creating Objects

When you submit a request for generation/compilation, the CA 2E generator automatically does the following.

- The member names to be generated are placed in a job list. The same job list controls generation and compilation. You can review this job list during generation to monitor the process and edit the job list.
- Source is produced and placed in the appropriate source file in the generation library associated with your CA 2E design model.
- The generated source is compiled from the source file.
- CA 2E assigns a status to each member on the list. Once source is successfully generated for the members, CA 2E automatically submits a request to compile the generated source. If errors occur, CA 2E flags the specific members in error.

You can submit generations and compilations of all the items in the list from the Display Services Menu.

To submit your generation/compilation request, select the Submit model create request (YSBMMDLCRT) option.

DISPLAY SERVICES MENU		My model
<b>Generation</b>	1. Submit model create request (YSBMMDLCRT) 2. Convert model data menu 3. Job list menu	
<b>Documentation</b>	6. Documentation menu 7. Convert model panel designs (YCVTMDLPNL)	
<b>Model</b>	8. Display all access paths 9. Display all functions 10. Display model values (YDSPMDLVAL) 11. Edit model profile (YEDTMDLPRF) 12. Work with model lists (YWRKMDLLST) 13. Edit model list (YEDTMDLLST *SESSION) 14. Impact analysis menu	
<b>Change Control</b>	21. Go to 2ECM menu	
Option: <b>1</b> (press F4 to prompt commands)		
F3=Exit F6=Messages F8=Submitted jobs F9=Command line F10=Display job log		

Press Enter to execute the Submit Model Create Requests (YSBMMDLCRT) command.

**Note:** CA 2E supplies default parameter values for the YSBMMDLCRT command based on your model profile and options you specified when you created your model. You can override these defaults by pressing F4 instead of Enter to prompt the command.

## List of Objects to be Generated and Created

CA 2E displays a list of the source members to be generated and compiled. Each member has either GEN or CRT next to it to indicate whether the member has been submitted for generation or compilation. In this case, all the functions and access paths should show GEN. You can review the list by pressing Roll Up and Roll Down.

SUBMIT MODEL GENERATIONS & CREATES. My model						
						GENLIB: MYGEN
? Member	Type	Act	Status	Text		
■ MYABREP	PF	GEN		Course		Physical file
■ MYACREP	PF	GEN		Race		Physical file
■ MYADCPP	PF	GEN		Race Entry		Physical file
■ MYAEREP	PF	GEN		Horse		Physical file
■ MYAFREP	PF	GEN		Jockey		Physical file
■ MYABREL0	LF	GEN		Course		Update index
■ MYACREL0	LF	GEN		Race		Update index
■ MYADCLP0	LF	GEN		Race Entry		Update index
■ MYAEREL0	LF	GEN		Horse		Update index
■ MYAFREL0	LF	GEN		Jockey		Update index
■ MYABREL1	LF	GEN		Course		Retrieval index
■ MYACREL1	LF	GEN		Race		Retrieval index
■ MYADCLP1	LF	GEN		Race Entry		Retrieval index +
SEL: G-Rqs GEN, C-Rqs CRT, E-STRSEU, D-Drop, JOB(1-DSP, 4-HLD, 6-RLS, 9-CNL) F3=Exit F5=Reload F6=Msgs F8=Submitted jobs F9=Command line ENTER=Submit						

Press Roll Up until you see the bottom of the list.

SUBMIT MODEL GENERATIONS & CREATES. My model						
						GENLIB: MYGEN
? Member	Type	Act	Status	Text		
■ MYAPEFRH	PNL	GEN		Edit Race		Edit file
■ MYABSRR	RPG	GEN		Select Course		Select record
■ MYACEFR	RPG	GEN		Edit Course		Edit file
■ MYAIFRR	RPG	GEN		Edit Horse		Edit file
■ MYAISRR	RPG	GEN		Select Horse		Select record
■ MYAISRR	RPG	GEN		Select Hares		Select record
■ MYAKSRR	RPG	GEN		Select Stallions		Select record
■ MYALDFR	RPG	GEN		Display Racing results		Display file
■ MYANSRR	RPG	GEN		Select Jockey		Select record
■ MYANEFR	RPG	GEN		Edit Jockey		Edit file
■ MYAOSRR	RPG	GEN		Select Race		Select record
■ MYAPEFR	RPG	GEN		Edit Race		Edit file
SEL: G-Rqs GEN, C-Rqs CRT, E-STRSEU, D-Drop, JOB(1-DSP, 4-HLD, 6-RLS, 9-CNL) F3=Exit F5=Reload F6=Msgs F8=Submitted jobs F9=Command line ENTER=Submit						

Press Enter to submit the job list.



## Confirming the Job List

The panel will be redisplayed with the option to confirm at the bottom right-hand corner. Accept the default of Y.

```

SUBMIT MODEL GENERATIONS & CREATES.  My model

                                     GENLIB: MYGEN
? Member      Type Act Status  Text
- MYAPEFRH    PHL  GEN      Edit Race      Edit file
- MYABSRR     RPG  GEN      Select Course  Select record
- MYACEFR     RPG  GEN      Edit Course   Edit file
- MYAEFR      RPG  GEN      Edit Horse    Edit file
- MYAISRR     RPG  GEN      Select Horse   Select record
- MYAISRR     RPG  GEN      Select Mares   Select record
- MYAKSRR     RPG  GEN      Select Stallions Select record
- MYALDFR     RPG  GEN      Display Racing results Display file
- MYAMSRR     RPG  GEN      Select Jockey  Select record
- MYANEFR     RPG  GEN      Edit Jockey    Edit file
- MYAOSRR     RPG  GEN      Select Race    Select record
- MYAPEFR     RPG  GEN      Edit Race      Edit file

SEL: G-Rqs GEN, C-Rqs CRT, E-STRSEU, D-Drop, JOB(1-DSP, 4-HLD, 6-RLS, 9-CNL)
F3=Exit F5=Reload F6=Msgs F8=Submitted jobs F9=Command line ENTER=Submit
                                     CONFIRM:  Y (Y,N)

```

Press Enter.

## Successful Submit for Generation and Compilation

CA 2E submits the jobs to generate and/or compile members in this list. If you request batch generation, compilation requests are also submitted by default. Compilation creates the i OS objects once the source has been generated. As a result, compilation is also referred to as *creation*. All generations are carried out in a single job called YGENSRC; a separate compilation job is submitted for each source member.

After confirming the list of objects, you will see a series of messages at the bottom of the panel. These are "Job YGENSRC is being prepared," "Existing objects are being deleted," and finally "Joblist successfully processed." In batch processing, the generations/ compilations take place in the background. You may continue to specify new objects (functions, access paths) while this is happening.

SUBMIT MODEL GENERATIONS & CREATES. My model					
GENLIB: MYGEN					
? Member	Type	Act	Status	Text	
■ MYABREP	PF	GEN	*SBM	Course	Physical file
- MYACREP	PF	GEN	*SBM	Race	Physical file
- MYADCPP	PF	GEN	*SBM	Race Entry	Physical file
- MYAEREP	PF	GEN	*SBM	Horse	Physical file
- MYAFREP	PF	GEN	*SBM	Jockey	Physical file
- MYABREL0	LF	GEN	*SBM	Course	Update index
- MYACREL0	LF	GEN	*SBM	Race	Update index
- MYADCPLO	LF	GEN	*SBM	Race Entry	Update index
- MYAEREL0	LF	GEN	*SBM	Horse	Update index
- MYAFREL0	LF	GEN	*SBM	Jockey	Update index
- MYABREL1	LF	GEN	*SBM	Course	Retrieval index
- MYACREL1	LF	GEN	*SBM	Race	Retrieval index
- MYADCL1	LF	GEN	*SBM	Race Entry	Retrieval index +
SEL: G-Rqs GEN, C-Rqs CRT, E-STRSEU, D-Drop, JOB(1-DSP, 4-HLD, 6-RLS, 9-CNL) F3=Exit F5=Reload F6=Msgs F8=Submitted jobs F9=Command line ENTER=Submit Joblist successfully processed					

## Examining the Job List

Once submitted, each item in the list is initially shown with a status of \*SBM. As each item in the list is processed, the status in the list is updated as follows.

Status	Description
*GENSRC	The source member is being generated
*JOBQ	The source is on the job queue to be compiled
*ACTIVE	The source is being compiled

Press F5 to refresh the list and see the latest status.

Once the source members have been successfully compiled, they are removed from the list.

SUBMIT MODEL GENERATIONS & CREATES. My model						GENLIB: MYGEN	
? Member	Type	Act	Status	Text			
■ MYABREP	PF	GEN	*GENSRC	Course		Physical file	
- MYACREP	PF	GEN	*SBM	Race		Physical file	
- MYADCPP	PF	GEN	*SBM	Race Entry		Physical file	
- MYAERP	PF	GEN	*SBM	Horse		Physical file	
- MYAFREP	PF	GEN	*SBM	Jockey		Physical file	
- MYABREL0	LF	GEN	*SBM	Course		Update index	
- MYACREL0	LF	GEN	*SBM	Race		Update index	
- MYADCPLO	LF	GEN	*SBM	Race Entry		Update index	
- MYAEREL0	LF	GEN	*SBM	Horse		Update index	
- MYAFREL0	LF	GEN	*SBM	Jockey		Update index	
- MYABREL1	LF	GEN	*SBM	Course		Retrieval index	
- MYACREL1	LF	GEN	*SBM	Race		Retrieval index	
- MYADCP1	LF	GEN	*SBM	Race Entry		Retrieval index	+

SEL: G-Rqs GEN, C-Rqs CRT, E-STRSEU, D-Drop, JOB(1-DSP, 4-HLD, 6-RLS, 9-CNL)  
 F3=Exit F5=Reload F6=Msgs F8=Submitted jobs F9=Command line ENTER=Submit

When the compilations have been submitted, press F3 to return to the Display Services Menu. You can check the status of your submitted jobs from the Display Services Menu by pressing F8.

## Converting Condition Values to a Database File

Before calling your program, you must convert the values that are entered into status fields using the Convert Condition Values (YCVTCNDVAL) command. This command moves the values you defined for status fields from the model library to the condition values list database file in the generation library.

When you convert condition values to a database file, you give end users the capability to prompt for valid condition values when they use the application. Horse gender is an example of a status field. In this case, the end user will be able to prompt the application and display a list of valid values for Horse gender, namely, M and F.

You can run the YCVTCNDVAL command to convert the condition values to a database file using the Convert model data menu, which is an option on the Display Services Menu.

Select the Convert model data menu option.

DISPLAY SERVICES MENU		My model
<b>Generation</b>	1. Submit model create request (YSBMDLCRT) 2. Convert model data menu 3. Job list menu	
<b>Documentation</b>	6. Documentation menu 7. Convert model panel designs (YCVTMDLPNL)	
<b>Model</b>	8. Display all access paths 9. Display all functions 10. Display model values (YDSPMDLVAL) 11. Edit model profile (VEDTMDLPRF) 12. Work with model lists (YWRKMDLLST) 13. Edit model list (VEDTMDLLST *SESSION) 14. Impact analysis menu	
<b>Change Control</b>	21. Go to 2ECM menu	
Option: <b>2</b> (press F4 to prompt commands)		
F3=Exit F6=Messages F8=Submitted jobs F9=Command line F10=Display job log		

Press Enter.

Select the Convert condition values to database file option.

DISPLAY CONVERT MODEL DATA MENU		My model
	1. Convert model messages to database file. 2. Convert condition values to database file. 3. Convert distributed files to database file.	
Option: <b>2</b>		
F3=Exit F6=Messages F8=Submitted jobs F9=Command line		

Press Enter to prompt the YCVTCNDVAL command.

## YCVTCNDVAL Command Prompt

CA 2E supplies default values for the CA 2E design model library and the generated library. Accept the defaults.

Convert Condition Values (YCVTCNDVAL)		
Type choices, press Enter.		
Library for data model . . . . .	<u>*MDLLIB</u>	Name, *MDLLIB, *CURLIB
Library for generation . . . . .	<u>*GENLIB</u>	Name, *GENLIB, *CURLIB
<div style="text-align: right;"><b>Bottom</b></div> F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display F24=More keys		

Press Enter to execute the YCVTCNDVAL command.

## Confirming Conversion of Condition Values

Once the values allowed for the status fields in your model have been converted to the database file, you will be returned to the Display Convert Model Data Menu. A message appears at the bottom of the panel to indicate that the conversion is complete.

DISPLAY CONVERT MODEL DATA MENU	My model
1. Convert model messages to database file. 2. Convert condition values to database file. 3. Convert distributed files to database file.	
Option: █	
F3=Exit   F6=Messages   F8=Submitted jobs   F9=Command line Condition values from model MYMDL converted to library MYGEN	

Press F3 twice to return to the Edit Database Relations panel.

## Exiting the CA 2E Design Model

When all the objects have been created, your program for editing the HORSE file is complete.

Press F3 to exit your design model.

```

EDIT DATABASE RELATIONS
=>
? Typ Object
  FIL Course
  FIL Horse
  FIL Horse
    For: Dam
  FIL Horse
    For: Sire
  FIL Jockey
  FIL Race
  FIL Race
  FIL Race
  FIL Race Entry
  FIL Race Entry
  FIL Race Entry
  FIL Race Entry
  Rel lvl: FIL
  Relation
  known by
  known by
  Refers to
  Sharing: *ALL
  Refers to
  Sharing: *ALL
  known by
  Owned by
  known by
  known by
  Owned by
  known by
  Refers to
  Refers to
  Seq Typ Referenced object
  FLD Course code
  FLD Horse code
  FIL Horse
  FIL Horse
  FLD Jockey code
  FIL Course
  FLD Race date
  FLD Race time
  FIL Race
  FLD Entry number
  FIL Horse
  FIL Jockey
Bottom
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

```

## Re-synchronizing an CA 2E Design Model

The Exit Edit Relations window displays. This window allows three options:

- Exit without re-synchronizing
- Exit and resynchronize data model
- Return to editing

If you have added or changed any files, fields, or relations in the current session, a message will be displayed on this panel, "Model is not resynchronized." In this case, the Exit and resynchronize data model option will be the default. If you select this option, the message "Data model is being resynchronized" will appear at the bottom of the panel.

The re-synchronizing process causes all the relations to be resolved into entries. Only users of type \*DSNR can resynchronize a model.

**Note:** If you have previously exited your design model, your model may already be resynchronized. In this case, the Exit Edit Relations panel defaults to Exit without re-synchronizing.

Accept the default (option 1 or 2).

Press Enter to exit your design model.

This topic explains how to call your compiled applications (programs) and test them by entering data.

Once the compilations you have submitted have completed successfully you can test the Edit Horse and Edit Course programs. A message is sent to your message queue when each compilation finishes.

Call your Edit Horse program from any command entry line by using the i OS CALL command to invoke your program, for example, MYAEEFR, as shown below. From within your model, you can display a command entry line by pressing F9 from the Display Services menu.

**Note:** If you do not know the name of your Edit Horse program, you can obtain it from the Edit Function Details panel or the Display all functions option on the Display Services Menu.

When you call your program you need to specify a dummy \*Return code parameter. All CA 2E programs require a dummy parameter to be passed to the program (' ' or ''). This return code can be used to communicate between programs in more complex applications.

Type the program name that corresponds to your Edit Horse function and return code.

```

MAIN                      2E Main Menu
Level . : 1
System: 2EDV1
Select one of the following:

Design Model          1. Display Designer (*DSNR) menu
                      2. Display Programmer (*PGMR) menu
                      3. Display User (*USER) menu
                      8. Work with Model Object Lists
                      9. Change to work with another model

Commands            50. 2E commands in alphabetical order
                      51. Commands to set up or alter a model
                      52. Commands to copy a model
                      53. Commands to create an application
                      54. Commands to document a model
More...

Selection or command
==> call myaeefr ''

F3=Exit  F6=Messages  F9=Prev. request  F10=Cmd Entry  F14=Submitted jobs
Maximum capability to access model MYMDL is *DSLK.

```

Press Enter.

**Note:** To test complex programs, you can use the CA 2E Call a Program (Y2CALL) command. This command loads your model and determines the parameters required by an external function directly from details contained in the model. You can provide values for all input-capable fields and you can re-use these values for subsequent calls. You can also retrieve and display output parameters when the called program ends.

## The Edit Horse Function Panel

An empty panel will be displayed. The Edit File function operates in two alternate modes, one to add new records (New mode) and one to change existing records (Open mode). Use the F9 function key to toggle between the two modes.

Note that the program mode is Open. Since there are no existing records to update, you first need to change to New mode to add records to the file.



```

File  Selector  Help
-----
MYAHEFR  OPEN                                2/03/95 15:50:31
                                Edit Horse
Horse code .  █
Select items, then select an action.

F3=Exit  F5=Reset  F9=New  F10=Actions
No data to display.

```

Press F9 to switch to New mode to add data to the HORSE file.

## Adding Data to the HORSE File

An empty input panel is displayed. Note that the program mode is New. This is where you may add new data. The panel should have the design you created and data should be validated according to the rules you specified in the design. Test this out by entering invalid values. Also try pressing F4 for the Horse gender field to display a selection list of available values.

To add data to the database file, type horse details.

```

File  Selector  Help
-----
MYAEEFR  NEW                                10/04/95 15:47:14
                                Edit Horse
Select items, then select an action.

Opt  Horse  Horse name  Horse  Horse value  Date of
code  code  name        gender  value         birth
-    -    -          -    -    -
  BONFIR  Bonfire          M      5000  020188
  Dam _____  Dam Date of birth
  Sire _____  Sire Date of birth
-    -    -          -    -    -
  DOBBIN  Faithful Dobbin     F      3500  053186
  Dam _____  Dam Date of birth
  Sire _____  Sire Date of birth
-    -    -          -    -    -
  PEGASU  Pegasus                M      3000  062392
  Dam _____  Dam Date of birth
  Sire _____  Sire Date of birth
                                +

F3=Exit  F5=Reset  F9=Open  F10=Actions

```

Press Enter.

## Confirming Data Entries for the Horse File

You will be prompted with a Confirm prompt before the update is completed. As you saw in the function option topic of this tutorial, the confirm prompt is an optional feature of a function. You could omit it from the design if desired.

Earlier in the tutorial, in the Function Options topic, you set the initial value of the Confirm prompt to Y. Verify that the initial value of the Confirm prompt is Y.

```

File  Selector  Help
-----
MYAEEFR  NEW                                10/04/95 15:50:47

                                Edit Horse

Select items, then select an action.

Opt  Horse  Horse name      Horse  Horse value  Date of
code  code   name            gender  value         birth
-    -    -              -    -    -
-    BONFIR Bonfire          M      5000.00     20188
   Dam _____   Dam Date of birth
   Sire _____   Sire Date of birth

-    DOBBIN Faithful Dobbin  F      3500.00     53186
   Dam _____   Dam Date of birth
   Sire _____   Sire Date of birth

-    PEGASU Pegasus         M      3000.00     62392
   Dam _____   Dam Date of birth
   Sire _____   Sire Date of birth

F3=Exit  F5=Reset  F9=Open  F10=Actions

CONFIRM: Y (Y/N)
  
```

Press Enter.

Note that once added, the key fields cannot be changed except by deletion.

After the new data has been successfully added, another empty subfile page is displayed, ready for more entries.

```

File  Selector  Help
-----
MYAEEFR  NEW                                10/04/95 15:52:45

                                Edit Horse

Select items, then select an action.

Opt  Horse  Horse name      Horse  Horse value  Date of
code  code   name            gender  value         birth
■    _____   _____   -    _____   _____
   Dam _____   Dam Date of birth
   Sire _____   Sire Date of birth

-    _____   _____   -    _____   _____
   Dam _____   Dam Date of birth
   Sire _____   Sire Date of birth

-    _____   _____   -    _____   _____
   Dam _____   Dam Date of birth
   Sire _____   Sire Date of birth

F3=Exit  F5=Reset  F9=Open  F10=Actions

+
  
```

## Switching from New to Open Mode

Press F9 to switch to Open mode to view the records you just added to the database.

Now that you have several horses in your database, you can make them parents. Type the details of the Dam and Sire of Pegasus.

```

File  Selector  Help
-----
MYAEEFR  OPEN                                10/04/95 15:54:48
                                Edit Horse
Horse code .  _____
Select items, then select an action.

Opt  Horse  Horse name      Horse  Horse value  Date of
code  code  name              gender  value         birth
-----
-    BONFIR  Bonfire          M      5000.00      20188
    Dam _____      Dam Date of birth
    Sire _____      Sire Date of birth

-    DOBBIN  Faithful Dobbin        E      3500.00      53186
    Dam _____      Dam Date of birth
    Sire _____      Sire Date of birth

-    PEGASU  Pegasus          M      3000.00      62392
    Dam DOBBIN          Dam Date of birth
    Sire BONFIR          Sire Date of birth

F3=Exit  F5=Reset  F9=New  F10=Actions

```

Press Enter to retrieve the values for the four virtual fields you added, Dam name, Sire name, Dam Date of birth, and Sire Date of birth.

```

File  Selector  Help
-----
MYAEEFR  OPEN                                10/04/95 15:56:04
                                Edit Horse
Horse code .  _____
Select items, then select an action.

Opt  Horse  Horse name      Horse  Horse value  Date of
code  code  name              gender  value         birth
-----
-    BONFIR  Bonfire          M      5000.00      20188
    Dam _____      Dam Date of birth
    Sire _____      Sire Date of birth

-    DOBBIN  Faithful Dobbin        E      3500.00      53186
    Dam _____      Dam Date of birth
    Sire _____      Sire Date of birth

-    PEGASU  Pegasus          M      3000.00      62392
    Dam DOBBIN  Faithful Dobbin      Dam Date of birth  5/31/86
    Sire BONFIR  Bonfire          Sire Date of birth  2/01/88

F3=Exit  F5=Reset  F9=New  F10=Actions

                                CONFIRM: Y (Y/N)

```

Press Enter to accept the default value of Y for the Confirm prompt.

## Exiting the Edit Horse Program

Exit your program by pressing F3.

```

File  Selector  Help
-----
MYAEEFR  OPEN                               10/04/95 15:56:46
                                Edit Horse
Horse code .  █
Select items, then select an action.

Opt  Horse  Horse name  Horse  Horse value  Date of
   code                gender                birth
-  BONFIR  Bonfire      M      5000.00   20188
   Dam      Sire      Dem Date of birth  Sire Date of birth
-  DOBBIN  Faithful Dobbin  E      3500.00   53186
   Dam      Sire      Dem Date of birth  Sire Date of birth
-  PEGASU  Pegasus      M      3000.00   62392
   Dam DOBBIN Faithful Dobbin  Dem Date of birth  5/31/86
   Sire BONFIR  Bonfire      Sire Date of birth  2/01/88

F3=Exit  F5=Reset  F9=New  F10=Actions
  
```

## Exercises

Do the following exercises in the order shown. If you did not write down the program names you can obtain them from the Edit Function Details panel or the Display all functions option on the Display Services Menu.

1. Call the Edit Course program and add some race courses.
2. Call the Edit Jockey program to add some jockeys.
3. Test the Select Stallions and Select Mares functions, by calling the Edit Horse program and typing ? or pressing F4 in the Sire and Dam fields.

# Chapter 6: Maintaining Your Application

---

This chapter introduces the following topics:

- Application Maintenance
- Prototyping (Animation)
- Model Object List Processing
- Function Versioning
- Model Object Cross References
- Impact Analysis

**Note:** These topics are not needed to complete the tutorial. In addition, some of the concepts covered are advanced and may be more suitable after you complete the *Advanced Functions* and *Report Functions* chapters. As a result you can safely skip them now and return to them after completing the *Report Functions* chapter.

## Application Maintenance

The life of an application does not end once development is complete. Over time, the business requirements may change or users may request additional functionality. These application updates can be accomplished easily by applying the same principles and procedures that have been discussed so far.

For example, if you change the length of a field, CA 2E automatically modifies all files and functions that use the changed field, including associated panels and reports. You will need to examine the panel and report designs to ensure that they have not exceeded their limits, and then regenerate the files and functions. The Submit model create process provides options to save data prior to generation and compile.

To aid you in managing changes and identifying the impact of a change on your model, CA 2E provides a set of powerful tools that are the subject of the rest of this chapter.

## Animating an Interactive Device Design

Before making a change, you can test and demonstrate the proposed device design using a process called *prototyping* or *animation*. In this topic you will use CA 2E animation to simulate the link between the Edit Horse function and the Display Racing results function.

New terms introduced

- Animation
- Prototyping
- CA 2E Toolkit

New panels introduced

- Animate Function Panels
- Open Functions
- Toolkit Work with Panel Title Details
- Toolkit Work with Panel Command Key Usage
- Toolkit Edit Choices
- Toolkit Edit Actions

### Overview of CA 2E Animation

CA 2E animation provides a direct link between CA 2E and the Toolkit prototyping facilities. *Toolkit* is a set of implementation, support, and system utilities that include menu design, panel design, prototyping, documentation aids, and object list processing. CA 2E animation lets you transfer control to Toolkit, where you can interactively simulate your CA 2E panel designs, and then easily return to CA 2E.

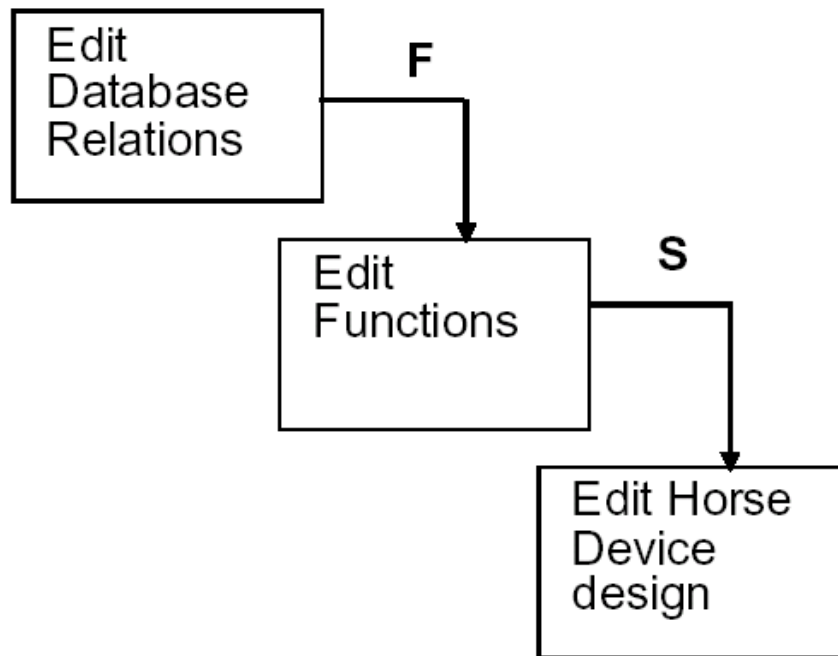
You can animate a function before you make any changes to its default action diagram. This lets you thoroughly test your device design before you commit to specific programming logic, which is more difficult and time-consuming to change. In addition, you do not need to generate and compile a function before you animate it.

For example, you can use animation to demonstrate a proposed design to end users for their review and approval. You can immediately implement any suggestions for improvement of the device design using the CA 2E Device Design Editor and then return to Toolkit to demonstrate the updated design. You can repeat this process until you are satisfied with the design. At that point, you can begin editing the action diagram to implement your design.

## Animate the Edit Horse Device Design

This step demonstrates the process required to animate the link between the Edit Horse and Display Racing results functions.

Begin by displaying the device design for the Edit Horse function. Here's a reminder.



File   fUncion   Selector   Help					
*PROGRAM		*PGHMOD		DD/MM/YY HH:MM:SS	
<b>Edit Horse</b>					
Horse code . _____					
Select items, then select an action.					
Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
—	Dam _____	00000000000000000000000000000000		Dam Date of birth	66/66/66
	Sire _____	00000000000000000000000000000000		Sire Date of birth	66/66/66
—	Dam _____	00000000000000000000000000000000		Dam Date of birth	66/66/66
	Sire _____	00000000000000000000000000000000		Sire Date of birth	66/66/66
—	Dam _____	00000000000000000000000000000000		Dam Date of birth	66/66/66
	Sire _____	00000000000000000000000000000000		Sire Date of birth	66/66/66
F3=Exit   F5=Reset   F9=Open   F10=Actions					

Press F2 to display the Animate Function Panels panel.

Animate Function Panels	My model
Convert Model Panel. : <b>Y</b> (Y-Yes,N-No)	Convert all panels : <b>Y</b> (Y-Yes,N-No)
Replace Navigation : <b>N</b> (Y-Yes,N-No)	
Replace Action Bar : <b>N</b> (Y-Yes,N-No)	
Clear Narrative. : <b>N</b> (Y-Yes,N-No)	
Clear Test Data. : <b>N</b> (Y-Yes,N-No)	
Panel Name(s) . . . . : <b>*SRCHBR</b>	*SRCHBR, *SELECT, *panel, name
File . . . . . : <b>YDSHPNL</b>	Name
Library. . . . . : <b>*MDLLIB</b>	*MDLLIB, *GENLIB, *LIBL, name
Member . . . . . : <b>*FILE</b>	*FILE, name
Display. . . . . : <b>Y</b> (Y-Yes,N-No)	
Display Option . . : <b>1</b> 1-DSPDTA, 2-DSPATR, 3-CHGDTA, 4-WRKPNL	
Return to this device design . . . . : <b>N</b> (Y-Yes,N-No)	
<b>Enter=Execute F3=Exit</b>	

## Animate Functions Panel

This panel is the link between CA 2E and Toolkit. It lets you convert an CA 2E device design to a Toolkit panel design and optionally transfer control to Toolkit. Use options on this panel to specify the tasks you want to accomplish in Toolkit; for example, you can

- Display and work with Toolkit panel designs
- View field and display attributes
- Enter sample data
- Animate panels, windows, and action bars from the current CA 2E device design
- Demonstrate and test navigation using function keys, subfile selection, and action bars

## Animating Edit Horse

In this step you will convert the Edit Horse device design to a Toolkit panel design by accepting the default of Y for the Convert Model Panel field. You also need to convert the CA 2E command key and action bar navigation so you can simulate the link between the two functions. To enter sample data, you need to change the Display Option field to 3 (CHGDTA).

Type **Y** for Replace Navigation, **Y** for Replace Action Bar, and **3** for the Display Option field.



Animate Function Panels		My model
Convert Model Panel :	<u>Y</u> (Y-Yes,N-No)	Convert all panels : <u>Y</u> (Y-Yes,N-No)
Replace Navigation :	<u>Y</u> (Y-Yes,N-No)	
Replace Action Bar :	<u>Y</u> (Y-Yes,N-No)	
Clear Narrative. . . :	<u>N</u> (Y-Yes,N-No)	
Clear Test Data. . . :	<u>N</u> (Y-Yes,N-No)	
Panel Name(s) . . . . :	<u>*SRCMBR</u>	*SRCMBR, *SELECT, *panel, name
File . . . . .	<u>YDSMPL</u>	Name
Library. . . . .	<u>*MDLLIB</u>	*MDLLIB, *GENLIB, *LIBL, name
Member . . . . .	<u>*FILE</u>	*FILE, name
Display. . . . .	<u>Y</u> (Y-Yes,N-No)	
Display Option . . . :	<u>3</u> 1-DSPDTA, 2-DSPATR, 3-CHGDTA, 4-WRKPNL	
Return to this device design . . . . . :	<u>N</u> (Y-Yes,N-No)	
Enter=Execute F3=Exit		

Press Enter to display the Toolkit panel design corresponding to your CA 2E device design. Notice the messages at the bottom of the panel as CA 2E converts the device design and the function key and action bar navigation.

## Entering Sample Data for Edit Horse

You can now enter sample data in the input-capable fields. These are indicated below by underscores. Output-only fields are shown in black.

**Note:** The Toolkit panel design reflects the colors or other display attributes you included in your CA 2E device design.

File fUnction Selector Help					
*PROGRAM		*PGMMOD		DD/MM/YY HH:MM:SS	
Edit Horse					
Horse code . <u>          </u>					
Select items, then select an action.					
Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth
-	Dam	<u>          </u>	<u>          </u>	Dam Date of birth	<u>          </u>
	Sire	<u>          </u>	<u>          </u>	Sire Date of birth	<u>          </u>
-	Dam	<u>          </u>	<u>          </u>	Dam Date of birth	<u>          </u>
	Sire	<u>          </u>	<u>          </u>	Sire Date of birth	<u>          </u>
-	Dam	<u>          </u>	<u>          </u>	Dam Date of birth	<u>          </u>
	Sire	<u>          </u>	<u>          </u>	Sire Date of birth	<u>          </u>
					+
F3=Exit F5=Reset F9=Open F10=Actions					

Type the following sample data. Use the Tab and Field Exit keys as you would on a real panel.

```

File  fUncion  Selector  Help
-----
*PROGRAM      *PGMMODD                                DD/MM/YY HH:MM:SS

Edit Horse

Horse code .  _____

Select items, then select an action.

Opt Horse  Horse name      Horse  Horse value  Date of
code                                     gender
-  BONFIR  Bonfire          M      5000      020188
   Dam _____
   Sire _____      Dam Date of birth _____
   Sire _____      Sire Date of birth _____
-  DOBBIN  Faithful Dobbin    F      3500      053186
   Dam _____
   Sire _____      Dam Date of birth _____
   Sire _____      Sire Date of birth _____
-  PEGASU  Pegasus          M      3000      062392
   Dam BONFIR _____      Dam Date of birth _____
   Sire DOBBIN _____      Sire Date of birth _____

F3=Exit  F5=Reset  F9=Open  F10=Actions
  
```

Press Enter to confirm your entries. This data will be retained until you choose to Clear Test Data on the Animate Function Panels panel. Press F3 to return to the CA 2E device design for Edit Horse.

## Editing and Maintaining Multiple Functions

In the next step you will animate the Display Racing results function using the same process you just used for the Edit Horse function. Since you will need to access the Edit Horse function again later in this process, it will save time not to exit and close it while you animate the Display Racing results function. The Open Functions panel lets you maintain several functions simultaneously.

From the Edit Horse device design, press F3 to exit to the Edit Function Devices panel.

```

EDIT FUNCTION DEVICES                                My model

Function name... : Edit Horse                      Type : Edit file
Received by file : Horse                            Acpth: Retrieval index

? Title
Screen title..... ■ Edit Horse

SEL: Z-Scr/rpt design, N-Narrative, A-Animate
F3=Exit  F5=Action diagram  F15=Open Functions
  
```

## Open Functions Panel

Press F15 to access the Open Functions panel. You can access this panel from many CA 2E panels; for example, option O on the Edit Functions panel lets you open multiple functions. This panel lets you open, edit, and maintain several functions simultaneously; in other words, you do not need to exit and close one function before you open another function for editing. This capability can save a significant amount of time. You can switch quickly and easily between the action diagrams, device designs, generated source, and parameter definitions for all open functions.

Note that the Edit Horse function is listed on the Open Functions panel since you have not yet exited and closed it. To open another function, type the name of the file and the name of the function in the File and Function fields. If you are uncertain of the names, type ? in these fields instead to display selection lists. You can also type \* for the File field if the function you wish to open is attached to the same file as the first open function listed, in this example, the HORSE file.

In this case, you want to open the Display Racing results function on the RACE ENTRY file, type the following names:

**OPEN FUNCTIONS** My Model

Edit Function Action Diagram,

File <b>Race Entry</b>	Function <b>Display Racing results</b>
---------------------------	---

OR enter options for the following:

? File	Function	Type	GEN name
— *2E reserved pgm data	*Notepad	EXCINTFUN	*N/A
— Horse	Edit Horse	EDTFIL	MYAEFR

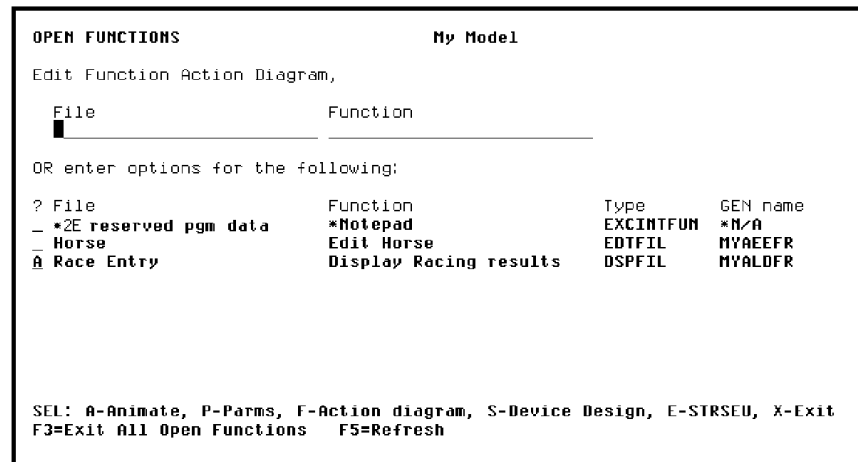
SEL: A-Animate, P-Parms, F-Action diagram, S-Device Design, E-STRSEU, X-Exit  
F3=Exit All Open Functions F5=Refresh

Press Enter to open the Display Racing results function and display its action diagram. Press F15 to return to the Open Functions panel.

Note that Display Racing results now appears on the list. From this panel you can edit a function's device design, action diagram, or parameters; you can also animate the function.

## Animating the Display Racing Results Function

Type **A** in the Subfile selector for the Display Racing results function as shown to animate this function.



Press Enter to display the Animate Function Panels panel.

## Converting Command Key Navigation

In this step you will convert both the device design and the command keys defined for the CA 2E device design, and edit the command key assignments in Toolkit.

To accomplish this, type **Y** for Replace Navigation and type **4** (WRKPNL) for the Display Option.



Press Enter. The CA 2E device design for the Display Racing results function is converted to a Toolkit panel design and control is transferred to Toolkit.

## Working with Toolkit Panel Designs

The Toolkit Work with Panel Titles panel displays. You can use this panel to edit the Toolkit panel design, including the panel layout, command keys, and action bar. Note the following on this panel.

The CA 2E related program name at the bottom of the panel; in this case, MYALDFR. This is the program implementation name of the CA 2E function corresponding to this Toolkit panel design. It serves as the link back to the CA 2E function. If the function is not already open on return to CA 2E, it is automatically added to the Open Functions panel before its device design is displayed.

The name of the panel design; in this case, MYALDFR1. By default this is the CA 2E program implementation name followed by 1. You will need this name when you set up the link in the Edit Horse Toolkit panel design.

Because you requested that command key navigation be replaced during the conversion, type **3** for the Option field as shown to view the results of the command key conversion.

Work with Panel Title Details

Panel. . . . .	MYALDFR1	Next panel. . . :	
Title . . . . .	<u>Display Racing results</u>		
Print sequence. :	<u>          </u>		
Fixed header. . . :	- (Y,blank)		
Fixed Footer. . . :	- (Y,blank)		
Window. . . . .	- (Y,blank)		
Action Bar. . . :	<u>Y</u> (Y,blank)		
Option. . . . .	<u>3</u> 1-Panel, 2-Narrative, 3-Command keys, 5-Window, 6-Action Bar		
2E related program name . . . . .	<u>MYALDFR</u>		

F3=Exit

## Toolkit Command Key Navigation

This panel shows which panel is to be displayed when a command key is pressed during program simulation in Toolkit. Entries are shown only for command keys defined for the function in CA 2E.

Work with Panel Command Key Usage				
Panel. . . . . : MYALDFR1    Display Racing results				
Cmd	Row	Next		
Key	Col	Panel	Text	
01		*SAME	CF01	
02				
03		*EXIT	CF03	
04		*SAME	CF04	
05		*SAME	CF05	
06				
07				
08				
09				
10		*ABAR	Actions	
11				
12				
F3=Exit    F12=Titles screen				

The possible values for the Next Panel column are:

Value	Description
*PRV	Display the previous panel
*SAME	Redisplay the current panel
*EXEC	Execute the command string shown in the Text column
*EXIT	Exit the program
*ABAR	Activate the action bar
name	Display the named Toolkit panel design

If you had not converted the command key navigation when you converted the Display Racing results function, only the F3 command key would be defined. This is done by default to provide a way to return to CA 2E that you can easily remember. You can also return to CA 2E by pressing the Home key.

**Note:** Refer to the documentation for your terminal or computer to learn which key is the Home key on your system.

## Reassigning a Command Key in Toolkit

Since the Display Racing results function is called from the Edit Horse function, the F3 command key should return to Edit Horse. Type the name of the Toolkit panel design that corresponds to the Edit Horse function in the Next Panel column for the F3 command key as shown. By default this is the function's implementation name followed by a 1; for example, MYAEEFR1.

Type the Toolkit panel design name that corresponds to your Edit Horse function. You can also type a ? to select from a list of Toolkit panel designs.

Work with Panel Command Key Usage

Panel. . . . : **MYALDFR1**    **Display Racing results**

Cmd	Row	Next	Panel	Text
01			<b>*SAME</b>	<b>CF01</b>
02				
03			<b>MYAEEFR1</b>	<b>CF03</b>
04			<b>*SAME</b>	<b>CF04</b>
05			<b>*SAME</b>	<b>CF05</b>
06				
07				
08				
09				
10			<b>*ABAR</b>	<b>Actions</b>
11				
12				

+

F3=Exit    F12=Titles screen

Press Enter to confirm your changes. Press F3 to exit.

Exit Work with Panel

1. Exit without update.
2. Exit and update panel design.
3. Return to editing.

Option . . . : **2**

Press Enter to update the panel design and return to the CA 2E Open Functions panel.

## Setting Up Action Bar Navigation for Edit Horse

In this step you will set up the command key and action bar navigation for the Edit Horse function so you can simulate the link between the Edit Horse and the Display Racing results functions using Toolkit.

Type **A** in the Subfile selector to animate the Edit Horse function.

OPEN FUNCTIONS		My Model	
Edit Function Action Diagram,			
File	Function		
OR enter options for the following:			
? File	Function	Type	GEN name
— *2E reserved pgm data	*Notepad	EXCINTFUN	*N/A
A Horse	Edit Horse	EDTFIL	MYAEFR
— Race Entry	Display Racing results	DSPFIL	MYALDR
SEL: A-Animate, P-Parms, F-Action diagram, S-Device Design, E-STREU, X-Exit F3=Exit All Open Functions F5=Refresh			

Press Enter.

Because you have already converted this panel design, including the command key and action bar navigation, you can change the value of the Convert Model Panel to **N**.

Type **N** for Convert Model Panel and change the Display Option to **4** (WRKPNL) to work with the Toolkit panel design.

Animate Function Panels		My model	
Convert Model Panel :	<b>N</b> (Y-Yes,N-No)	Convert all panels :	<b>Y</b> (Y-Yes,N-No)
Replace Navigation :	<b>N</b> (Y-Yes,N-No)		
Replace Action Bar :	<b>N</b> (Y-Yes,N-No)		
Clear Narrative. :	<b>N</b> (Y-Yes,N-No)		
Clear Test Data. :	<b>N</b> (Y-Yes,N-No)		
Panel Name(s) :	<b>*SRCMBR</b>	*SRCMBR, *SELECT, *panel, name	
File :	<b>YDSNPHL</b>	Name	
Library :	<b>*MDLLIB</b>	*MDLLIB, *GENLIB, *LIBL, name	
Member :	<b>*FILE</b>	*FILE, name	
Display :	<b>Y</b> (Y-Yes,N-No)		
Display Option :	<b>4</b> 1-DSPDTA, 2-DSPATR, 3-CHGDTA, 4-WRKPNL		
Return to this device design :	<b>N</b> (Y-Yes,N-No)		
Enter=Execute F3=Exit			

Press Enter.

## Assigning Action Bar Navigation

You need to change the action bar navigation so the Display Racing results Toolkit panel design displays when you select it from the Selector Choice menu on the Edit Horse panel design. Type **6** in the Option field to set up the action bar navigation.



```

Work with Panel Title Details

Panel. . . . . : MYAEFR1      Next panel. . . : _____
Title . . . . . : Edit Horse
Print sequence. : _____
Fixed header. . : _ (Y,blank)
Fixed Footer. . : _ (Y,blank)
Window. . . . . : _ (Y,blank)
Action Bar. . . : Y (Y,blank)

Option. . . . . : 3 1-Panel, 2-Narrative, 3-Command keys,
                    5-Window, 6-Action Bar

2E related program name . . . . . : MYAEFR

F3=Exit
    
```

The Toolkit Edit Choices panel shows the action bar choices that were defined on the CA 2E device design. You need to edit actions for the Selector Choice menu.

Type **A** to edit the actions for the Selector Choice menu.

```

Edit Choices

Panel name . . . . . : MYAEFR1
File . . . . . : YDSNPNL
Library. . . . . : MYMDL
Member . . . . . : YDSNPNL

Choice Sequence . _ (position)

Type options, press Enter.
A=Actions D=Delete

? Sequence Mnemonic Text
- 1 F File
- 4 U fUnction
A 5 S Selector
- 99 H Help

F3=Exit F4=Prompt F9=Go to 'Add' mode F12=Titles screen
    
```

Press Enter to display the Toolkit Edit Actions panel.

Note that the Next Panel column contains \*SAME. This means that, by default, if you select Display Racing results from the action bar while simulating your program, the Edit Horse panel would be displayed again. Instead, you want to display the Toolkit panel design for the Display Racing results function.

Edit Actions			
Panel name . . . : MYAEEFR1			
Choice Sequence . : 5			
Choice Text . . . : Selector			
Action number . <input type="text"/> (position)			
Type options, press Enter.			
C=Command String D=Delete			
?	Number	Text	Next Panel
-	1	<u>Display Racing results</u>	<u>*SAME</u>
F3=Exit F4=Prompt F9=Go to 'Add' mode			

Type the name of the Toolkit panel design assigned to the Display Racing results function in the Next Panel column as shown. By default, this is the CA 2E program implementation name followed by a 1; in this example, MYALDFR1 or type ? for a selection list of Toolkit panel designs.

Edit Actions			
Panel name . . . : MYAEEFR1			
Choice Sequence . : 5			
Choice Text . . . : Selector			
Action number . <input type="text"/> (position)			
Type options, press Enter.			
C=Command String D=Delete			
?	Number	Text	Next Panel
-	1	<u>Display Racing results</u>	<u>MYALDFR1</u>
F3=Exit F4=Prompt F9=Go to 'Add' mode			

Press Enter to confirm the change. Press F3 to return to the Edit Choices menu.

```

Edit Choices

Panel name . . . . . : MYAEEFR1
File . . . . . : YDSNPNL
Library . . . . . : MYMDL
Member . . . . . : YDSNPNL

Choice Sequence .  (position)

Type options, press Enter.
A=Actions D=Delete

? Sequence Mnemonic Text
- 1 F File
- 4 U fUnction
- 5 S Selector
- 99 H Help

F3=Exit F4=Prompt F9=Go to 'Add' mode F12=Titles screen
  
```

Press F3 to exit.

```

Exit Work with Panel

1. Exit without update.
2. Exit and update panel design.
3. Return to editing.

Option . . : 2
  
```

Press Enter to exit and update the Toolkit panel design.

## Testing the Function Link

You are now ready to demonstrate the link between the two functions. Type **S** in the Subfile selector for Edit Horse to display the CA 2E Edit Horse device design.

OPEN FUNCTIONS		My model	
Edit Function Action Diagram,			
File	Function		
OR enter options for the following:			
? File	Function	Type	GEN name
*2E reserved pgm data	*Notepad	EXCINTFUN	*N/A
\$ Horse	Edit Horse	EDTFIL	HYAEEFR
_ Race Entry	Display Racing results	DSPFIL	HYALDFR
SEL: A-Animate, P-Parms, F-Action diagram, S-Device Design, E-STRSEU, X-Exit			
F3=Exit ALL Open Functions F5=Refresh F22=File Locks			

Press Enter.

**Note:** You could have typed **A** in the Subfile selector instead to animate the function. However, if you start the animation from the device design, it is easier to switch between the CA 2E device design and the Toolkit panel design, namely, you can press F2 from the device design to animate and press F3 from the panel design to return to CA 2E.

File	fFunction	Selector	Help
*PROGRAM	*PGM10D		DD/MM/YY HH:MM:SS
Edit Horse			
Horse code . _____			
Select items, then select an action.			
Opt	Horse code	Horse name	Horse value
		Horse gender	Date of birth
-	Dam	00000000000000000000000000000000	Dam Date of birth 66/66/66
	Sire	00000000000000000000000000000000	Sire Date of birth 66/66/66
-	Dam	00000000000000000000000000000000	Dam Date of birth 66/66/66
	Sire	00000000000000000000000000000000	Sire Date of birth 66/66/66
-	Dam	00000000000000000000000000000000	Dam Date of birth 66/66/66
	Sire	00000000000000000000000000000000	Sire Date of birth 66/66/66
F3=Exit F5=Reset F9=Open F10=Actions			

## Animating Edit Horse

Press F2 to display the Animate Function Panels panel. Since you have already converted the CA 2E device design to a Toolkit panel design you can either accept the defaults or type **N** for the Convert Model panel field. Typing **N** can save time. In this case accept the defaults.

Animate Function Panels		My model
Convert Model Panel. :	<u>Y</u> (Y-Yes,N-No)	Convert all panels : <u>Y</u> (Y-Yes,N-No)
Replace Navigation :	<u>N</u> (Y-Yes,N-No)	
Replace Action Bar :	<u>N</u> (Y-Yes,N-No)	
Clear Narrative. :	<u>N</u> (Y-Yes,N-No)	
Clear Test Data. :	<u>N</u> (Y-Yes,N-No)	
Panel Name(s) . . . . :	<u>*SRCMBR</u>	*SRCMBR, *SELECT, *panel, name
File . . . . . :	<u>YDSHPNL</u>	Name
Library. . . . . :	<u>*MDLLIB</u>	*MDLLIB, *GENLIB, *LIBL, name
Member . . . . . :	<u>*FILE</u>	*FILE, name
Display. . . . . :	<u>Y</u> (Y-Yes,N-No)	
Display Option . . . :	<u>1</u> 1-DSPDTA, 2-DSPATR, 3-CHGDTA, 4-WRKPNL	
Return to this device design . . . . . :	<u>N</u> (Y-Yes,N-No)	
Enter=Execute F3=Exit		

Press Enter. The Toolkit panel design for the Edit Horse function displays showing the sample data you entered earlier. Input-capable fields are indicated by underscores; output-only fields are shown in black.

## Activate the Action Bar

To demonstrate the link between the two functions type / against one of the subfile records and press F10 to activate the action bar.

File Selector Help						
*PROGRAM		*PGM*MOD		DD/MM/YY HH:MM:SS		
Edit Horse						
Horse code . _____						
Select items, then select an action.						
Opt	Horse code	Horse name	Horse gender	Horse value	Date of birth	
/	<u>BONFIR</u>	<u>Bonfire</u>	<u>M</u>	<u>5000</u>	<u>020188</u>	
	Dam _____			Dam Date of birth _____		
	Sire _____			Sire Date of birth _____		
-	<u>DOBBIN</u>	<u>Faithful Dobbin</u>	<u>F</u>	<u>3500</u>	<u>053186</u>	
	Dam _____			Dam Date of birth _____		
	Sire _____			Sire Date of birth _____		
-	<u>PEGASU</u>	<u>Pegasus</u>	<u>M</u>	<u>3000</u>	<u>062392</u>	
	Dam _____			Dam Date of birth _____		
	Sire _____			Sire Date of birth _____		+
F3=Exit F5=Reset F9=Open F10=Actions						

Type **S** to select the Selector Choice menu from the action bar; note that it includes Display Racing results as an action. Type **1** to select Display Racing results.

```

File  Selector  Help
*PROGR  1  1. Display Racing results  DD/MM/YY HH:MM:SS
Horse code .  orse
Select items, then select an action.

Opt Horse  Horse name  Horse  Horse value  Date of
code      gender      birth
< BONFIR  Bonfire      5000  020188
  Dam      Sire      Dam Date of birth  Sire Date of birth
  DOBBIN  Faithful Dobbin  3500  053186
  Dam      Sire      Dam Date of birth  Sire Date of birth
  PEGASU  Pegasus      3000  062392
  Dam      Sire      Dam Date of birth  Sire Date of birth
F3=Exit  F5=Reset  F9=Open  F10=Actions
  
```

Press Enter.

## Display Racing Results Panel Design

The Toolkit panel design for the Display Racing results function displays. Input-capable fields are indicated by underscores; output-only fields are shown in black.

```

-----
*PROGRAM  *PGMMOD  Display Racing results  DD/MM/YY HH:MM:SS
Horse code  Horse name  Race date  Race time
-----
Opt  Race date  Race time  Course code  Entry number  Jockey code  Finishing position  Handicap  Entry Status
-
-
-
-
-
-
-
F3=Exit  F5=Reset  F10=Actions
  
```

Next test the command key navigation you set up for the Display Racing results panel design by pressing F3. This should cause a return to the Toolkit panel design for Edit Horse.

```

File  fUnction  Selector  Help
-----
*PROGRAM  *PGMMOD                      DD/MM/YY HH:MM:SS

Horse code .                      Edit Horse

Select items, then select an action.

Opt  Horse  Horse name          Horse  Horse value  Date of
   code                gender                birth
-  BONFIR  Bonfire                M          5000      020188
   Dam                                Dam Date of birth           
   Sire                                Sire Date of birth           
-  DOBBIN  Faithful Dobbin          F          3500      053186
   Dam                                Dam Date of birth           
   Sire                                Sire Date of birth           
-  PEGASU  Pegasus                M          3000      062392
   Dam                                Dam Date of birth           
   Sire                                Sire Date of birth            +

F3=Exit  F5=Reset  F9=Open  F10=Actions
  
```

Press F3 to return to the CA 2E device design for the Edit Horse function.

You could now make design changes to the device design and repeat the animation process until you are satisfied with the results.

**Note:** If you have defined command key or action bar navigation in Toolkit, do **not** choose the options to replace action bar or command key navigation when you reanimate. In other words, accept the defaults for the Replace options on the Animate Function Panels panel.

## Exiting Both Functions

Press F3 to exit the Device Design Editor. At the Edit Function Devices panel press F15 to return to the Open Functions panel. From this panel you can either exit all open functions, or you can exit functions individually using the **X** Subfile selector option. In this case, you will exit all open functions.

```

OPEN FUNCTIONS                      My Model

Edit Function Action Diagram,

File                    Function                   

OR enter options for the following:

? File          Function          Type          GEN name
- *2E reserved pgm data  *Notepad          EXCINTFUN      *N/A
- Horse          Edit Horse          EDTFIL        MYAEEFR
- Race Entry      Display Racing results  DSPFIL        MYALDFR

SEL: A-Animate, P-Parms, F-Action diagram, S-Device Design, E-STRSEU, X-Exit
F3=Exit All Open Functions  F5=Refresh
  
```

Press F3 to exit all open functions. You will proceed through the following steps.

The Exit Function Definition panel displays a second time for the Edit Horse function. Press Enter to accept the defaults.

The Exit Function Definition panel displays first for the Display Racing results function. Press Enter to accept the defaults.

The Edit Functions panel for the HORSE file displays. Press F3 to exit and return to the Edit Database Relations panel.

Press F3 to exit the model.

## Working with Model Object Lists

This topic introduces the CA 2E model object list utility.

New terms introduced

- Model object
- Model object type
- Model object list
- Model object list entry
- Session list
- All Objects list (\*ALLOBJ)

New panels introduced

- Edit Model Object List
- Display Model Object
- Work with Model Lists
- Subset Model Objects
- Position the List window

## Objectives

To introduce the Edit Model Object List panel, to use some of its options and capabilities, and to encourage you to explore the model object list facilities.



**Note:** This topic presents only examples of ways to use model object lists. After you complete these examples be sure to experiment on your own until you feel comfortable with this tool. Refer to the CA 2E guide, *Generating and Implementing Applications* for more information.

## Overview of Model Objects and Model Object Lists

By definition a *model object* is anything in the model that you can refer to by name; for example, a file (HORSE) or a function (Edit Horse).

### Model Object Types

Within CA 2E there are seven types of model objects, many of which you have defined and used in this tutorial. Each *model object type* has a 3-letter identifier as shown in the following table.

Model Object	Model Object Type
Files	FIL
Fields	FLD
Conditions	CND
Access Paths	ACP
Functions	FUN
Messages	MSG
Arrays	ARR
Application Areas	APP

### Model Object Lists

A *model object list* is a logical grouping of model objects. The way in which you use model object lists is limited only by your imagination. For example, a model object list might consist of all

- Model objects changed since a specified date
- Model objects related to a particular development project
- Access paths and functions that need to be generated and compiled as a result of a change to the model
- Model objects needed to implement the accounts payable feature of your application

- Functions called by an external function in which an execution error occurred
- Model objects that use a specified model object

Since a model object list can be used in many ways, it can be referred to by other names based on the way in which it is used. For example, a specific model object list might be called a session list, copy list, change list, or model list, for short. In reality, these are all just variations on the same concept.

## Model Object List Entry

A model object list is comprised of a set of references to model objects within the model. Each reference is known as a *model object list entry* or *list entry* for short. Each list entry contains information about a model object at the time the list entry was created. In other words, it provides a persistent historic record of the object at the time the list entry was created.

## Editing Your Session List

At the beginning of this tutorial, you created a session list with the same name as your model profile. A *session list* is a model object list to which all model objects you change, add, or delete during a session are logged. Your session list should now contain all objects you added or changed while working on the horse racing model.

From the CA 2E Designer (\*DSNR) Menu, select the Edit Session List (changed objects) option.

```
DSNR                      2E Designer (*DSNR) Menu
Level . :    1
System:                2EDV1

Select one of the following:

Enter Model              1. Edit Database Relations
                        2. Services Menu
                        3. Edit Default Model Object List
                        4. Edit Session List (changed objects)
                        5. Work with Model Objects
                        6. Load model and display command line
                        8. Work with Model Object Lists
                        9. Change to work with another model

Open Access:             ? 10. Change Open Access Model Value
enter with *NO           11. Edit Database Relations
                        12. Services Menu

Selection or command
==> 4

More...

F3=Exit  F6=Messages  F9=Prev. request  F10=Cmd Entry  F14=Submitted jobs _
```

Press Enter.

The Edit Model Object List panel displays showing all model objects you changed during the tutorial.

## Edit Model Object List Panel

The Edit Model Object List panel serves as an alternate entry point into your model. It has a PDM-like interface that you can use to perform most functions available from the Edit Database Relations panel. The exceptions are that you cannot edit relations or create model objects.

In the following steps you will use the Edit Model Object List panel to edit the session list you created at the beginning of this tutorial.

**Edit Model Object List**

Model . . : MYMDL  
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.  
1=Select      2=Edit      3=Copy      4=Delete entry  
5=Display      8=Details      9=Deselect      10=Action diagram

Opt	Object	Type	Attr	Owner
<u>1</u>	*ALL values	CND	LST	Horse gender
—	*ALL values	CND	LST	Jockey gender
—	*ALL values	CND	LST	Going conditions
—	*ALL values	CND	LST	Entry Status
—	Change Course	FUN	DBF	Course
—	Change Horse	FUN	DBF	Horse
—	Change Jockey	FUN	DBF	Jockey

More...

Parameters or command  
===> \_\_\_\_\_

F3=Exit    F4=Prompt    F5=Refresh    F6=Build    F7=Position to  
F8=Reverse retrieve    F9=Retrieve    F23=More options    F24=More keys

## Viewing Model Object Types

By default model objects are listed alphabetically by object name and type. Press Roll Up several times to view some of the list. Note the values in the Type column.

## List Entry Differs from Model Object

Press Roll Up until you see the list entry for Dam Horse name. Note the number 8 to the right of the Subfile selector. This indicates that information for the list entry on your session list differs from that of the actual model object that the list entry represents. Recall that earlier in the tutorial, you changed the name of this field from Dam Horse name to Dam name.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select      2=Edit      3=Copy      4=Delete entry
5=Display     8=Details   9=Deselect 10=Action diagram

Opt   Object                                Type  Attr Owner
---   ---
8     Dam Horse name                       FLD   REF
      Dam younger than horse               MSG   ERR  *Messages
      Date of birth                         FLD   DT#
      Delete Course                         FUN   DBF  Course
      Delete Horse                         FUN   DBF  Horse
      Delete Jockey                        FUN   DBF  Jockey
      Delete Race                          FUN   DBF  Race

Parameters or command
===>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys
More...

```

**Note:** CA 2E allows this difference to occur so you can use your model object list as a historical record.

You can update the model object list entry to reflect the current information for the actual model object. To do so, type **33** for the Subfile selector for Dam Horse name.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select      2=Edit      3=Copy      4=Delete entry
5=Display     8=Details   9=Deselect 10=Action diagram

Opt   Object                                Type  Attr Owner
---   ---
33    8     Dam Horse name                       FLD   REF
      Dam younger than horse               MSG   ERR  *Messages
      Date of birth                         FLD   DT#
      Delete Course                         FUN   DBF  Course
      Delete Horse                         FUN   DBF  Horse
      Delete Jockey                        FUN   DBF  Jockey
      Delete Race                          FUN   DBF  Race

Parameters or command
===>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys
More...

```

Press Enter. The list entry is updated from the detail information for the actual model object. In this case, the name of the list entry changes to match the name of the model object it represents.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select      2=Edit      3=Copy      4=Delete entry
5=Display     8=Details   9=Deselect 10=Action diagram

Opt  Object                Type  Attr  Owner
---  ---
1    Dam name              FLD   REF
2    Dam younger than horse MSG   ERR  *Messages
3    Date of birth         FLD   DT#
4    Delete Course         FUN   DBF  Course
5    Delete Horse          FUN   DBF  Horse
6    Delete Jockey         FUN   DBF  Jockey
7    Delete Race           FUN   DBF  Race

Parameters or command
==>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

## Positioning a Model Object List

You can position the model list displayed to a specific object type, object name, owner, or implementation name. This topic presents a few examples of ways to use positioning. Feel free to experiment on your own.

Press F7 to display the Position the List window. The values you enter in this positioner window also determine the order in which the model objects display.

Type **FUN** (function).

```

Edit Model Object List

Position the List

Type . . : FUN  Object . . : 
Owner . . :      Imp. Name . . : 

F12=Cancel

Opt  Object                Type  Attr  Owner
---  ---
1    Dam name              FLD   REF
2    Dam younger than horse MSG   ERR  *Messages
3    Date of birth         FLD   DT#
4    Delete Course         FUN   DBF  Course
5    Delete Horse          FUN   DBF  Horse
6    Delete Jockey         FUN   DBF  Jockey
7    Delete Race           FUN   DBF  Race

Parameters or command
==>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

Press Enter.

Note that the list is now positioned at the first model object of type FUN. The following model objects are displayed alphabetically by type, and for each type, alphabetically by object name. Note the message displayed at the bottom of the panel indicating this.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select  2=Edit    3=Copy    4=Delete entry
5=Display 8=Details 9=Deselect 10=Action diagram

Opt  Object                Type  Attr  Owner
---  ---
1    Change Course          FUN   DBF   Course
2    Change Horse           FUN   DBF   Horse
3    Change Jockey          FUN   DBF   Jockey
4    Change Race            FUN   DBF   Race
5    Change Race Entry      FUN   DBF   Race Entry
6    Create Course          FUN   DBF   Course
7    Create Horse           FUN   DBF   Horse

More...

Parameters or command
===>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build    F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys
Data displayed in Object Type/Object Name order.

```

Scroll through the list to verify this.

Press F7 again. Blank out the Type field and type **HORSE** for Owner.

```

Edit Model Object List

Position the List

Type . . : Object . . :
Owner . . : HORSE      Imp. Name
F12=Cancel

Opt  Object                Type  Attr  Owner
---  ---
1    *ALL values           CND   LST   Horse gender
2    *ALL values           CND   LST   Jockey gender
3    *ALL values           CND   LST   Going conditions
4    *ALL values           CND   LST   Entry Status
5    Change Course          FUN   DBF   Course
6    Change Horse           FUN   DBF   Horse
7    Change Jockey          FUN   DBF   Jockey

More...

Parameters or command
===>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build    F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

Press Enter.

Note that the list is now positioned at the first model object that has HORSE as its owner. Note the message at the bottom of the panel. The following model objects are displayed alphabetically by Owner, and for each Owner, alphabetically by Object Type, and then by Object Name. Scroll through the list to verify this.

## Returning to the Top of the List

You can also use the positioner window to return quickly to the top of the model list displayed.

Scroll down so the list is not positioned at the top. Press F7 again. Blank out all fields.

**Edit Model Object List**

**Position the List**

Type . . .      Object . . .      Imp. Name

Owner . . .     

F12=Cancel

Opt	Object	Type	Attr	Owner
—	Mares	ACP	RTU	Horse
—	Physical file	ACP	PHY	Horse
—	Retrieval index	ACP	RTU	Horse
—	Stallions	ACP	RTU	Horse
—	Update index	ACP	UPD	Horse
—	Change Horse	FUN	DBF	Horse
—	Create Horse	FUN	DBF	Horse

More . . .

Parameters or command  
 ==>

F3=Exit   F4=Prompt   F5=Refresh   F6=Build   F7=Position to  
 F8=Reverse retrieve   F9=Retrieve   F23=More options   F24=More keys

**Data displayed in Object Owner Name/Object Type/Object Name order.**

Press Enter. The list will be repositioned to the top of the list in order by object name and type.

## Viewing a Subset of a Model Object List

You can also display a subset of a model object list, based on such criteria as Object name, Type, Owner, Implementation name, and the date the model objects were created or changed.

Suppose you want to view all Select Record (SELRCRD) functions on your session list. To do so, press F17 to display the Subset Model Objects panel. (Press F24 to see more command keys.) Type **\*FUN** for the Type field and type **SELRCRD** for the Function type field.

✖

Press Enter twice.

```

                                Edit Model Object List

Model . . : MYMDL
List . . . JAR_____ List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select  2=Edit    3=Copy    4=Delete entry
5=Display 8=Details 9=Deselect 10=Action diagram

Opt  Object                                Type  Attr Owner
┌───┬──────────────────────────────────┬───┬───┬───
│ 1 │ Select Course                      │ FUN │ RPG │ Course
│───┴──────────────────────────────────┴───┴───┴───
│ 2 │ Select Horse                      │ FUN │ RPG │ Horse
│───┴──────────────────────────────────┴───┴───┴───
│ 3 │ Select Jockey                    │ FUN │ RPG │ Jockey
│───┴──────────────────────────────────┴───┴───┴───
│ 4 │ Select Mares                     │ FUN │ RPG │ Horse
│───┴──────────────────────────────────┴───┴───┴───
│ 5 │ Select Race                      │ FUN │ RPG │ Race
│───┴──────────────────────────────────┴───┴───┴───
│ 6 │ Select Stallions                 │ FUN │ RPG │ Horse
└───┴──────────────────────────────────┴───┴───┴───

Parameters or command
==>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve F9=Retrieve  F23=More options F24=More keys
This is a subsetting list.
                                Bottom

```

The list now displays only the model objects that meet the criteria you entered on the Subset Model Objects panel. Note the message at the bottom of the panel indicating that the list is subsetting; in other words, not all model objects in the list are displayed.

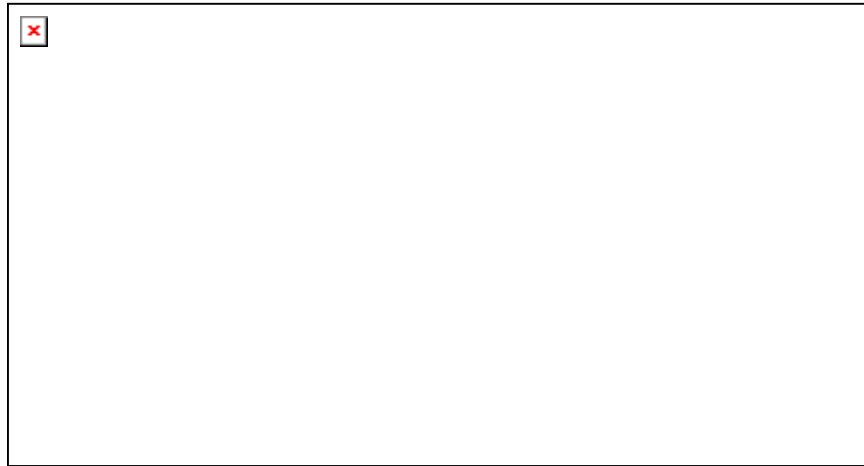
Subsetting is useful when you work with long lists or when you want to work only with specific types of model objects; for example, functions or access paths. Spend time experimenting with this feature.

## Opening Multiple Functions at One Time

Suppose you want to work with each of the Select Record functions on your model list; for example, to edit their device designs or action diagrams. Press F23 three times to view additional Subfile selector options. Option 30 lets you open several functions at once.

Type **30** in the Subfile selector of the first function displayed, Select Course.





Next, press F24 to view additional command keys. The F13 command key automatically repeats the Subfile selector option you entered for all following subfile records to the end of the list.

Press F13 to repeat option 30 for each function displayed.

Edit Model Object List				
Model . . : MYMDL				
List . . . JAR List JAR in MYMDL created by user JAR.				
Type options, press Enter.				
24=Delete object	25=Document function	26=Redirect	28=Checkout	
30=Open function	31=Object locks	33=Refresh entry		
Opt	Object	Type	Attr	Owner
30	Select Course	FUN	RPG	Course
30	Select Horse	FUN	RPG	Horse
30	Select Jockey	FUN	RPG	Jockey
30	Select Mares	FUN	RPG	Horse
30	Select Race	FUN	RPG	Race
30	Select Stallions	FUN	RPG	Horse
				<b>Bottom</b>
Parameters or command				
===>				
F10=Execute list	F11=Alt view	F12=Cancel	F13=Repeat	F14=Filter
F15=Check list	F17=Subset	F18=Change model profile	F24=More keys	
<b>Option 30 was repeated to the end of the list.</b>				

A message at the bottom of the screen tells you that the option was repeated to the end of the model list.

Press Enter to open each of the selected functions. The functions are added to the Open Functions panel.

OPEN FUNCTIONS		My model	
Edit Function Action Diagram,			
File	Function		
OR enter options for the following:			
? File	Function	Type	GEN name
- *2E reserved pgm data	*Notepad	EXCINTFUN	*N/A
- Course	Select Course	SELRCO	MYABSRR
- Horse	Select Horse	SELRCO	MYAISRR
- Jockey	Select Jockey	SELRCO	MYAMSRR
- Horse	Select Mares	SELRCO	MYAJSRR
- Race	Select Race	SELRCO	MYAOSRR
- Horse	Select Stallions	SELRCO	MYAKSRR
SEL: A-Animate, P-Parms, F-Action diagram, S-Device Design, E-STRSEU, X-Exit F3=Exit All Open Functions F5=Refresh F22=File Locks			

## Working with Open Functions

From the Open Functions panel you can work with each of the functions listed without incurring the overhead of exiting and reopening each time you need to work with a different function.

Press F3 to exit all the open functions. CA 2E displays the Exit Function Definition panel for each open function in turn where you can choose whether to save the function and submit it for batch generation.

When you exit the last open function, CA 2E returns to the Edit Model Object List panel.

## Displaying the Unsubsetting Model List

Because only SELRCO functions are displayed, press F17 to display the Subset Model Objects panel. Press F5 to restore the default values for all fields.

Subset Model Objects		
Type choices, press Enter.		
Object . . . . .	<u>*ALL</u>	*ALL, name *generic*
Type . . . . .	<u>*ALL</u>	*ALL,*ACP,*APP,*ARR,*CND, *FIL,*FLD,*FUN,*MSG
Function type . . . .	<u>*ALL</u> (F4 for list)	*ALL, name *generic*
Attribute . . . . .	<u>*ALL</u>	*ALL, name
Owner . . . . .	<u>*ALL</u>	*ALL, name *generic*
Implementation name .	<u>*ALL</u>	*ALL, name *generic*
Create date:		
From date . . . . .	<u>00/00/00</u>	Date
To date . . . . .	<u>99/99/99</u>	Date
Change date:		
From date . . . . .	<u>00/00/00</u>	Date
To date . . . . .	<u>99/99/99</u>	Date
Omit system objects .	<u>*YES</u>	*YES,*NO
Change type . . . . .	<u>*ALL</u>	*ALL,*OBJ,*GEN,*PUT,*PUB
Selection status . . .	<u>*ALL</u>	*ALL, *SELECTED
F5=Refresh F12=Cancel		

Press Enter to display the original unsubsetted list.

## Editing Model Objects

Subfile selector option **2** on the Edit Model Object List panel accesses the appropriate CA 2E editing panel according to the type of the selected model object. The following table shows examples for object types and panels you have used earlier in this tutorial.

Model Object Type	Tutorial Panel
Type	CA 2E Editing Panel Displayed
ACP	Edit Access Path Details
CND	Edit List Condition for LST Edit Field Condition Details for VAL
FIL	Edit File Details
FLD	Edit Field Details
FUN	Edit Function Details
MSG	Edit Message Function Details

If you type **2** for the Subfile selector of several model objects of differing types, CA 2E displays the appropriate editing panel for each model object, one at time.

## Exercise

Type **2** in the Subfile selector for model objects of several different types of model objects and press Enter. Notice which CA 2E editing panel displays in each case. Press F3 to return to the Edit Model Object List panel.

## Editing Conditions for Entry Status

You can also edit only specific objects by subsetting the model list first. Suppose you want to view and possibly change the condition values you defined for the Entry Status field on the RACE ENTRY file. For example, you can change the condition names and status values. Note that you cannot add a new condition in this way since you need to use the Edit Database Relations panel to create a model object.

Press F17 to display the Subset Model Objects panel. Type **\*CND** for the Type field, **VAL** for the Attribute field, and **Entry Status** for the Owner field.

Subset Model Objects		
Type choices, press Enter.		
Object . . . . .	<u>*ALL</u>	*ALL, name *generic*
Type . . . . .	<u>*CND</u>	*ALL, *ACP, *APP, *ARR, *CND,
		*FIL, *FLD, *FUN, *MSG
Function type . . . .	<u>*ALL</u> (F4 for list)	*ALL, name *generic*
Attribute . . . . .	<u>VAL</u>	*ALL, name
Owner . . . . .	<u>Entry Status</u>	*ALL, name *generic*
Implementation name .	<u>*ALL</u>	*ALL, name *generic*
Create date:		
From date . . . . .	<u>0/00/00</u>	Date
To date . . . . .	<u>99/99/99</u>	Date
Change date:		
From date . . . . .	<u>0/00/00</u>	Date
To date . . . . .	<u>99/99/99</u>	Date
Omit system objects .	<u>*YES</u>	*YES, *NO
Change type . . . . .	<u>*ALL</u>	*ALL, *OBJ, *GEN, *PUT, *PUB
Selection status . . .	<u>*ALL</u>	*ALL, *SELECTED
F5=Refresh F12=Cancel		

Press Enter twice to display the subsetted list.

**Edit Model Object List**

Model . . : MYMDL  
List . . . : JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter:

24=Delete object	25=Document function	26=Redirect
30=Open function	31=Object locks	33=Refresh entry

28=Checkout

Opt	Object	Type	Attr	Owner
█	Disqualified	CND	VAL	Entry Status
—	Finished	CND	VAL	Entry Status
—	Not yet run	CND	VAL	Entry Status
—	Scratched	CND	VAL	Entry Status

**Bottom**

Parameters or command  
==>

F10=Execute list	F11=Alt view	F12=Cancel	F13=Repeat	F14=Filter
F15=Check list	F17=Subset	F18=Change model profile	F24=More keys	

**This is a subsetted list.**

## Exercise

View and optionally change each condition value. Namely, type **2** for the first condition value displayed, and then press F13 to repeat this option for the rest of the condition values. Press Enter to display the Edit Field Condition Details panel for each condition value. Press F3 to exit each editing panel and finally to return to the Edit Model Object List panel.

## Displaying the Unsubsetted Model List

From the Edit Model Object List panel, press F17 to display the Subset Model Objects panel. Press F5 to restore the default values for all fields and press Enter to display the original unsubsetted list.

## Editing Relations and Creating Objects

As already mentioned you cannot edit relations or create model objects using the Edit Model Object List panel. If you need to edit a relation or create a model object, you can transfer temporarily to the Edit Database Relations panel by entering the Edit Model (YEDTMDL) command on the command line.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
24=Delete object      25=Document function      26=Redirect      28=Checkout
30=Open function      31=Object locks           33=Refresh entry

Opt   Object                                     Type  Attr  Owner
—
—      *ALL values                               CND   LST   Horse gender
—      *ALL values                               CND   LST   Jockey gender
—      *ALL values                               CND   LST   Going conditions
—      *ALL values                               CND   LST   Entry Status
—      Change Course                             FUN   DBF   Course
—      Change Horse                             FUN   DBF   Horse
—      Change Jockey                             FUN   DBF   Jockey
More...

Parameters or command
==> YEDTMDL
F10=Execute list      F11=Alt view      F12=Cancel      F13=Repeat      F14=Filter
F15=Check list        F17=Subset        F18=Change model profile      F24=More keys

```

Press Enter.

```

EDIT DATABASE RELATIONS           My model
=>
? Typ Object                      Rel lvl:  Seq Typ Referenced object
■ FIL Course                     Known by   FLD Course code
— FIL Course                     Has       FLD Course name
— FIL Horse                      Known by   10 FLD Horse code
— FIL Horse                      Has       20 FLD Horse name
— FIL Horse                      Has       30 FLD Horse gender
— FIL Horse                      Has       40 FLD Horse value
— FIL Horse                      Has       50 FLD Date of birth
— FIL Horse                      Refers to  60 FIL Horse
  For: Dam                      Sharing: *ALL
— FIL Horse                      Refers to  70 FIL Horse
  For: Sire                      Sharing: *ALL
— FIL Jockey                     Known by   FLD Jockey code
— FIL Jockey                     Has       FLD Jockey name
— FIL Jockey                     Has       FLD Jockey gender
— FIL Race                      Owned by   FIL Course
More...

Z(n)=Details  F=Functions  E(n)=Entries  S(n)=Select      F23=More options
F3=Exit      F5=Reload    F6=Hide/Show  F7=Fields      F9=Add/Change  F24=More keys

```

When you finish editing relations, press F3 to return to the Edit Model Object List panel.

## Deleting a Model List Entry

The Edit Model Object List panel provides two deletion options. Option 24 deletes the actual model object from the model; option 4 deletes a list entry from the displayed model object list. When you delete a model object list entry (option 4), CA 2E displays a confirm panel for each selected list entry.

Note that you cannot delete a model object if it is used by other objects in the model. CA 2E provides impact analysis tools to help you determine whether and how a model object is used by other model objects. Impact analysis is discussed later in this chapter.

**Note:** Because you can use a model object list as a historic record for your model, it can contain entries for model objects that you have deleted from the model. These list entries are indicated by an X to the right of the Subfile selector.

For practice you will delete a list entry from your session list in this step. You will replace it later so your session list continues to represent all changes to your model. Remember, you are deleting a list entry, not the model object.

Enter **4** in the Subfile selector for the Change Course internal function.

```

Edit Model Object List

Model . . : MYMDL
List . . : JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
24=Delete object    25=Document function    26=Redirect    28=Checkout
30=Open function    31=Object locks        33=Refresh entry

Opt    Object                Type Atr Owner
---
---      *ALL values            CND   LST   Horse gender
---      *ALL values            CND   LST   Jockey gender
---      *ALL values            CND   LST   Going conditions
---      *ALL values            CND   LST   Entry Status
4      Change Course           FUN   DBF   Course
---      Change Horse           FUN   DBF   Horse
---      Change Jockey          FUN   DBF   Jockey

Parameters or command
==>
F10=Execute list    F11=Alt view    F12=Cancel    F13=Repeat    F14=Filter
F15=Check list      F17=Subset      F18=Change model profile    F24=More keys
More...

```

Press Enter. CA 2E displays a confirm panel.

```

Confirm Delete of Model List Entry

Model . . : MYMDL
List . . : JAR      List JAR in MYMDL created by user JAR.

Press ENTER to confirm your choices for Delete.
Press F12=Cancel to return to change your choices.

Object                Type Owner                Checkout Impl.
status      name
XChange Course           FUN   Course

F12=Cancel

Bottom

```

Press Enter to confirm the deletion.

## All Objects List

Each model contains a special list containing current information about each object in the model. This list is referred to as the All Objects list and is accessed within the model as \*ALLOBJ.

The major purpose of the All Objects List is to provide a central location for model object information and to record information related to changes. The information for each model object is contained in a model object description, or detail. As you create, delete, update, and generate model objects, CA 2E automatically updates the corresponding model object description in the All Objects list to reflect the changes.

## Accessing the All Objects List

One way to access the All Objects List is to type \*ALLOBJ in the List field at the top of the Edit Model Object List panel.

Type **\*ALLOBJ** for the List field.

**Edit Model Object List**

Model . . : MYMDL  
List . . . \*ALLOBJ List JAR in MYMDL created by user JAR.

Type options, press Enter.  
1=Select    2=Edit    3=Copy    4=Delete entry  
5=Display    8=Details    9=Deselect    10=Action diagram

Opt	Object	Type	Atr	Owner
—	*ALL values	CND	LST	Horse gender
—	*ALL values	CND	LST	Jockey gender
—	*ALL values	CND	LST	Going conditions
—	*ALL values	CND	LST	Entry Status
—	Change Horse	FUN	DBF	Horse
—	Change Jockey	FUN	DBF	Jockey
—	Change Race	FUN	DBF	Race

More...

Parameters or command  
===>

F3=Exit	F4=Prompt	F5=Refresh	F6=Build	F7=Position to
F8=Reverse retrieve	F9=Retrieve	F23=More options	F24=More keys	

Press Enter.

The List field now contains \*ALLOBJ. This means there is a one-to-one correspondence between the model objects displayed and the actual model objects that comprise your model. In particular, note that the model object you deleted from your session list, namely, Change Course, appears on the All Objects List.



## How \*ALLOBJ and Model Lists Differ

Technically, the All Objects list is not really a model object list; however, it is often useful to think of it as a list. In general, you can use the Edit Model Object List panel to work with the All Objects List as you would a named model object list. However, since the All Objects list represents actual model objects, note the following exceptions.

You can delete a model object, but you cannot delete a list entry. In other words, you can use option 24, but not option 4.

You cannot add a model object to the All Objects list since that requires creating a model object.

While the All Objects list contains all active model objects, named model object lists typically represent a subset of all model objects in the model.

## Displaying Alternate Views of Detail Information

The Edit Model Object List panel provides five alternate views of detail information for each model object. The information shown in the five views is dynamic when you are working with the All Objects list; in other words, if you update a model object, CA 2E automatically updates the information displayed to reflect the change. For named model object lists, the information displayed is static and reflects the state of the model object at the time it was added to the model object list.

The following views are available:

- Basic information
- Implementation information
- Component Change information
- Change information
- Check out information

Details about each of these views are beyond the scope of this tutorial. To learn more, see the CA 2E module, *Generating and Implementing Applications*.

## Exercise

Press F11 to view the five alternate views of information from the detail for the model objects displayed. Some of this information depends on the type of the model object so you should scroll through the list while viewing each alternate view.

## Restoring the Deleted Session List Entry

In this step you will restore the Change Course function to your session list. Subfile selector option 11 lets you select model objects to be added to another model object list. The default for this alternate list is the Model list for commands specified in your model profile. In your case it is the same as your session list.

Type **11** in the Subfile selector for the Change Course model object.

```

Edit Model Object List

Model . . : MYMDL
List . . . *ALLOBJ  *All objects list for model MYMDL.

Type options, press Enter.
11=Add to alternate list      13=Parameters      14=GEN batch
15=GEN interactive          16=Y2CALL          17=Device design

Opt   Object                Type  Attr Owner
---   ---
---   *ALL values           CND   LST  Horse gender
---   *ALL values           CND   LST  Jockey gender
---   *ALL values           CND   LST  Going conditions
---   *ALL values           CND   LST  Entry Status
11    Change Course         FUN   DBF  Course
---   Change Horse         FUN   DBF  Horse
---   Change Jockey        FUN   DBF  Jockey

Parameters or command
===>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build    F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

Press Enter. A message at the bottom of the panel indicates that the model object has been added to your session list.

**Note:** Alternatively you can press F4 instead of Enter to prompt option 11. This lets you enter the name of another alternate list or \*SELECT to select from a list of all model object lists.

## Accessing Other Model Lists

You can type the name of a model list in the List field on the Edit Model Object List panel to transfer quickly among model object lists defined for your model. You can also press F4 to prompt a selection list of all model object lists you have defined.

## Returning to Your Session List

Type the name of your session list in the List field as shown. Be sure to substitute the name of your session list.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      *All objects list for model MYMDL.

Type options, press Enter.
1=Select  2=Edit    3=Copy    4=Delete entry
5=Display 8=Details 9=Deselect 10=Action diagram

Opt  Object                Type  Attr  Owner
---  ---
1    *ALL values            CND   LST   Horse gender
2    *ALL values            CND   LST   Jockey gender
3    *ALL values            CND   LST   Going conditions
4    *ALL values            CND   LST   Entry Status
5    Change Course          FUN   DBF   Course
6    Change Horse           FUN   DBF   Horse
7    Change Jockey          FUN   DBF   Jockey

More...

Parameters or command
==>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build    F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys
'Course/Change Course/*FUN' added to list MYMDL/JAR.

```

Press Enter. Note that Change Course has been restored to your session list.

## Working with Model Object Lists

The Work with Model Lists panel lets you manage the model object lists in your model. You can access this panel from the CA 2E Designer (\*DSNR) Menu, the Display Services Menu, or by typing the Work with Model Lists (**YWRKMDLLST**) command on a command line

Type **YWRKMDLLST** on the command line.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select  2=Edit    3=Copy    4=Delete entry
5=Display 8=Details 9=Deselect 10=Action diagram

Opt  Object                Type  Attr  Owner
---  ---
1    *ALL values            CND   LST   Horse gender
2    *ALL values            CND   LST   Jockey gender
3    *ALL values            CND   LST   Going conditions
4    *ALL values            CND   LST   Entry Status
5    Change Course          FUN   DBF   Course
6    Change Horse           FUN   DBF   Horse
7    Change Jockey          FUN   DBF   Jockey

More...

Parameters or command
==> YWRKMDLLST
F3=Exit  F4=Prompt  F5=Refresh  F6=Build    F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

Press Enter. The Work with Model Lists panel displays showing all the model lists defined for your model. In your case, only your session list will be shown. The following is an example of how this panel might look in a working model.

```

Work with Model Lists

Model . . . : MYMDL
List . . . : _____ <-Position

Type options, press Enter.
2=Edit      3=Copy      4=Remove      5=Display
8=List details  9=Clear list  10=Execute list  13=Change description

Opt  List name  List description
--
1  AP          Accounts Payable
--
2  AR          Accounts Receivable
--
3  COMMANDS    Default model list for commands
--
4  EDITKEY     Model objects needing edit - Course code change
--
5  EDTLST      Model objects needing edit to fix PR295
--
6  FUNCTIONS   Changed functions for AP change
--
7  GENERATE    Model objects to generate - Course code change
--
8  JAR         List JAR in MYMDL created by user JAR.
--
9  PMH         List PMH in MYMDL created by user PMH.
--
10 PR3049      Changed model objects for PR3049
--
More...

F3=Exit  F5=Refresh  F6=Build  F9=Command line  F11=Alt view  F12=Cancel
F10=Create empty list  F23=More options

```

For any list displayed you can do the following:

- View the list entries
- Edit the list
- Clear all entries from the list
- Copy the list to another list
- Change the list's descriptive text
- Remove the list

If you type option **2** in the Subfile selector of any list, the Edit Model Object List panel displays for the selected list. As a result, this panel provides another method of entering your model.

Press F3 to return to the Edit Model Object List panel for your session list.

## Function Versioning

This topic describes how to create and test a version of a function.

**Note:** You can create versions for both functions (FUN) and messages (MSG); however, this topic discusses only function versions.

New terms introduced

- Version
- Group
- Current
- Redirection

New panel introduced

- Work with Versions

New commands introduced

- Create Model Version
- Compare Model Objects
- Redirect Model Object

## Objectives

To create a version of the Edit Horse function, edit its action diagram, test the change, and replace the original function with the new version.

## Overview of Versions

A *version* of a function is similar to a copy of the function. The major difference is that a copy is completely independent of the original function. Although a version is also a separate model object, CA 2E maintains an internal link between a function and its versions. A function can have an unlimited number of versions.

A function and its versions are known as a *group*. The interactive Work with Versions panel lets you view and work with a group of versions.

In any group of versions, one of the versions in the group may be *current*. The current version is the version that is active in the model; in other words, it is the function that is referenced by other objects in the model and that appears on CA 2E editing panels.

## Benefits of Versions

Three benefits of using versions are:

- You can test changes on a version of a function without interfering with the functionality of the existing model.
- When you finish testing a new version of a function and make it active (current) in the model, the original function remains unchanged and can easily be made active again if needed.
- Only the currently active version of a function is displayed on CA 2E editing panels. As a result, the panels are not cluttered with inactive versions.

## A Reason NOT to Use Versions

When you make a version of a function current in the model, CA 2E globally changes all the model objects that referenced the original function to reference the version instead. If not all of the referencing model objects need the changed functionality, you should create a new function rather than a version. After updating and testing the new function, you would then need to update references to the new function manually.

## Using Versions to Update an Existing Function

Following are the steps needed to update and test changes for a working external function using CA 2E's versioning feature. You will use this process to add an action to the action diagram for the Edit Horse function.

1. Create a version of the function you want to change.
2. Edit the version.
3. For an external function, generate the source and create the program object for the version.  
  
**Note:** Internal functions and messages do not result in separate program objects. As a result, the testing process for them is more complex and beyond the scope of this tutorial. Refer to the CA 2E module, *Generating and Implementing Applications*, for more information.
4. Test the version by calling the program object.
5. When you are satisfied that the version works properly, make the version current.
6. If errors occur, make the original function current again.

## Accessing the Session List

The Edit Model Object List panel should be displayed on your screen for your session list. If it is not, from the Designer Menu, type **4** on the Selection line as shown to select the Edit Session List (changed objects) option.

```

DSNR                      2E Designer (*DSNR) Menu
Level . : 1
System: 2EDV1

Select one of the following:

Enter Model      1. Edit Database Relations
                 2. Services Menu
                 3. Edit Default Model Object List
                 4. Edit Session List (changed objects)
                 5. Work with Model Objects
                 6. Load model and display command line
                 8. Work with Model Object Lists
                 9. Change to work with another model

Open Access:    ? 10. Change Open Access Model Value
enter with *HO 11. Edit Database Relations
                12. Services Menu

Selection or command
==> 4

F3=Exit  F6=Messages  F9=Prev. request  F10=Cmd Entry  F14=Submitted jobs
Maximum capability to access model MYMDL is *DSLK.

```

Press Enter to display the Edit Model Object List panel for your session list.

```

Edit Model Object List

Model . : MYMDL
List . . : JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select      2=Edit      3=Copy      4=Delete entry
5=Display     8=Details   9=Deselect 10=Action diagram

Opt  Object      Type  Attr  Owner
---  ---
1    *ALL values  CND   LST   Horse gender
2    *ALL values  CND   LST   Jockey gender
3    *ALL values  CND   LST   Going conditions
4    *ALL values  CND   LST   Entry Status
5    Change Course  FUN   DBF   Course
6    Change Horse   FUN   DBF   Horse
7    Change Jockey  FUN   DBF   Jockey

Parameters or command
==>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

## Positioning to the Edit Horse

Since you are going to create a version of the Edit Horse function, you first need to locate it. One way to do so is by using the F7 command key, which lets you position and sort entries for the displayed model object list.

Press F7 to display the Position the List window. There are several ways in which you can specify information on this window in order to locate the function you need. For example,

- If you know the name of the function, simply type the name in the Object field
- If you know the implementation name of the function (the program name), type the name in the Imp. Name field

- If you know the name of the file that owns the function, type the name of the file in the Owner field and scroll down to locate the function

Note that you can type partial names in the Object, Owner, and Imp. Name fields on the Position the List window; for example, if you type **H** or **H\*** for the Object field, the list will be positioned at the first object name that begins with the letter H.

Because you know the function name, type **Edit Horse** in the Object field as shown.

A rectangular window with a thin black border. In the top-left corner, there is a small square icon containing a red 'X'. The rest of the window is empty.

Press Enter.

## Creating a Version for Edit Horse

To create a version you first need to access the Work with Versions panel for the selected function. Press F23 twice to view additional Subfile selector options. The appropriate option is 19.

Type **19** in the Subfile selector for the Edit Horse function.



```

Edit Model Object List

Model . . . : MYMDL
List . . . : JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
18=Device structure   19=Work with versions   20=Access path
21=Narrative/object   22=Narrative/owner       23=STRSEU

Opt  Object              Type  Alt  Owner
18  Edit Horse           FUN   RPG   Horse
   Edit Jockey          FUN   RPG   Jockey
   Edit Race            FUN   RPG   Race
   Entry number         FLD   CDE
   Entry Status         FLD   STS
   Female               CND   UAL   Jockey gender
   Finished             CND   UAL   Entry Status
More...

Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh   F6=Build   F7=Position to
F8=Reverse retrieve   F9=Retrieve   F23=More options   F24=More keys

```

Press Enter.

## Working with Versions

The Work with Versions panel provides a set of options for working with a group of versions. Use it to perform tasks on versions, such as,

- Create a new version
- Edit
- Delete
- Display
- View detail
- Generate
- Impact analysis

The versions are displayed on this panel in reverse chronological order; in other words, the most recently created version appears at the top of the list of entries. The current version is shown highlighted with an \* to the right of the Subfile selector, and the Status column contains the word Current. Since you have not yet created versions for Edit Horse, only one entry is listed.

To create a new version for the Edit Horse function, type **3** in the Subfile selector.

Work with Versions				
Type options, press Enter.				
2=Edit	3=Create version	4=Delete object	5=Display	8=Details
10=Action diagram		12=Resolve conflicts		13=Parameters
Opt	Object	Version Type	Implementation Name	Status
<u>3</u>	* Edit Horse	Development	HYAEEFR	Current
F3=Exit   F5=Refresh   F11=Alt view   F12=Cancel   F23=More options <span style="float: right;">Bottom</span>				

Press Enter.

## Naming the Version

The prompt panel for the Create Model Version (YCRTMDLVSN) command displays. By default, CA 2E automatically generates a name for the version based on the name of the original function. The automatically generated name consists of the original function name followed by seven digits to ensure uniqueness; for example, Edit Horse1101481. In addition, by default, the original function retains its name and remains current (active).

In this example, you will specify a version name and accept the other defaults. Type **Edit Horse - Version 1** in the To model object name field.

Create Model Version (YCRTMDLVSN)		
Type choices, press Enter.		
From model object name:		
Object owner . . . . .	> Horse	Character value...
Object name . . . . .	> 'Edit Horse'	Character value
Object type . . . . .	> *FUN	*FUN, *MSG
To model object name . . . . .	<u>Edit Horse - Version 1</u>	
Make model object current . . .	<u>*NO</u>	*NO, *YES
Transfer model object name . . .	<u>*NO</u>	*NO, *YES
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display F24=More keys <span style="float: right;">Bottom</span>		

Press Enter. Note the messages at the bottom of the panel as CA 2E creates the new version.

### Implementation Name for the New Version


The new version you just created, Edit Horse - Version 1, now appears at the top of the list on the Work with Versions panel. Note that a new implementation name has been assigned to the version; in other words, the version is a separate object in the model. However, since the version is not current, you will not see it on CA 2E editing panels.

### Editing the New Function

You are now ready to edit the action diagram for Edit Horse - Version 1 in order to insert a call to an information message function. Type **10** in the Subfile selector.

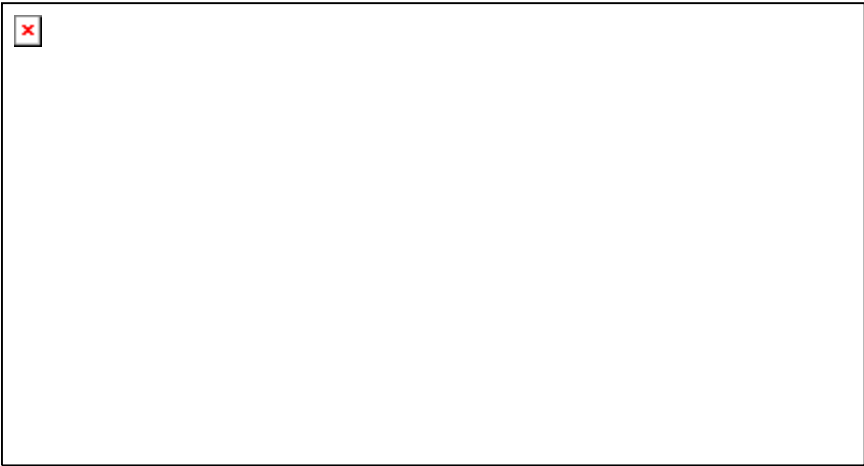
Work with Versions

Type options, press Enter.  
2=Edit 3=Create version 4>Delete object 5=Display 8=Details  
10=Action diagram 12=Resolve conflicts 13=Parameters

Opt	Object	Version Type	Implementation Name	Status
10	Edit Horse - Version 1	Development	MYAPEFR	
	* Edit Horse	Development	MYAEFR	Current

Bottom  
F3=Exit F5=Refresh F11=Alt view F12=Cancel F23=More options  
Model version 'Horse/Edit Horse/\*FUN' copied to version 'Edit Horse - Versio

Press Enter.



Press F5 to display the user exit points and select the user point to which you added logic to call the Display Racing results function for a selected horse.

Type **Z** against the Validate subfile record relations user point.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse - Version 1

___ > Edit ACTION DIAGRAM EXIT POINTS F3=Exit SEL:X,2-Select.
___ .-- USER: Initialize program
___ .-- USER: Initialize subfile header <--
___ .-- USER: Initialize subfile record (existing record)
___ .-- USER: Initialize subfile record (new record)
___ .-- CALC: Subfile control function fields <--
___ .-- PG USER: Validate subfile control
___ .-- USER: Validate subfile record fields
___ .-- CALC: Subfile record function fields
___ .-- Z USER: Validate subfile record relations + <<<
___ .--
___ .-- Process response <--
___ .-- ENDWHILE
___ .-- ENDWHILE
___ .-- Closedown <--
___ .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

## Inserting a Message Function

The objective is to add an information message to the Edit Horse function on return from the Display Racing results function. You will use the IMF command to insert the message. This command provides a shortcut for inserting and prompting a message function in one step. Remember, to view a list of available commands, type **?** in the Subfile selector.

Type **IMF** in the Subfile selector as shown to insert an information message function after calling the Display Racing results function.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse - Version 1

___ > USER: Validate subfile record relations <<<
___ .-- <<<
___ .-- -CASE <<<
___ .-- -RCD.Dam Date of birth GE RCD.Date of birth <<<
___ .-- Send error message - 'Dam younger than horse' <<<
___ .-- -ENDCASE <<<
___ .-- -CASE <<<
___ .-- -RCD.Sire Date of birth GE RCD.Date of birth <<<
___ .-- Send error message - 'Sire younger than horse' <<<
___ .-- -ENDCASE <<<
___ .-- -CASE <<<
___ .-- -RCD.*SFLSEL is *Zoom#1 <<<
___ .-- Display Racing results - Race Entry * <<<
___ .-- PGM.*Defer confirm = CHD.Defer confirm <<<
___ IMF PGM.*Reload subfile = CHD.*YES <<<
___ .-- -ENDCASE <<<
___ .-- <<<

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```



```
EDIT FUNCTION PARAMETERS                               My model
```

```
Function name... : Racing results displayed   Type : Send information message
```

```
Received by file : *Messages                  Acpth:
```

	File/*FIELD	Access path/Field	Passed as	Seq
-	*FIELD	Horse name	FLD	
-				
-				
-				
-				
-				
-				
-				
-				
-				
-				

|  
Values

```
FLD: One parameter per field
```

```
RCD: One parameter for all fields
```

```
KEY: One parameter for key fields only
```

```
SEL: Z-Details (field selection).
```

```
F3=Exit F5=Rload F22=File Locks
```

Press Enter.

Press F3 to return to the Edit Message Functions panel.

## Editing the Message Text

Next you will edit the message text to include the value of the Horse name parameter. Remember that, by default, the name of a message function and its associated text are the same.

Type **Z** in the Subfile selector for the Racing results displayed message function.

```

EDIT MESSAGE FUNCTIONS                               My model
File . . : *Messages                             Default msg file. . : QUSRMSG
                                                Generation library. : HYGEM

Message _____ <== Position

? Message                                     Type   Msgid      Ovr Msgf
Z Racing results displayed                 INF    USR0013

_____  
_____  
_____  
_____  
_____  
_____  
_____  
_____  
_____
SEL: Z=Details, P=Param, N=Harr, D>Delete, C=Copy, L=Locks, U=Usage, X>Select.  
F3=Exit F5=Reload F7=Change seq. F9=Add message F21=Convert messages
```

Press Enter to display the Edit Message Function Details panel.

## Using a Substitution Variable

Note the &1 in the No. column; this is the substitution variable that represents the Horse name parameter. You will use this substitution variable to include the selected horse's name in the text of the message. Recall that you used this process earlier in the tutorial when you defined error messages for the condition that compared a horse's date of birth with those of its parents.

The original message text is "Racing results displayed." Add "for &1" to the end of this text.

```

EDIT MESSAGE DETAILS                               My model
File name . . . : *Messages

Message . . . . : Racing results displayed
Message type. . . : INF (INF,ERR,STS,CMP)
Message id. . . . : USR0013      Override message file. :      *LIBL
                                   Default message file . :  QUSRMSG  MYGEN

Severity. . . . . : 20
Message text: Racing results displayed for &1

```

---

```

Parameters . . :
No.  Field                Type  Length
  &1  Horse name           TXT    25

```

F3=Exit F7=Second level text F8=Change name

Press Enter to return to the Edit Message Functions panel.

## Selecting the New Message

You still need to select this message function for inclusion in the Edit Horse function. Type **X** in the Subfile selector.

```
EDIT MESSAGE FUNCTIONS
```

File . . : *Messages  Message _____  ? Message _____ X Racing results displayed  _____ _____ _____ _____ _____ _____ _____ _____	<b>My model</b> Default msg file. . : QUSRMSG Generation library. : MYGEN  <== Position  Type Msgid Ovr Msgr INH USR0013  _____ _____ _____ _____ _____ _____ _____ _____
--	---

+  
SEL: Z=Details, P=Param, H=Harrt, D>Delete, C=Copy, L=Locks, U=Usage, X>Select.  
F3=Exit F5=Rload F7=Change seq. F9>Add message F21=Convert messages





```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Edit Horse - Version 1

___ > USER: Validate subfile record relations
___ .
___ . -CASE <<<
___ . -RCD.Dam Date of birth GE RCD.Date of birth <<<
___ . Send error message - 'Dam younger than horse' <<<
___ . -ENDCASE <<<
___ . -CASE <<<
___ . -RCD.Sire Date of birth GE RCD.Date of birth <<<
___ . Send error message - 'Sire younger than horse' <<<
___ . -ENDCASE <<<
___ . -CASE <<<
___ . -RCD.*SFLSEL is *Zoom#1 <<<
___ . Display Racing results - Race Entry * <<<
___ . PGM.*Defer confirm = CHD.Defer confirm <<<
___ . PGM.*Reload subfile = CHD.*YES <<<
___ . Send information message - 'Racing results displayed' <<<
___ . -ENDCASE <<<
___ .
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Note that the call to the information message function, Racing results displayed, has been inserted in the action diagram.

## Returning to the Action Diagram

Press F13 to exit and display the Exit Function Definition panel.

```

EXIT FUNCTION DEFINITION      My model

Type choices, press Enter.

Change/create function. . . . Y Y=Yes, N=No

Function name . . . . . Edit Horse - Version 1 Name
Access path name. . . . . Retrieval index Name
File name . . . . . Horse Name
Function type . . . . . Edit file

Print function. . . . . N Y=Yes, N=No

Return to editing . . . . . N Y=Yes, N=No

Submit generation . . . . . N Y=Yes, N=No

F5=Refresh F12=Cancel F15=Open Functions

```

Press Enter to accept the defaults and return to the Work with Versions panel.

Press F3 to return to the Edit Model Object List panel. Note that a **0** displays to the right of the Subfile selector of the Edit Horse - Version 1 list entry to indicate that it is a non-current version.

## Submitting the Function Version for Generation

You can submit generation requests and process job lists directly from the Edit Model Object List panel. Press F23 to see additional Subfile selector options. The appropriate option for submitting access paths and functions for batch generation is 14.

Type **14** in the Subfile selector for the Edit Horse - Version 1 function.

Edit Model Object List				
Model . . : MYMDL				
List . . . JAR List JAR in MYMDL created by user JAR.				
Type options, press Enter.				
11=Add to alternate list		13=Parameters	14=GEN batch	
15=GEN interactive		16=Y2CALL	17=Device design	
Opt	Object	Type	Atr	Owner
—	Delete Race Entry	FUN	DBF	Race Entry
—	Display Racing results	FUN	RPG	Race Entry
—	Disqualified	CND	VAL	Entry Status
—	Distance	FLD	QTY	
—	Edit Course	FUN	RPG	Course
—	Edit Horse	FUN	RPG	Horse
<u>14</u> 0	Edit Horse - Version 1	FUN	RPG	Horse
More...				
Parameters or command				
===>				
F3=Exit	F4=Prompt	F5=Refresh	F6=Build	F7=Position to
F8=Reverse retrieve	F9=Retrieve	F23=More options	F24=More keys	

Press Enter. A message at the bottom of the panel indicates that the request for batch generation has been submitted.

## Viewing Job List Commands

You also have access to a set of job list commands from the Edit Model Object List panel. This includes the Submit Model Create Requests (YSBMMDLCRT) command, which you previously accessed from the Display Services Menu. Press F24 twice to view additional command keys. The appropriate command key is F19.

Press F19 to display the Job List Commands Menu. Type **1** on the Selection line to select the YSBMMDLCRT command.

```

Edit Model Object List
-----
Job List Commands Menu
Model : MYMDL

Select one of the following:

1. YSBMMDLCRT Submit model create requests
2. YBLDJOBLS  Build a job list
3. YDSPJOBLS  Display a job list
4. YCVIMDLST  Convert a model list
5. YCUTJOBLS  Convert a job list
6. YCHKJOBLE  Check job list entries
7. YCRTJOBLE  Create a job list entry

Selection or command
==> 1
F3=Exit  F4=Prompt  F8=Rev retrieve  F9=Retrieve  F12=Cancel

F19=Job list menu  F20=Usages  F21=Print  F22=References
F23=More options  F24=More keys

```

Press Enter.

When the prompt screen for the YSBMMDLCRT command displays, press Enter again to accept the defaults and display the job list.

```

SUBMIT MODEL GENERATIONS & CREATES. My model

GENLIB: MYGEN

? Member      Type Act Status  Text
■ MYAPEFRD    DSPF GEN      Edit Horse - Version 1  Edit file
- MYAPEFRH    PHL  GEN      Edit Horse - Version 1  Edit file
- MYAPEFR     RPG  GEN      Edit Horse - Version 1  Edit file

SEL: G-Rqs GEN, C-Rqs CRT, E-STRSEU, D-Drop, JOB(1-DSP, 4-HLD, 6-RLS, 9-CNL)
F3=Exit F5=Reload F6=Msgs F8=Submitted jobs F9=Command line ENTER=Submit

```

Press Enter twice to submit the request to the job queue. Check the status of the job as described in the *Generating, Compiling, and Executing* chapter. Once the generation and compilation completes successfully, you can test Edit Horse - Version 1 to see if the new message displays.

Press F3 twice to return to the Edit Model Object List panel.

## Displaying Alternate Views

If you do not know the implementation name for Edit Horse - Version 1, you can use the F11 command key to view additional information for the model objects displayed. Five alternate views are available; however, this tutorial discusses only the first two. You can press F11 five times to see all five views.

Press F11 to display the alternate view showing the implementation name. Note the name that corresponds to Edit Horse - Version 1, MYAPEFR on the panel below.

## Testing Edit Horse - Version 1

On the Command line, type the following to test Edit Horse - Version 1. Be sure to substitute the correct implementation name.

CALL MYAPEFR ``

Edit Model Object List						
Model . . : MYMDL						
List . . . JAR List JAR in MYMDL created by user JAR.						
Type options, press Enter.						
1=Select		2=Edit		3=Copy		4=Delete entry
5=Display		8=Details		9=Deselect		10=Action diagram
Opt	Object	Implementation Name	Generation Date	Time	Type	Fun/Msg
—	Edit Horse	MYAEEFR	08/31/95	11:15:04	EDTFIL	
— 0	Edit Horse - Version 1	MYAPEFR	09/01/95	10:33:02	EDTFIL	
—	Edit Jockey	MYAIEFR	08/31/95	10:33:57	EDTFIL	
—	Edit Race	MYAKEFR	08/31/95	10:35:06	EDTFIL	
—	Entry number	ACCD				
—	Entry Status	ACST				
—	Female					
						More...
Parameters on command						
====> CALL MYAPEFR ``						
F3=Exit	F4=Prompt	F5=Refresh	F6=Build	F7=Position to		
F8=Reverse retrieve	F9=Retrieve	F23=More options	F24=More keys			

Press Enter.

The interactive panel for Edit Horse - Version 1 should display showing the data you entered earlier for the Edit Horse function.

Following are the steps needed to test your change.

1. Type **/** in the Subfile selector for any horse.
2. Press F10 to access the action bar.
3. Type **S** to access the Selector Choice menu.
4. Type **1** next to the Display Racing Results action.
5. Press Enter.

**Note:** Because you will not enter data for the RACE and RACE ENTRY files until the *Advanced Functions* chapter, Display Racing Results will have no data to display; however, you should see the new information message on return to Edit Horse - Version 1.

- 6. Press F3 to return to Edit Horse - Version 1. The new message should display at the bottom of the panel.

Press F3 to return to the Edit Model Object List panel.

### Comparing Two Versions of a Function

You can compare two versions of a function using the Compare Model Objects (YCMPMDLOBJ) command. This command is useful for solving problems that arise while testing a new version and for identifying changes made to one version of a function for retrofitting to another version.

**Note:** You can also use this command to compare two functions or two messages. In other words, the model objects being compared need not be versions.

To compare two versions of a function, type YCMPMDLOBJ on the Command line.

Edit Model Object List

Model . . : MYMDL  
List . . . JAR

List JAR in MYMDL created by user JAR.

Type options, press Enter.  
1=Select    2=Edit    3=Copy    4=Delete entry  
5=Display    8=Details    9=Deselect    10=Action diagram

Opt	Object	Implementation Name	----Generation---	Fun/Msg
			Date    Time	Type
— 0	Edit Horse	MYAEEFR	08/31/95 11:15:04	EDTFIL
—	Edit Horse - Version 1	MYAPEFR	09/01/95 10:33:02	EDTFIL
—	Edit Jockey	MYAIEFR	08/31/95 10:33:57	EDTFIL
—	Edit Race	MYAKEFR	08/31/95 10:35:06	EDTFIL
—	Entry number	ACCD		
—	Entry Status	ACST		
—	Female			

More...

Parameters on command  
==> YCMPMDLOBJ

F3=Exit    F4=Prompt    F5=Refresh    F6=Build    F7=Position to  
F8=Reverse retrieve    F9=Retrieve    F23=More options    F24=More keys

### Entering Parameters for YCMPMDLOBJ

Press F4 to prompt the command. Enter parameters to identify the functions to be compared; namely, enter the Object owner, name, and type of both functions. In this case, type **HORSE, Edit Horse, \*FUN, HORSE, Edit Horse - Version 1**, and **\*FUN** as shown. Accept the defaults for the other parameters.

```

Compare Model Objects (YCHPMDLOBJ)

Type choices, press Enter.

Model object name:
Object owner . . . . . > HORSE
Object name--*generic . . . . > Edit Horse
Object type . . . . . > *FUN          *FUN, *MSG, *FIL
Model object surrogate . . . . > *OBJNAM1      Number, *OBJNAM1
From model library . . . . . > *MDLLIB        Name, *MDLLIB
Model object name:
Object owner . . . . . > HORSE
Object name--*generic . . . . > Edit Horse - Version 1
Object type . . . . . > *FUN          *OBJNAM1, *FUN, *MSG, *FIL
Model object surrogate . . . . > *OBJNAM2      Number, *OBJNAM2, *PRD...
To model library . . . . . > *MDLLIB1       Name, *MDLLIB1
Ignore case differences . . . . > *NO          *NO, *YES
Print headers . . . . . > *NO          *NO, *YES

F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F13=How to use this display  F24=More keys

```

Press Enter.

When the command completes, it returns automatically to the Edit Model Object List panel. If differences were found a message displays at the bottom of the panel.

## Viewing Differences between Versions

Use the i OS Work with Spool Files (WRKSPLF) command to view the differences. Type WRKSPLF on the Command line.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select  2=Edit  3=Copy  4=Delete entry
5=Display 8=Details 9=Deselect 10=Action diagram

Opt  Object  Implementation  ---Generation---  Fun/Msg
Name  Date  Time  Type
---  ---  ---  ---  ---
0  Edit Horse  MYAEFR  08/31/95  11:15:04  EDTFIL
  Edit Horse - Version 1  MYAEFR  09/01/95  10:33:02  EDTFIL
  Edit Jockey  MYAEFR  08/31/95  10:33:57  EDTFIL
  Edit Race  MYAEFR  08/31/95  10:35:06  EDTFIL
  Entry number  ACCT
  Entry Status  ACCT
  Female

Parameters or command
==> WRKSPLF
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys
HORSE/EDIT HORSE/*FUN in MYMDL differs from HORSE/Edit Horse - Version 1/*FU

```

Press Enter. Use Subfile selector option 5 to display the job named YCHPMDLOB\$. The differences are shown as follows.

Display Spooled File

File . . . . . : YCMPMDLOB#Page/Line 1/3  
Control . . . . :Columns 1 - 78  
Find . . . . . :  
\*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...  
2EDV1: Development YCMPSRC - Source Compare.  
OPTIONS:- FILE1 : YSRCF.QTEMP.FUN1100247 (HORSE/Edit Horse/\*FUN -  
FILE2 : YSRCF.QTEMP.FUN1101305 (HORSE/Edit Horse - Versi  
MATCHSIZE: 00000003 PRTBEFORE: 00001 PRTAFTER : 00001  
LIST : \*NOLINES IGNSPACE : \*NO IGNCASE : \*NO  
\*----- Files differ -----\*  
Lines 00002/00003 in file : YSRCF.QTEMP.FUN1100247 ← Edit Horse  
>> PGM.\*Reload subfile = CND.\*YES  
>> -ENDCASE  
Lines 00002/00005 in file : YSRCF.QTEMP.FUN1101305 ← Edit Horse  
>> PGM.\*Reload subfile = CND.\*YES - Version 1  
\*\*\* >> Send information message - 'Racing results displayed'  
\*\*\* >> I Horse name = RCD.Horse name  
>> -ENDCASE  
Files YSRCF.QTEMP.FUN1100247 (HORSE/Edit Horse/\*FUN - MYMDL  
and YSRCF.QTEMP.FUN1101305 (HORSE/Edit Horse - Version 1/\*FUN - MYM  
More...  
F3=Exit F12=Cancel F19=Left F20=Right F24=More keys

Press F3 twice to return to the Edit Model Object List panel.

Making the Edit Horse - Version 1 Current

Once you have tested Edit Horse - Version 1 and are satisfied that both the new and old functionality work correctly, you can replace the original Edit Horse function in the model with the new version by making the version current.

First display the Work with Version panel by typing **19** in the Subfile selector for either version of the Edit Horse function.

Edit Model Object List

Model . . : MYMDL  
List . . . JAR List JAR in MYMDL created by user JAR.  
Type options, press Enter.  
18=Device structure 19=Work with versions 20=Access path  
21=Narrative/object 22=Narrative/owner 23=STRSEU  

Opt	Object	Implementation Name	Generation Date	Time	Fun/Msg Type
19 0	Edit Horse	MYAEEFR	08/31/95	11:15:04	EDTFIL
	Edit Horse - Version 1	MYAPEFR	09/01/95	10:33:02	EDTFIL
	Edit Jockey	MYAIEFR	08/31/95	10:33:57	EDTFIL
	Edit Race	MYAKEFR	08/31/95	10:35:06	EDTFIL
	Entry number	ACCD			
	Entry Status	ACST			
	Female				

More...

Parameters or command  
==>  
F3=Exit F4=Prompt F5=Refresh F6=Build F7=Position to  
F8=Reverse retrieve F9=Retrieve F23=More options F24=More keys

Press Enter.

On the Work with Version panel, press F23 to view additional options. The appropriate option is 26, Make current. Type **26** in the Subfile selector for Edit Horse - Version 1.

Work with Versions					
Type options, press Enter.					
14/15=Generate batch/interactive		17=Device design	18=Structure		
20=Access path 21/22=Narrative		23=STRSEU	26=Make current		
Opt	Object	Version Type	Implementation Name	Status	
<u>26</u>	Edit Horse - Version 1	Development	MYAPEFR		
— *	Edit Horse	Development	MYAEEFR	Current	
<div style="text-align: right;">Bottom</div> <div> F3=Exit   F5=Refresh   F11=Alt view   F12=Cancel   F23=More options </div>					

Press Enter.

## Naming the New Current Version

The prompt screen for the Redirect Model Object (YRDRMDLOBJ) command displays.

When you make a version current, by default, the object names of the original and new current versions are exchanged; this is indicated by \*YES in the Transfer model object name parameter. The implementation names of the two versions are always exchanged. To avoid misunderstanding, be sure to inform other developers when you make a new version current.

Redirect Model Object (YRDRMDLOBJ)		
Type choices, press Enter.		
From model object name:		
Object owner . . . . .	> *CURRENT	Character value...
Object name . . . . .		Character value
Object type . . . . .		*FUN, *MSG
To model object name:		
Object owner . . . . .	> 'Horse'	Character value, *TOOBSJGT, .
Object name . . . . .	> 'Edit Horse - Version 1'	
Object type . . . . .	> '*FUN'	*FRMOBJNAM, *FUN, *MSG
Transfer model object name . . .	<u>*YES</u>	*NO, *YES
Change type . . . . .	<u>*PUBLIC</u>	*NONE, *PUBLIC, *PRIVATE...
<div style="text-align: right;">Bottom</div> <div> F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display  F24=More keys </div>		

Press Enter to accept the defaults.



**Note:** If other objects in the model reference Edit Horse, they are automatically changed to reference the new current version instead. For example, if another function called Edit Horse, CA 2E would change the function so that it calls the new current version of Edit Horse instead. This process is known as *redirection*. Messages display at the bottom of the panel during redirection.

The new current version now contains the call to the information message. Its object name is now Edit Horse and its implementation name is MYAEEFR.

### Regenerating the New Current Version

You need to generate and compile the new current version so the source and program object contain the new functionality. Type **14** in the Subfile selector as shown to submit a request for batch generation.

Work with Versions

Type options, press Enter.  
14/15=Generate batch/interactive    17=Device design    18=Structure  
20=Access path    21/22=Narrative    23=STRSEU    26=Make current

Opt	Object	Version Type	Implementation Name	Status
<b>14</b> *	<b>Edit Horse</b>	Development	MYAEEFR	Current
—	Edit Horse - Version 1	Development	MYAPEFR	

Bottom  
F3=Exit    F5=Refresh    F11=Alt view    F12=Cancel    F23=More options  
Usages of Model object 'Horse/Edit Horse/\*FUM' in library MYMDL redirected t

Press Enter to submit the request. Press F3 to return to the Edit Model Object List panel. The 0 to the right of the Subfile selector of the Edit Horse list entry indicates that it is a non-current version.

### Exercise

Submit the batch job using the procedure you used previously in this topic. Press F19 to display the Job List Commands menu and select the YSBMMDLCRT option. When you finish, return to the Edit Model Object List panel.

## Viewing Model Object Information for the Versions

Note that an 8 appears to the left of both the Edit Horse and Edit Horse - Version 1 list entries. This indicates that the list entry information does not match that of the corresponding model object. In this case, the object names and implementation names are mismatched since they were exchanged during the process of making Edit Horse - Version 1 current.

**Note:** This difference is allowed to occur so you can use your model object list as a historical record.

To view current details for a model object, use Subfile selector option 8. (Note that you can press F23 to see other options.) For example, type **8** against the Edit Horse list entry.

Edit Model Object List						
Model . . : MYMDL						
List . . . JAR List JAR in MYMDL created by user JAR.						
Type options, press Enter.						
18=Device structure		19=Work with versions		20=Access path		
21=Narrative/object		22=Narrative/owner		23=STRSEU		
Opt	Object	Implementation Name	Generation Date	Time	Fun/Msg Type	
8 0 8	Edit Horse	MYAEEFR	08/31/95	11:15:04	EDTFIL	
— 8	Edit Horse - Version 1	MYAPEFR	09/01/95	10:33:02	EDTFIL	
—	Edit Jockey	MYAIEFR	08/31/95	10:33:57	EDTFIL	
—	Edit Race	MYAKEFR	08/31/95	10:35:06	EDTFIL	
—	Entry number	ACCD				
—	Entry Status	ACST				
—	Female					
						More...
Parameters or command						
===>						
F3=Exit	F4=Prompt	F5=Refresh	F6=Build	F7=Position to		
F8=Reverse retrieve	F9=Retrieve	F23=More options	F24=More keys			

Press Enter.

Note that the Object Name, Edit Horse - Version 1, and the implementation name shown in the Source column, MYAPEFR, do not match the corresponding information shown on the Edit Model Object List panel for the Edit Horse list entry.

Display Model Object

Model : MYMDL

Object . . . . Edit Horse - Version 1

Owner . . . . Horse

Type . . . . FUN Attribute . . . . RPG Surrogate . 1100110

Copy name . . Edit Horse

Create date . 08/29/95

Create time . 16:57:40

Change date . 09/01/95

Change time . 11:38:25

Comp chg date. 09/01/95

Comp chg time. 11:38:25

Checkout date

Import date .

Import time .

Generate date 08/31/95

Generate time 11:15:04

Version type . . DEU

Current object . N

Change type . . . PUT

Change user . . . JAR

Action required . GEN

Checkout status

Checkout user . .

Import model . .

Import status . .

Function type . . EDTFIL

Impact processed N

List . . . .

Promotion .

Source

Type Text

MYAPEFR RPG Edit Horse - Version 1 Edit file

MYAPEFRD DSP Edit Horse - Version 1 Edit file

MYAPEFRH HLP Edit Horse - Version 1 Edit file

F5=Refresh F8=Change copy name F12=Cancel

Bottom

Press F12 to return to the Edit Model Object List panel.

Refreshing List Entries for the Versions

To update the two list entries to match the actual model objects, type **33** in the Subfile selector for both entries.

Edit Model Object List

Model . . : MYMDL

List . . . : JAR List JAR in MYMDL created by user JAR.

Type options, press Enter.

18=Device structure 19=Work with versions 20=Access path

21=Narrative/object 22=Narrative/owner 23=STRSEU

Opt	Object	Implementation Name	-----Generation---	Fun/Msg
			Date Time	Type
33 0 8	Edit Horse	MYAEEFR	08/31/95 11:15:04	EDTFIL
33 8	Edit Horse - Version 1	MYAPEFR	09/01/95 10:33:02	EDTFIL
—	Edit Jockey	MYAIEFR	08/31/95 10:33:57	EDTFIL
—	Edit Race	MYAKEFR	08/31/95 10:35:06	EDTFIL
—	Entry number	ACCD		
—	Entry Status	ACST		
—	Female			

More...

Parameters or command

==>

F3=Exit F4=Prompt F5=Refresh F6=Build F7=Position to

F8=Reverse retrieve F9=Retrieve F23=More options F24=More keys

Model object details displayed.

Press Enter.

**Note:** If many discrepancies exist between entries on a model object list and the actual model objects, you can refresh the entire list by pressing the F15 command key. This executes the Check a Model List (YCHKMDLLST) command.

The information for the list entries for the two versions now match that of the corresponding model objects.

```

                                Edit Model Object List

Model . . . : MYMDL
List . . . : JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
18=Device structure      19=Work with versions      20=Access path
21=Narrative/object      22=Narrative/owner       23=STRSEU

Opt   Object              Implementation  ---Generation--- Fun/Msg
---   ---                Name             Date       Time   Type
---   Edit Horse - Version 1 MYAPEFR      08/31/95  11:15:04 EDTFIL
---   Edit Horse           MYAEEFR      09/01/95  10:33:02 EDTFIL
---   Edit Jockey          MYAIEFR      08/31/95  10:33:57 EDTFIL
---   Edit Race            MYAKEFR      08/31/95  10:35:06 EDTFIL
---   Entry number         ACCD
---   Entry Status         ACST
---   Female

Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh   F6=Build   F7=Position to
F8=Reverse retrieve F9=Retrieve   F23=More options F24=More keys
More...

```

## Exercise

Use the Position the List panel to display all entries for your session list.

**Note:** Press F7, blank out all the fields, and press Enter. CA 2E automatically repositions your session list at the top in order by Object Name and Object Type.

## Model Object Cross References

This topic introduces the CA 2E model object cross reference panels.

New terms introduced

- Model object cross reference
- Expansion
- Usages
- Using objects
- References

New panels introduced

- Display Model Usages
- Display Model References

## Objectives

To use CA 2E's model object cross reference facilities, including the Display Model Usages panel and the Display Model References panel.

## Overview of Model Object Cross References

CA 2E's *model object cross reference* facilities consist of a set of commands and interactive panels that you can use to determine, for any model object, which other objects it references and which other objects it is used by.

The process of determining either usages or references for a model object is known as *expansion*. Using model object cross reference facilities, you can expand usages or references for a model object to any level.

You can display or print model object usages and references; you can also convert them to a model object list.

## Model Object Usages

*Usages* for a model object are all the model objects that use it. Usages are external to the model object and require the model object in order to be complete. A model object's usages are sometimes referred to as *using objects*.

Suppose you want to change the Mares access path. Before you do so, you need to determine which model objects use it. In this step you will use the Display Model Usages panel to display usages for the Mares access path.

## Accessing Your Session List

Start by accessing the Edit Model Object List panel for your session list. Use one of the methods you used earlier in this chapter.

First position the session list to the Mares access path. Press F7 to display the Position the List window. Type **ACP** for the Type field and **Mares** for the Object field.

**Edit Model Object List**

Position the List

Type . . . ACP    Object . . Mares    Imp. Name \_\_\_\_\_

Owner . . . \_\_\_\_\_

F12=Cancel

---

Opt	Object	Type	Atr	Owner
—	*ALL values	CND	LST	Horse gender
—	*ALL values	CND	LST	Jockey gender
—	*ALL values	CND	LST	Going conditions
—	*ALL values	CND	LST	Entry Status
—	Change Course	FUN	DBF	Course
—	Change Horse	FUN	DBF	Horse
—	Change Jockey	FUN	DBF	Jockey

More...

Parameters or command  
==>

F3=Exit    F4=Prompt    F5=Refresh    F6=Build    F7=Position to  
F8=Reverse retrieve    F9=Retrieve    F23=More options    F24=More keys

Press Enter. Your session list is now displayed in order by Object Type and Object Name beginning with the Mares access path.

Press F23 until option 91 is displayed. Option 91 displays usages for the selected object. Type **91** in the Subfile selector for the Mares access path.

**Edit Model Object List**

Model . . . : MYMDL  
List . . . : JAR    List JAR in MYMDL created by user JAR.

Type options, press Enter.  
34=Compare objects    81=References/object    82=References/owner  
91=Usages/object    92=Usages/owner    ^=Merge command

---

Opt	Object	Type	Atr	Owner
<b>91</b>	Mares	ACP	RTU	Horse
—	Physical file	ACP	PHY	Race Entry
—	Physical file	ACP	PHY	Race
—	Physical file	ACP	PHY	Course
—	Physical file	ACP	PHY	Horse
—	Physical file	ACP	PHY	Jockey
—	Races for a Horse	ACP	RSQ	Race Entry

More...

Parameters or command  
==>

F3=Exit    F4=Prompt    F5=Refresh    F6=Build    F7=Position to  
F8=Reverse retrieve    F9=Retrieve    F23=More options    F24=More keys

Press Enter. Notice the messages at the bottom of the panel as CA 2E expands model usages.

## Display Model Usages Panel

When all usages have been expanded, the Display Model Usages panel displays showing the first level of objects in the model that use the Mares access path. Notice the 001 in the Lvl column for each using object; this indicates that the object uses the Mares access path directly.

The Display Model Usages panel provides a variety of controls and filters including, recursion, scope, and positioning to help you analyze your model to determine the impact of proposed changes.

**Note:** Most of the options provided on the Display Model Usages panel are beyond the scope of this tutorial. However, you can use the online help if you would like to experiment further. Or, you can refer to the CA 2E guide, *Generating and Implementing Applications*.

Gen objs : 5  
Total : 5  
Object : Mares  
Type : ACP  
Scope : \*NEXT  
Object :

**Display Model Usages**  
Attribute : RTU  
Filter : \*ANY  
Type :  
2=Edit  
13=Parms  
3=Copy  
14=GEN batch  
4=Delete object  
5=Display  
8=Details  
10=Action diagram  
15=GEN interactive  
16=Y2CALL

Model : MYMDL  
Level : 001  
Owner : Horse  
Exclude system objs : \*YES  
Current objects only : \*YES  
Reason : \*FIRST

Opt	Object	Typ	Atr	Owner	Lvl	Reason
<input checked="" type="checkbox"/>	Edit Horse	FUN	RPG	Horse	001	*REFACP
<input type="checkbox"/>	Select Horse	FUN	RPG	Horse	001	*REFACP
<input type="checkbox"/>	Select Mares	FUN	RPG	Horse	001	*FUNPAR
<input type="checkbox"/>	Mares	ACP	RTU	Horse	000	*OBJECT

F3=Exit  
F16=Build model list

F5=Refresh  
F21=Print list

F9=Command line  
F12=Previous

**Bottom**  
F15=Top level  
F23=More options

### Usage Reason

The Reason column shows how each of the listed model objects use the Mares access path. For example,

- \*REFACP - The Edit Horse and Select Horse functions use Mares as a referenced access path.
- \*FUNPAR - The Select Mares function uses Mares as the definition of a function parameter.
- \*OBJECT - This identifies the model object for which you requested usages. This object is included on the list of usages so you can use the Subfile selection options on the original model object.

The model objects that use the Mares access path are all external functions that will at least need to be regenerated and compiled if you change the Mares access path. However, if your proposed change requires relations to be added to or removed from the Mares access path, you might also need to edit these functions.

In this model no other model objects use the using objects for Mares. You can test whether other objects use the objects displayed by typing **91** in the Subfile selector for any of the using objects.

## Exercise

Type **91** against Edit Horse and press Enter to display its usages. The Display Model Usages panel redisplay showing only the model object itself. This indicates that no other model objects use Edit Horse; in other words, it has no usages.

Display Model Usages									
Gen objs :	1					Model :	MYMDL		
Total :	1					Level :	002		
Object :	Edit Horse					Owner :	Horse		
Type :	FUN	Attribute :	RPG			Exclude system objs :	*YES		
Scope :	*NEXT	Filter :	*ANY			Current objects only :	*YES		
Object :		Type :				Reason :	*FIRST		
2=Edit    3=Copy    4=Delete object    5=Display    6=Details    10=Action diagram 13=Parms    14=GEN batch    15=GEN interactive    16=Y2CALL									
<b>Opt</b>	<b>Object</b>	<b>Typ</b>	<b>Atr</b>	<b>Owner</b>	<b>Lvl</b>	<b>Reason</b>			
<input type="checkbox"/>	Edit Horse	FUN	RPG	Horse	000	*OBJECT			
<b>Bottom</b>									
F3=Exit		F5=Refresh		F9=Command line		F12=Previous		F15=Top level	
F16=Build model list		F21=Print list						F23=More options	

## Using Usage Levels

Notice that the Level number in the upper right corner has changed to 002. This indicates the number of times you have expanded usages beginning with the original model object.

For a more complex example, a model object may have many levels of using objects. The value of \*NEXT in the Scope field lets you step through the expansion of usages one level and one model object at a time. In other words, you can type **91** for any using objects to determine its usage. The Level shown at the upper right of the screen shows the current level. The F3 key lets you back up one level at a time; the F15 key returns to the top level.

Note that you can change the Scope field to \*NOMAX to expand all usages for a selected model object. However, for working models this can require significant processing time.

## Exiting Model Usages

Press F3 twice to return to the Edit Model Object List panel.

Press F7 to display the Position the List window. Blank out all fields and press Enter to reposition the session list and display all entries.



## Model Object References

*References* for a model object are the model objects it refers to internally. In other words, references are the model objects the referring model object requires in order to be complete or to exist. For example, Display Racing results and Change Horse are references of the Edit Horse function.

This step shows how to access the Display Model References panel and how you can use it to solve problems in program applications.

## Accessing the \*ALLOBJ List

Start by accessing the Edit Model Object List panel for the All Objects list (\*ALLOBJ). Since your session list is currently displayed, type **\*ALLOBJ** for the List field at the top of the panel and press Enter.

## Positioning \*ALLOBJ to an Implementation Name

Suppose you only know the implementation name of the program in which an error occurred; for example, MYAEFR (Edit Horse).

From the Edit Model Object List press F7. Type the implementation name of the program in the Imp. Name field as shown. Be sure to substitute the name corresponding to your Edit Horse function.

Edit Model Object List

Position the List

Type . . .      Object . . .      Imp. Name MYAEFR

Owner . . .

F12=Cancel

Opt	Object	Type	Atr	Owner
—	Mares	ACP	RTU	Horse
—	Physical file	ACP	PHY	Race Entry
—	Physical file	ACP	PHY	Race
—	Physical file	ACP	PHY	Course
—	Physical file	ACP	PHY	Horse
—	Physical file	ACP	PHY	Jockey
—	Races for a Horse	ACP	RSQ	Race Entry

More...

Parameters or command

====>

F3=Exit    F4=Prompt    F5=Refresh    F6=Build    F7=Position to

F8=Reverse retrieve    F9=Retrieve    F23=More options    F24=More keys

Press Enter.

The \*ALLOBJ list is now positioned at the Edit Horse function. Note the message at the bottom of the panel and that the alternate view showing the Implementation name displays automatically. You can change views by pressing F11.

## Displaying References for Edit Horse

Option 81 lets you display references for a selected model object. Type **81** in the Subfile selector for Edit Horse.

```

Edit Model Object List

Model : MYMDL
List : *ALLOBJ *All objects list for model MYMDL.

Type options, press Enter.
34=Compare objects      81=References/object      82=References/owner
91=Usages/object       92=Usages/owner        /=Merge command

Opt   Object              Implementation  ---Generation--- Fun/Msg
81    Edit Horse          MYAEEFR       09/01/95 10:33:02 EDTFIL
      Retrieval index     MYAEREL0      08/31/95 11:13:48
      Update index        MYAEREL1      08/31/95 11:13:44
      Mares               MYAEREL2      08/31/95 11:13:55
      Stallions           MYAEREL3      08/31/95 11:13:59
      Physical file        MYAEREF       08/31/95 11:13:41
      Retrieval index     MYAFREL0      08/31/95 10:30:09

Parameters or command
===>
F3=Exit   F4=Prompt   F5=Refresh   F6=Build     F7=Position to
F8=Reverse retrieve F9=Retrieve   F23=More options F24=More keys
Data displayed in Implementation Name order.
More...

```

Press Enter. Notice the messages at the bottom of the panel as CA 2E expands the model references.

## Display Model References Panel

When all references have been expanded, the Display Model References panel displays showing all objects in the model referenced by Edit Horse. This panel provides a variety of controls and filters including, recursion, scope, and positioning to help you analyze your model.

**Note:** Most of the options provided on the Display Model References panel are beyond the scope of this tutorial. However, you can use the online help if you would like to experiment further. Or, refer to the CA 2E module, *Generating and Implementing Applications*.

Display Model References					Model . . : MYMDL
Gen objs . . :	3				Level . . : 001
Total . . :	63				
Object . . :	Edit Horse			Owner . . : Horse	
Type . . :	FUN	Attribute . . :	RPG	Exclude system objs . .	*YES
Scope . . :	*NEXT	Filter . . :	*ANY	Current objects only . .	*YES
Object . . :			Type . . :		Reason . . : *FIRST
2=Edit 3=Copy 4=Delete object 5=Display 8=Details 10=Action diagram					
13=Parms 14=GEN batch 15=GEN interactive 16=Y2CALL					
<b>Opt Object</b>					
<b>Obj</b> Horse					
Type Atr Owner Lvl Reason					
— Horse					
— Change Horse					
— Create Horse					
— Delete Horse					
— Display Racing results					
— Edit Horse					
— Retrieval index					
— Dam name					
— Dam Date of birth					
F3=Exit F5=Refresh F9=Command line F12=Previous F15=Top level					
F16=Build model list F21=Print list F23=More options					

Displaying Only External Functions

Since you are using references to solve a problem in an application program, you only need to see external functions that are referenced in the Edit Horse action diagram.

Type **\*EXTFUN** for the Scope field and type **\*ACTION** for the Reason field.

Display Model References					Model . . : MYMDL
Gen objs . . :	3				Level . . : 001
Total . . :	63				
Object . . :	Edit Horse			Owner . . : Horse	
Type . . :	FUN	Attribute . . :	RPG	Exclude system objs . .	*YES
Scope . . :	*EXTFUN	Filter . . :	*ANY	Current objects only . .	*YES
Object . . :			Type . . :		Reason . . : *ACTION
2=Edit 3=Copy 4=Delete object 5=Display 8=Details 10=Action diagram					
13=Parms 14=GEN batch 15=GEN interactive 16=Y2CALL					
<b>Opt Object</b>					
<b>Obj</b> Horse					
Type Atr Owner Lvl Reason					
— Horse					
— Change Horse					
— Create Horse					
— Delete Horse					
— Display Racing results					
— Edit Horse					
— Retrieval index					
— Dam name					
— Dam Date of birth					
F3=Exit F5=Refresh F9=Command line F12=Previous F15=Top level					
F16=Build model list F21=Print list F23=More options					

Press Enter. Only the functions that make up the program in which the error occurred are now displayed. In a working model this list would generally be much longer. This is a useful starting place for the developers whose task it is to fix the problem.

Gen objs : 2				Display Model References		Model : MYMDL	
Total : 9						Level : 001	
Object : Edit Horse				Owner : Horse			
Type : FUN				Attribute : RPG		Exclude system objs : *YES	
Scope : *EXTFUN				Filter : *ANY		Current objects only : *YES	
						Reason : *ACTION	
2=Edit		3=Copy		4=Delete object		5=Display	
13=Parms		14=GEN batch		15=GEN interactive		8=Details	
						10=Action diagram	
						16=Y2CALL	
<b>Opt Object</b>				<b>Typ Atr Owner</b>		<b>Lvl Reason</b>	
<input checked="" type="checkbox"/> Racing results displayed				MSG INF *Messages		001 *ACTION	
<input type="checkbox"/> Display Racing results				FUN RPG Race Entry		001 *ACTION	
<input type="checkbox"/> Dam younger than horse				MSG ERR *Messages		001 *ACTION	
<input type="checkbox"/> Sire younger than horse				MSG ERR *Messages		001 *ACTION	
<b>Bottom</b>							
F3=Exit		F5=Refresh		F9=Command line		F12=Previous	
F16=Build model list		F21=Print list				F15=Top level	
						F23=More options	

## Creating a Model List of the References

In this step you will use the F16 command key to create a model object list and convert the references displayed to model list entries. You can specify the name of an existing list or you can type a new name and CA 2E will automatically create the model list.

If you specify the name of an existing list be sure the Outlist option is correct. A value of \*REPLACE clears the list before converting the references; a value of \*ADD appends the new entries to any existing entries.

## Naming the List of References

You should assign a name to the list that will be meaningful to the development staff. By default the model list name is the Model list for commands specified in your model profile.

Press F16 to display the window where you will enter the name of the model list that is to contain the references. Suppose you are responding to Problem Report 3049; as a result, you might name the model list PR3049.

Type **PR3049** in the Output to Model List window.

Gen objs : 2

Total : 9

Object : Edit Horse

Type : FUN Attribute : RPG

Scope : \*EXTFUN

Display Model References

Model : MYMDL

Level : 001

Owner : Horse

Exclude system objs : \*YES

2=Edit 3=Copy 4=Delete

13=Perms 14=GEN batch

Opt Object

— Racing results displayed

— Display Racing results

— Dam younger than horse

— Sire younger than horse

Output to Model List

Model list name : PR3049

Outlist option : \*REPLACE

Name, \*MDLPRF

\*SELECT, \*USER

\*ADD, \*REPLACE

F3=Exit F5=Refresh F9=Command line F12=Previous

F16=Build model list F21=Print list

F15=Top level

F23=More options

Bottom

Press Enter. Notice the messages at the bottom of the window as the references are converted to entries on the model object list. On return to the Display Model References panel a message at the bottom of the panel verifies that the conversion is complete.

### Using the List of References

You can now print the PR3049 model list using the F21 command key. You can either give the printed copy of the list to your development staff or they can use the model list online as an aid to solving the problem.

### Exiting Display Model References

Press F3 to return to the Edit Model Object List panel. Reposition the list to the top in Object Name/Object Type order as follows: press F7 to display the Position the List window, blank out all fields, and press Enter.

### Redisplay Your Session List

Type the name of your session list in the List field at the top of the panel. Recall that by default your session list has the same name as your user profile.

```

Edit Model Object List

Model . . : MYMDL
List . . . JAR      *All objects list for model MYMDL.

Type options, press Enter.
1=Select      2=Edit      3=Copy      4=Delete entry
5=Display     8=Details   9=Deselect 10=Action diagram

Opt      Object      Type Atr Owner
█          *ALL values  CND  LST  Horse gender
—          *ALL values  CND  LST  Jockey gender
—          *ALL values  CND  LST  Going conditions
—          *ALL values  CND  LST  Entry Status
—          Change Course  FUN  DBF  Course
—          Change Horse   FUN  DBF  Horse
—          Change Jockey  FUN  DBF  Jockey

Parameters or command
===>
F3=Exit  F4=Prompt  F5=Refresh  F6=Build  F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys
Data displayed in Object Name/Object Type order.
More...

```

Press Enter.

## Impact Analysis

This topic introduces the CA 2E impact analysis tools, including simulating a proposed change and component change processing.

New terms introduced

- Change type
- Private change
- Public change
- Simulating a change
- Component change processing

## Objectives

To show how to determine the impact of a proposed change to a model object on other objects in the model.

## Overview of Impact Analysis

Impact analysis lets you determine the impact of a proposed change or an actual change to any object in your model. For example, you can determine which objects are affected if you change the length of a key field or which functions need to be regenerated if you change logic in the action diagram of an internal function.

## Change Type

Whenever you change a model object, CA 2E assigns a *change type*. The change type describes the way in which a change impacts other objects that use the changed object. The four possible change types are Object only (\*OBJONLY), Generation required (\*GEN), Private (\*PRIVATE), and Public (\*PUBLIC). The first two indicate changes that affect only the changed object itself. As a result, this tutorial covers just the last two change types.

- \*PRIVATE - this change type indicates a change to an object that requires that you regenerate and compile all access paths and external functions that use it.

For example, if you change the action diagram of an internal function, you need to regenerate the external functions that call it.

- \*PUBLIC - this change type indicates that the interface of the model object with other objects has changed. As a result, some model objects that use the changed object may require editing. When the editing is complete, you need to regenerate the access paths and external functions that use the changed object.

For example, if you change the parameters of an internal function you need to edit all functions that call it and then regenerate all external functions that use it.

The change type depends on which attributes of a model object are changed and is derived internally by CA 2E.

## Simulating a Change to a Model Object

Simulating a change to a model object lets you see the impact of a proposed change on other objects in the model before you actually make the change. Simulation identifies which other model objects need to be edited or generated as a result of the proposed change.

When you change a model object, the only objects that can be affected by the change are those that *use* the changed object. As a result a major part of simulating a change consists of expanding usages for the object to be changed.

Suppose you want to change the length of the *Course code* field. This is a \*PUBLIC change; in other words, some using objects will need to be edited to incorporate the change.

## Positioning the List to Course Code

From the Edit Model Object List panel, press F7 to display the Position the List window. Type **FLD** for the Type field and type **Course code** for the Object field.

**Edit Model Object List**

---

**Position the List**

Type . . . FLD    Object . Course code    Imp. Name \_\_\_\_\_

Owner . \_\_\_\_\_

F12=Cancel

---

Opt	Object	Type	Atr	Owner
—	*ALL values	CND	LST	Horse gender
—	*ALL values	CND	LST	Jockey gender
—	*ALL values	CND	LST	Going conditions
—	*ALL values	CND	LST	Entry Status
—	Change Course	FUN	DBF	Course
—	Change Horse	FUN	DBF	Horse
—	Change Jockey	FUN	DBF	Jockey

More...

Parameters or command  
==>

F3=Exit	F4=Prompt	F5=Refresh	F6=Build	F7=Position to
F8=Reverse retrieve	F9=Retrieve	F23=More options	F24=More keys	

Press Enter. Type **91** in the Subfile selector for *Course code* as shown to display the Display Model Usages panel.

**Edit Model Object List**

---

Model . . . MYMDL  
List . . . JAR    List JAR in MYMDL created by user JAR.

Type options, press Enter.  
1=Select    2=Edit    3=Copy    4=Delete entry  
5=Display    8=Details    9=Deselect    10=Action diagram

---

Opt	Object	Type	Atr	Owner
91	Course code	FLD	CDE	
█	Course name	FLD	TXT	
—	Dam Date of birth	FLD	REF	
—	Dam Horse code	FLD	REF	
—	Dam name	FLD	REF	
—	Date of birth	FLD	DT#	
—	Distance	FLD	QTY	

More...

Parameters or command  
==>

F3=Exit	F4=Prompt	F5=Refresh	F6=Build	F7=Position to
F8=Reverse retrieve	F9=Retrieve	F23=More options	F24=More keys	

**Data displayed in Object Type/Object Name order.**

Press Enter.

## Positioning the Usages to Course Code

To simulate a change to Course code, you first need to locate it. Since Course code has many usages, an easy way to locate it is to use the positioning options. Type **Course code** in the Object field and type **FLD** in the Type field.



```

Gen objs :      5          Display Model Usages      Model : MYMDL
Total :      18          Level : 001
Object : Course code
Type : FLD Attribute : CDE      Exclude system objs : *YES
Scope : *NEXT      Filter : *ANY      Current objects only : *YES
Object : Course code      Type : FLD      Reason : *FIRST

2=Edit      3=Copy      4=Delete object      5=Display      8=Details      10=Action diagram
13=Parms      14=GEN batch      15=GEN interactive      16=Y2CALL

Opt Object          Typ Attr Owner          Lvl Reason
--- Course          FIL REF          001 *FILENT
--- Race          FIL REF          001 *FILENT
--- Race Entry      FIL CPT          001 *FILENT
--- Change Course    FUN DBF Course      001 *FUNPDT
--- Change Race      FUN DBF Race         001 *FUNPDT
--- Change Race Entry FUN DBF Race Entry    001 *FUNPDT
--- Create Course    FUN DBF Course      001 *FUNPDT
--- Create Race      FUN DBF Race         001 *FUNPDT
--- Create Race Entry FUN DBF Race Entry    001 *FUNPDT
More...

F3=Exit      F5=Refresh      F9=Command line      F12=Previous      F15=Top level
F16=Build model list      F21=Print list      F23=More options

```

Press Enter.

## Simulating a \*PUBLIC Change

Press F23 to display additional Subfile selector options. Option 95 lets you simulate a \*PUBLIC change for the selected model object.

Type **95** in the Subfile selector for Course code.

```

Gen objs :      5          Display Model Usages      Model : MYMDL
Total :      18          Level : 001
Object : Course code
Type : FLD Attribute : CDE      Exclude system objs : *YES
Scope : *NEXT      Filter : *ANY      Current objects only : *YES
Object : Course code      Type : FLD      Reason : *FIRST

81/82=References      91/92=Usages
94=Simulate *PRIVATE change      95=Simulate *PUBLIC change

Opt Object          Typ Attr Owner          Lvl Reason
95 Course code      FLD CDE          000 *OBJECT

Bottom

F3=Exit      F5=Refresh      F9=Command line      F12=Previous      F15=Top level
F16=Build model list      F21=Print list      F23=More options

```

Press Enter to display all objects that will be affected by a \*PUBLIC change to Course code.

```

Gen objs :      15          Display Model Usages           Model . : MYMDL
Total    :      27                               Level  :   002
Object   : Course code
Type     : FLD Attribute . : CDE       Exclude system objs . *YES
Scope    : *GENFUN Filter . : *ANY   Current objects only . *YES

2=Edit      3=Copy    4>Delete object    5=Display    8=Details    10>Action diagram
13=Parms    14=GEN batch    15=GEN interactive    16=Y2CALL

Opt Object                                Typ Attr Owner                                Lvl Chg Action
┌─── Physical file                          ACP PHY Course                                002 PUB EDT
├─── Update index                         ACP UPD Course                                003 PUB EDT
├─── Update index                         ACP UPD Course                                003 PUB EDT
├─── Retrieval index                     ACP RTU Course                                003 PUB EDT
├─── Update index                       ACP UPD Course                                002 PUB EDT
├─── Retrieval index                   ACP RTU Course                                003 PUB EDT
├─── Create Course                     FUN DBF Course                                003 PUT EDT
├─── Create Course                     FUN DBF Course                                003 PUB EDT
└─── Change Course                     FUN DBF Course                                003 PUT EDT

More...
F3=Exit    F5=Refresh    F9=Command line    F12=Previous    F15=Top level
F16=Build model list    F21=Print list    F23=More options

Warning: Simulation of a *PUB change to object 'Course code'.

```

Gen objs : 15		<b>Display Model Usages</b>		Model : MYMDL	
Total : 27				Level : 002	
Object : Course code					
Type : FLD Attribute : CDE		Exclude system objs : *YES			
Scope : *GENFUN		Filter :			
<b>Output to Model List</b>					
2=Edit 3=Copy 4=Delete		Model list name : <u>COURSECODE</u>		Name, *MDLPRF	
13=Parms 14=GEN batch		Outlist option : <u>REPLACE</u>		*SELECT, *USER	
		F12=Cancel		*ADD, *REPLACE	
<b>Opt Object</b>					
— Physical file					
— Update index					
— Update index					
— Retrieval index					
— Update index		ACP UPD Course 002 PUB EDT			
— Retrieval index		ACP RTU Course 003 PUB EDT			
— Create Course		FUN DBF Course 003 PVT EDT			
— Create Course		FUN DBF Course 003 PUB EDT			
— Change Course		FUN DBF Course 003 PVT EDT			
		<b>More...</b>			
F3=Exit F5=Refresh F9=Command line F12=Previous F15=Top level		F23=More options			
F16=Build model list F21=Print list					

Press Enter. Notice the messages at the bottom of the window as CA 2E creates the list and converts the displayed usages to model list entries.

Press F3 twice to return to the Edit Model Object List panel. Use the F7 key to reposition the session list to the top and display all entries.

## Optional Exercise

Type **COURSECODE** in the List field and press Enter to display the model object list you just created. Use the F11 key to view information for each list entry.

When you finish viewing this list, type the name of your session list in the List field and press Enter.

## Component Change Processing

*Component change processing* refers to an automated impact analysis tool. When you simulate a change you are in effect simulating component change processing. The difference is that component change processing updates the All Objects list for the changed object and all objects affected by the change, including an indication of whether the object needs to be edited or regenerated.

Component change processing is optional; you control it using the model profile and the Component Change Processing (YCMPCHG) model value. You can choose to run it interactively, by request, or in batch.

Further discussion of this tool is beyond the scope of this tutorial; however, the simulation topic should have given you some familiarity with the process. Component change processing is discussed in detail in the CA 2E module, *Generating and Implementing Applications*.

## Accessing Edit Database Relations

At this point you can exit the model by pressing F3. If you want to continue with the tutorial, you can use either the Edit Model (YEDTMDL) command or the Start CA 2E (YSTRY2) command to reenter the model at the Edit Database Relations panel.

For example, type YEDTMDL on the command line.

```

                                Edit Model Object List

Model . . : MYMDL
List . . . JAR      List JAR in MYMDL created by user JAR.

Type options, press Enter.
1=Select      2=Edit      3=Copy      4=Delete entry
5=Display     8=Details   9=Deselect 10=Action diagram

Opt   Object                                Type  Attr  Owner
---   ---
---   *ALL values                          CND   LST   Horse gender
---   *ALL values                          CND   LST   Jockey gender
---   *ALL values                          CND   LST   Going conditions
---   *ALL values                          CND   LST   Entry Status
---   Change Course                        FUN   DBF   Course
---   Change Horse                         FUN   DBF   Horse
---   Change Jockey                        FUN   DBF   Jockey

Parameters or command
==> YEDTMDL
F3=Exit      F4=Prompt   F5=Refresh  F6=Build    F7=Position to
F8=Reverse retrieve  F9=Retrieve  F23=More options  F24=More keys

```

Press Enter.

# Chapter 7: Advanced Functions

---

This chapter demonstrates the use of the Edit Transaction (EDTTRN) function with a Span (SPN) access path and the use of function fields. This chapter introduces the following topics.

- The Edit Transaction Function Type
- Span Access Path
- Defining an Edit Transaction Function
- Using Function Fields
- Defining Function Field Parameters

## Span Access Path and Edit Transaction Function

This topic demonstrates the use of the Edit Transaction (EDTTRN) function with a Span (SPN) access path.

New terms introduced

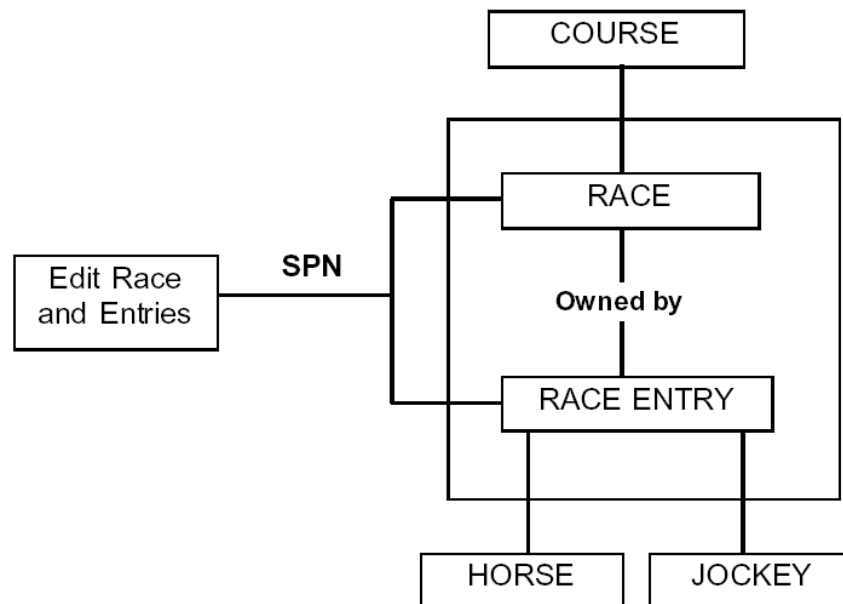
- Access path format
- New panels introduced
- Display Access Path Formats

## Objectives

You will specify a SPN access path based on the RACE and RACE ENTRY files.

## Overview

The Edit Transaction function allows two database files to be updated simultaneously. The Edit Transaction function type requires a SPN access path, which is a logical file with two formats.



In order for a SPN access path to be built over two files, the two files must be connected by a file-to-file relation; for example, an Owned by relation. The Edit Transaction function provides a display consisting of a subfile control format for the owning file and a subfile record format for the owned file.

You will set up an Edit Transaction function for the RACE and RACE ENTRY files. This provides the facility to enter both the race details and the race entry details for a given race on the same interactive panel.

## Overview of Processing Steps

To implement the Edit Transaction function you will:

1. Define and generate a SPN access path for the RACE and RACE ENTRY files.
2. Create the Edit Transaction function and modify the device design for the Edit Race and Entries function.
3. Generate source for the function.
4. Compile the access path, display file, and program source.

## Entering Your Design Model

Return to the Edit Database Relations panel for your design model. Select the RACE and RACE ENTRY files by typing **Race\*** on the selection line.

EDIT DATABASE RELATIONS				My model			
=>	Rel	lvl:		Seq	Typ	Referenced object	
? Typ	Object	Relation		Seq	Typ	Referenced object	
—	FIL	Course	Known by	—	FLD	Course code	
—	FIL	Course	Has	—	FLD	Course name	
—	FIL	Horse	Known by	10	FLD	Horse code	
—	FIL	Horse	Has	20	FLD	Horse name	
—	FIL	Horse	Has	30	FLD	Horse gender	
—	FIL	Horse	Has	40	FLD	Horse value	
—	FIL	Horse	Has	50	FLD	Date of birth	
—	FIL	Horse	Refers to	60	FIL	Horse	
—	For: Dam				Sharing: *	ALL	
—	FIL	Horse	Refers to	70	FIL	Horse	
—	For: Sire				Sharing: *	ALL	
—	FIL	Jockey	Known by	—	FLD	Jockey code	
—	FIL	Jockey	Has	—	FLD	Jockey name	
—	FIL	Jockey	Has	—	FLD	Jockey gender	
—	FIL	Race	Owned by	—	FIL	Course	

More...

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options  
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

Press Enter.

## Creating a Span Access Path

Create a SPN access path on the RACE file. A SPN access path must be attached to the file that is to form the header format of the access path. In this case, you will create the SPN access path over the RACE file.

Type **Z** against any relation on the RACE file to display all existing access paths.

EDIT DATABASE RELATIONS				My model			
=>	Rel	lvl:		Seq	Typ	Referenced object	
? Typ	Object	Relation		Seq	Typ	Referenced object	
<b>Z</b> FIL	Race	Owned by		—	FIL	Course	
—	FIL	Race	Known by	—	FLD	Race date	
—	FIL	Race	Known by	—	FLD	Race time	
—	FIL	Race	Has	—	FLD	Race name	
—	FIL	Race	Has	—	FLD	Going conditions	
—	FIL	Race	Has	—	FLD	Distance	
—	FIL	Race	Has	—	FLD	Prize money	
—	FIL	Race Entry	Owned by	—	FIL	Race	
—	FIL	Race Entry	Known by	—	FLD	Entry number	
—	FIL	Race Entry	Refers to	—	FIL	Horse	
—	FIL	Race Entry	Refers to	—	FIL	Jockey	
—	FIL	Race Entry	Has	—	FLD	Finishing position	
—	FIL	Race Entry	Has	—	FLD	Handicap	
—	FIL	Race Entry	Has	—	FLD	Entry Status	

Bottom

Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options  
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys

Press Enter. The Edit File Details panel displays.

## Defining the Span Access Path

To specify the SPN access path you need to:

1. Define two access path file formats, one for the subfile control (header) and one for the subfile record (detail)
2. Specify the keys for those formats

Define the new access path by calling it Race and Entries of type SPN. Type the access path type, name, and **Z**.

EDIT FILE DETAILS		My model																																																							
File name . . . . .	Race	Field reference file. : *	NONE																																																						
Attribute . . . . .	REF	Source library. . . . .	MYGEN																																																						
Documentation sequence. . .		Distributed . . . . .	N (Y,N)																																																						
GEN format prefix . . . . .	AC																																																								
Assimilated physical. . . .																																																									
Record not found message. .	Race	NF Msgid. : USR0003																																																							
Record exists message . . .	Race	EX Msgid. : USR0004																																																							
<table border="1"> <thead> <tr> <th>? Typ</th> <th>Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>PHY</td> <td>Physical file</td> <td>MYACREP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>UPD</td> <td>Update index</td> <td>MYACREL0</td> <td>UNIQUE</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td>RTV</td> <td>Retrieval index</td> <td>MYACREL1</td> <td>UNIQUE</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td><u>Z</u></td> <td><u>SPN Race and Entries</u></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>				? Typ	Access path	Source mbr	Key	Index options	Auto add	PHY	Physical file	MYACREP	NONE		ATR ONLY	UPD	Update index	MYACREL0	UNIQUE	IMMED	ATR ONLY	RTV	Retrieval index	MYACREL1	UNIQUE	IMMED	ATR ONLY	<u>Z</u>	<u>SPN Race and Entries</u>					-						-						-						-					
? Typ	Access path	Source mbr	Key	Index options	Auto add																																																				
PHY	Physical file	MYACREP	NONE		ATR ONLY																																																				
UPD	Update index	MYACREL0	UNIQUE	IMMED	ATR ONLY																																																				
RTV	Retrieval index	MYACREL1	UNIQUE	IMMED	ATR ONLY																																																				
<u>Z</u>	<u>SPN Race and Entries</u>																																																								
-																																																									
-																																																									
-																																																									
-																																																									
SEL: Z=Details, G/J=Generate, E=STRSEU, D=Delete, L=Locks, O=Overrides H=Hold/Release, T=Trim, V=Virtualize, U=Usage, F=Func refs., M=Narrative F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative																																																									

Press Enter. The Edit Access Path Details panel displays.

## Entering the Access Path Formats

An access path format defines a view upon a physical file. By default, it makes all of the non-virtual entries in that file available. An access path format can be defined as a list of access path relations that will be resolved into a list of field entries. Relations can be removed from a view on the Edit Access Path Relations panel.

When you previously used the Edit Access Path Details panel, the access path format was automatically provided. For the SPN access path you must explicitly add the formats. Note that it is important that you add the formats in the correct sequence: header format first, detail format second.

Press F9 to display the Display Access Path Formats panel where you will add the format for the subfile control (header).



```

EDIT ACCESS PATH DETAILS                               My model
File name . . . . . : Race                               Attribute . : REF
Access path name . . . : Races and Entries               Type . . . : SPN
Unique or duplicate order : F (U-Unique,F-FIFO,L-LIFO,C-FCFO,'-Undefined)
Index maintenance option : I (I-IMMED, D-DLY, R-REBLD)
Alternate collating table :                               (S-Static, D-Dynamic, '-None)
Allow select/omit . . . :                               (M-MDLVAL, D-DDS, S-SQL, X-UNX)
Generation mode . . . : M                               Source member name . : MYACREL2
Source member text . . : Race                               Races and Entries

SEL: Z=Entries, R=Rel, S=Sel/omit, A=Assoc.acp, T=Trim, V=Virtualize, D=Delet
F3=Exit F5=Reload F8=Rename F9=Add format F20=Narrative

```

## Selecting the First Access Path Format

The Display Access Path Formats panel shows you the files that you can select for the SPN access path. In this case, the two files shown are connected to the RACE file by an Owned by and a Defined as relation.

Select the RACE format with **X**.

```

DISPLAY ACCESS PATH FORMATS                             My model
File name . . . . . : Race                               Attribute . : SPN
Access path name . . . : Race and Entries

? File           Relation      For
X Race           Defined as
■ Race Entry      Owned by

SEL: X=Select format of specified file.
F3=Exit

```

Press Enter.

## Confirming Selection of the First Format

The selection of the first format for the SPN access path, Race and Entries, has now been confirmed. Press F9 to add a second format for the subfile record (detail).

```

EDIT ACCESS PATH DETAILS                               My model
File name . . . . . : Race                               Attribute . : REF
Access path name . . . : Race and Entries                 Type . . . : SPN
Unique or duplicate order : F (U-Unique,F-FIFO,L-LIFO,C-FCFO,''-Undefined)
Index maintenance option : I (I-IMMED, D-DLY, R-REBLD)
Alternate collating table : 
Allow select/omit . . . : (S-Static, D-Dynamic, ' '-None)
Generation mode . . . : M (M-MDLVAL, D-DDS, S-SQL, X-UNX)
Source member name . . : MYACREL2
Source member text . . : Race                             Race and Entries

      Format      GEN  Format text      Associated
? Seq name      pfx  (Based on file)  Retrieval access path
■ 1 FACREAO     AC   Race              Retrieval index

SEL: Z=Entries, R=Rels, S=Sel/omit, A=Assoc.acp, T=Trim, V=Virtualize, D=Delet
F3=Exit F5=Reload F8=Rename F9=Add format F20=Narrative

```

## Specifying the Second Format

The Display Access Path Formats panel displays showing the file-to-file relations for the RACE file. Type **X** to select the Race Entry format for the subfile record.

```

DISPLAY ACCESS PATH FORMATS                           My model
File name . . . . . : Race                               Attribute . : SPN
Access path name . . . : Race and Entries

? File          Relation      For
■ Race          Defined as
X Race Entry     Owned by

SEL: X=Select format of specified file.
F3=Exit

```

Press Enter to return to the Edit Access Path Details panel.

## Adding Virtual Fields to the Access Path Formats

When the end user enters races and race entries, it would be helpful to have the following virtual fields display on the interactive panel: Course name, Horse name, and Jockey name. As mentioned previously, each access path initially contains all of the relations for the file on which it is based, but none of the virtual entries. As a result, if you want virtual fields included on the two formats for the Race and Entries access path, you need to add them explicitly.

In this step, you will add these virtual fields to the Race and Race Entry access path formats. Type **V** in the Subfile selector for both the Race format and the Race Entry format.

```

EDIT ACCESS PATH DETAILS                               My model
File name . . . . . : Race                               Attribute . : REF
Access path name . . . : Races and Entries               Type . . . : SPH
Unique or duplicate order : F (U-Unique,F-FIFO,L-LIFO,C-FCFO,' '-Undefined)
Index maintenance option : I (I-IMMED, D-DLY, R-REBLD)
Alternate collating table :
Allow select/omit . . . : (S-Static, D-Dynamic, ' '-None)
Generation mode . . . : M (M-MDLVAL, D-DDS, S-SQL, X-UNIX)
Source member name . . : MYACREL2
Source member text . . : Race                           Races and Entries

Format      GEN  Format text      Associated
? Seq name  pfx  (Based on file)  Retrieval access path
V 1 FACREA2  AC   Race           Retrieval index
V 2 FACCPA3  AD   Race Entry     Retrieval index

SEL: Z=Entries, R=Rels, S=Sel/omit, A=Assoc.acp, T=Trim, V=Virtualize, D=Delet
F3=Exit F5=Reload F8=Rename F9=Add format F20=Narrative

```

Press Enter.

The Virtualize Access Path panel displays showing the virtual field, Course name that can be added to the Race access path format.

```

VIRTUALIZE ACCESS PATH                               My model
File name . . . . . : Race                               Attribute . : REF
Access path . . . . . : Retrieval index                 Type . . . : RTV

Field      Type      Ocr  Et Ksq GEN name  Length  Renamed
Course name  TXT          V    ABTX      25

F3=Exit, no update  ENTER=Validate

```

Press Enter to add the Course name virtual field to the Race access path format.

A Confirm prompt displays in the lower right-hand corner of the panel.

VIRTUALIZE ACCESS PATH		My model					
File name . . . . .	: Race						Attribute. : REF
Access path . . . . .	: Retrieval index						Type . . . . : RTV
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Course name	TXT		V		ABTX	25	
F3=Exit, no update ENTER=Validate							
						CONFIRM: <input checked="" type="checkbox"/> (Y,N)	

Press Enter to confirm the addition of the Course name virtual field.

CA 2E again displays the Virtualize Access Path panel showing the virtual fields that can be added to the Race Entry access path format. In this case, the virtual fields are Horse name and Jockey name.

VIRTUALIZE ACCESS PATH		My model					
File name . . . . .	: Race Entry						Attribute. : CPT
Access path . . . . .	: Retrieval index						Type . . . . : RTV
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Horse name	TXT		V		AOTX	25	
Jockey name	TXT		V		AETX	25	
F3=Exit, no update ENTER=Validate							

Press Enter to add these virtual fields to the Race Entry access path format. A Confirm prompt displays in the lower right-hand corner of the panel.

VIRTUALIZE ACCESS PATH		My model					
File name . . . . .	: Race Entry					Attribute . : CPT	
Access path . . . . .	: Retrieval index					Type . . . : RTV	
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Horse name	TXT		V		ADTX	25	
Jockey name	TXT		V		AETX	25	
F3=Exit, no update ENTER=Validate							
CONFIRM: <input checked="" type="checkbox"/> (Y,N)							

Press Enter to confirm the addition of the Horse name and Jockey name virtual fields and return to the Edit Access Path Details panel.

## Details for the First Access Path Format

The Edit Access Path Details panel displays both formats. Next, display the details for the access path format by zooming against the Race access path format. This displays the Edit Access Path Format Entries panel.

Type **Z** against the Race access path format.

EDIT ACCESS PATH DETAILS		My model	
File name . . . . .	: Race	Attribute . : REF	
Access path name . . . . .	: Race and Entries	Type . . . : SPN	
Unique or duplicate order . . . . .	: F (U-Unique, F-FIFO, L-LIFO, C-FCFO, ' '-Undefined)		
Index maintenance option . . . . .	: I (I-IMMED, D-DLV, R-REBLD)		
Alternate collating table . . . . .	: (S-Static, D-Dynamic, ' '-None)		
Allow select/omit . . . . .	: M (M-MDLVAL, D-DDS, S-SQL, X-UNX)		
Generation mode . . . . .	: MYACREL2		
Source member name . . . . .	: Race	Race and Entries	
Source member text . . . . .			
Format	GEN	Format text	Associated
? Seq name	pfx (Based on file)		Retrieval access path
Z <u>1</u> <u>FACREAB</u>	AC	<u>Race</u>	<u>Retrieval index</u>
- <u>2</u> <u>FADCPA1</u>	AD	<u>Race Entry</u>	<u>Retrieval index</u>
SEL: Z-Entries, R-Rel, S-Sel/omit, A-Assoc.acp, T-Trim, V-Virtualize, D-Delet			
F3=Exit F5=Reload F8=Rename F9=Add format F20=Narrative			

Press Enter.

## Specifying the Key for the First Access Path Format

The next step is to check the keys of the Race access path format. You may specify any of the fields from the RACE file as keys, in either ascending or descending order. However, there must be a common key between the two formats. For example, the header record key fields must be part of the detail record's key.

The keys will be defaulted to the keys as specified by the file relations. In the case of two files linked by an Owned by relation, the default keys are recommended. If the two file formats are linked by a Refers to relation, you need to change the key order.

EDIT ACCESS PATH FORMAT ENTRIES				My model																																																																									
File name . . . . .	:	Race		Attribute . . .	:	REF																																																																							
Access path name. . . . .	:	Races	and Entries	Type. . . . .	:	SPN																																																																							
Format text . . . . .	:	Race																																																																											
Based on. . . . .	:	Race		Format No . . .	:	1																																																																							
<table border="1"> <thead> <tr> <th>? Field</th> <th></th> <th>GEN</th> <th>Type</th> <th>Key</th> <th>Altcol</th> <th>Ref</th> </tr> <tr> <th></th> <th></th> <th>Name</th> <th></th> <th>no. Dsc seq</th> <th></th> <th>cnt</th> </tr> </thead> <tbody> <tr> <td>■ Course code</td> <td>CDE</td> <td>ABCD</td> <td>K</td> <td><u>1</u></td> <td>-</td> <td>1</td> </tr> <tr> <td>- Course name</td> <td>TXI</td> <td>ABTX</td> <td>V</td> <td></td> <td></td> <td>1</td> </tr> <tr> <td>- Race date</td> <td>DT#</td> <td>ABDZ</td> <td>K</td> <td><u>2</u></td> <td>-</td> <td>1</td> </tr> <tr> <td>- Race time</td> <td>TM#</td> <td>ABTZ</td> <td>K</td> <td><u>3</u></td> <td>-</td> <td>1</td> </tr> <tr> <td>- Race name</td> <td>TXI</td> <td>ACTX</td> <td>A</td> <td></td> <td></td> <td>1</td> </tr> <tr> <td>- Going conditions</td> <td>STS</td> <td>ABST</td> <td>A</td> <td></td> <td></td> <td>1</td> </tr> <tr> <td>- Distance</td> <td>QTY</td> <td>AAQT</td> <td>A</td> <td></td> <td></td> <td>1</td> </tr> <tr> <td>- Prize money</td> <td>VAL</td> <td>AAVA</td> <td>A</td> <td></td> <td></td> <td>1</td> </tr> </tbody> </table>								? Field		GEN	Type	Key	Altcol	Ref			Name		no. Dsc seq		cnt	■ Course code	CDE	ABCD	K	<u>1</u>	-	1	- Course name	TXI	ABTX	V			1	- Race date	DT#	ABDZ	K	<u>2</u>	-	1	- Race time	TM#	ABTZ	K	<u>3</u>	-	1	- Race name	TXI	ACTX	A			1	- Going conditions	STS	ABST	A			1	- Distance	QTY	AAQT	A			1	- Prize money	VAL	AAVA	A			1
? Field		GEN	Type	Key	Altcol	Ref																																																																							
		Name		no. Dsc seq		cnt																																																																							
■ Course code	CDE	ABCD	K	<u>1</u>	-	1																																																																							
- Course name	TXI	ABTX	V			1																																																																							
- Race date	DT#	ABDZ	K	<u>2</u>	-	1																																																																							
- Race time	TM#	ABTZ	K	<u>3</u>	-	1																																																																							
- Race name	TXI	ACTX	A			1																																																																							
- Going conditions	STS	ABST	A			1																																																																							
- Distance	QTY	AAQT	A			1																																																																							
- Prize money	VAL	AAVA	A			1																																																																							
SEL: Z-Field details, L-Locks.																																																																													
F3=Exit F7=Relations																																																																													

Because in this example, RACE ENTRY is Owned by RACE, press Enter to accept the defaults.

Press F3 to exit and return to the Edit Access Path Details panel.

## Details for the Second Access Path Format

In this step you will zoom into the Race Entry access path format to check its key.

Type **Z** against the Race Entry access path format.

```

EDIT ACCESS PATH DETAILS          My model
File name . . . . . : Race          Attribute . : REF
Access path name. . . . : Race and Entries  Type. . . . : SPN
Unique or duplicate order : F (U-Unique,F-FIFO,L-LIFO,C-FCFO,' '-Undefined)
Index maintenance option : I (I-IMMED, D-DLY, R-REBLD)
Alternate collating table :
Allow select/omit . . . . : (S-Static, D-Dynamic, ' '-None)
Generation mode . . . . : M (M-MDLVAL, D-DDS, S-SQL, X-UNX)
Source member name . . . : MYACREL2
Source member text . . . : Race          Race and Entries

Format      GEN  Format text      Associated
? Seq name  pfx  (Based on file)  Retrieval access path
1 FACREA0   AC   Race          Retrieval index
2 FADCPA1   AD   Race Entry      Retrieval index

SEL: Z-Entries, R-Rels, S-Sel/omit, A-Assoc.acp, T-Trim, V-Virtualize, D-Delet
F3=Exit F5=Reload F8=Rename F9=Add format F20=Narrative

```

Press Enter.

The keys Course code, Race date, Race time, and Entry number have been selected by default to be the keys of the format since these are the keys as defined by the file relations for the RACE ENTRY file.

```

EDIT ACCESS PATH FORMAT ENTRIES    My model
File name . . . . . : Race          Attribute . : REF
Access path name. . . . : Races and Entries  Type. . . . : SPN
Format text . . . . . : Race Entry      Format No . : 2
Based on. . . . . : Race Entry

? Field      GEN  Name      Type  Key  Altcol Ref
            no. Dsc seq  cnt
- Course code CDE  ABCD      K      1  -      1
- Race date   DT#  ABDZ      K      2  -      1
- Race time   TM#  ABTZ      K      3  -      1
- Entry number CDE  ACCD      K      4  -      1
- Horse code  CDE  ADCD      A      -  -      1
- Horse name  TXT  ADTX      V      -  -      1
- Jockey code CDE  AECD      A      -  -      1
- Jockey name TXT  AETX      V      -  -      1
- Finishing position HBR  ABHB      A      -  -      1
- Handicap    QTY  ABQT      A      -  -      1
- Entry Status STS  ACST      A      -  -      1

SEL: Z-Field details, L-Locks.
F3=Exit F7=Relations

```

Press Enter to confirm the keys.

Press F13 to exit and return to the Edit File Details panel.

## Requesting Batch Generation of the SPN Access Path

You have now finished the definition of the Races and Entries SPN access path. Request batch generation of the Races and Entries access path by typing **J** against it

EDIT FILE DETAILS		My model		
File name . . . . .	Race	Field reference file. :	*NONE	
Attribute . . . . .	REF	Source library. . . . :	MYGEN	
Documentation sequence. :		Distributed . . . . .	N (Y,N)	
GEN format prefix . . . :	AC			
Assimilated physical. . . :				
Record not found message. :	Race	HF Msgid. :	USR0003	
Record exists message . . :	Race	EX Msgid. :	USR0004	
? Typ Access path Source mbr Key Index options Auto add				
PHY Physical file	MYACREP	NONE		ATR ONLY
UPD Update index	MYACREL8	UNIQUE	IMMED	ATR ONLY
RTV Retrieval index	MYACREL1	UNIQUE	IMMED	ATR ONLY
J SPN Races and Entries	MYACREL2	FIFO	IMMED	ATR ONLY
SEL: Z=Details, G/J=Generate, E=STRSEU, D=Delete, L=Locks, O=Overrides H=Hold/Release, T=Trim, V=Virtualize, U=Usage, F=Func refs., N=Narrative F3=Exit F5=Reload F7=Funcs. F8=Change name F17=Serv. F20=Narr. F22=File Locks				

Press Enter.

## The Edit Transaction Function

In this step you will define a function of type Edit Transaction (EDTTRN).

## Objectives

You will create an Edit Transaction function for the RACE and RACE ENTRY files. The name of the function will be Edit Race and Entries and it will use the SPN access path Race and Entries you just created. This will provide the facility to enter both the race details and the race entry details for a given race on a single panel.

## Defining the Edit Transaction Function

From the Edit File Details panel, you can transfer directly to the Edit Functions panel for the RACE file by pressing F7.



EDIT FILE DETAILS

My model

File name : Race

Attribute : REF

Documentation sequence :

GEN format prefix : AC

Assimilated physical :

Record not found message : Race

Record exists message : Race

Field reference file : \*NONE

Source library : MYGEN

Distributed : N (Y,N)

HF Msgid : USR0003

EX Msgid : USR0004

? Typ Access path

PHY Physical file

UPD Update index

RTV Retrieval index

SPH Race and Entries

Source mbr

Key

Index options

Auto add

MYACREP NONE

MYACREL0 UNIQUE IMMED

MYACREL1 UNIQUE IMMED

MYACREL2 FIFO IMMED

ATR ONLY

ATR ONLY

ATR ONLY

ATR ONLY

SEL: Z-Details, G/J-Generate, E-STRSEU, D-Delete, L-Locks, O-Overrides

H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., H-Narrative

F3=Exit F5=Reload F7=Functions F8=Change name F17=Services F20=Narrative

Source generation request for MYACREP accepted.

Press F7 to access the Edit Functions panel.

Type the function details; namely, type **Edit Race and Entries** for the Function name, **EDTTRN** for the Function type, **Race and Entries** for the Access path, and **S** to edit the device design.

EDIT FUNCTIONS

My model

File name : Race

\*\* 1ST LEVEL \*\*

? Function

Change Race

Create Race

Delete Race

Edit Race

Select Race

Edit Race and Entries

Function type

Change object

Create object

Delete object

Edit file

Select record

EDTTRN

Access path

Update index

Update index

Update index

Retrieval index

Retrieval index

Race and Entries

More...

SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, H-Narr, O-Open,

T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock,

F3=Exit F5=Reload F7=File details F9=Add functions F17=Services

Press Enter to create the Edit Race and Entries function and display the device design.

The Default Device Design

The default device design for the Edit Race and Entries function should look like the panel below. Note that the subfile control (header) is positioned at the top with the subfile record (detail) below. The common key of the two file formats displays only in the subfile control and acts as an implicit restrictor parameter to the subfile detail records.

Note also the virtual fields Course name, Horse name, and Jockey name. If you had

```

File  fFunction  Selector  Help
-----
*PROGRAM  *PGMMOD                      DD/MM/YY HH:MM:SS

Edit Race and Entries
Course code .   _____  Course name :  00000000000000000000000000000000  Race date .

Select items, then select an action.



| Opt | Entry<br>number | Horse<br>code | Horse name                       | Jockey<br>code | Jockey name                      |
|-----|-----------------|---------------|----------------------------------|----------------|----------------------------------|
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____          | 00000000000000000000000000000000 |



F3=Exit  F5=Reset  F9=Open  F10=Actions
Design exceeds device size limits for function Edit Race and Entries

```

not explicitly added these fields to the formats of the SPN access path, they would not appear on this display.

## The Modified Device Design

Update this default device design to look like the one shown on the next panel. Use the function keys you used to edit the device design for the Edit Horse and Select Horse functions earlier in this tutorial.

```

File  fFunction  Selector  Help
-----
*PROGRAM  *PGMMOD                      DD/MM/YY HH:MM:SS

Edit Race and Entries
Course code . . .   _____  00000000000000000000000000000000
Race date . . . .   _____  Name . _____
Going conditions   -  Distance .   _____  Prize money .   _____

Select items, then select an action.



| Opt | Entry<br>number | Horse<br>code | Horse<br>name                    | Finishing<br>position | Entry<br>Status | H'cap |
|-----|-----------------|---------------|----------------------------------|-----------------------|-----------------|-------|
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |
| -   | _____           | _____         | 00000000000000000000000000000000 | _____                 | -               | _____ |



F3=Exit  F5=Reset  F9=Open  F10=Actions

```

**Note:** To move a field to another position on the panel, place the cursor on the field to be moved and press F8. Position the cursor on the field that appears before the desired position for the field you are moving. Press F8. For example, place the cursor on the Finishing position field, and press F8. Then position the cursor on the Horse name field, and press F8 again. This moves the Finishing position field after Horse name.

Defining Optional Entry Fields

You have now updated the device design to meet the initial requirements. However, by default, CA 2E defines all input-capable fields as required entry fields. To allow the end user to enter race entries without specifying all the details about those entries, you can make some of the fields optional. Note that only non-key fields can be optional.

To make some of the fields in the RACE ENTRY file optional, position the cursor on a field in the first subfile record.

FileFunctionSelectorHelp

\*PROGRAM\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Race and Entries

Course code . . . 000000000000000000000000

Race date . . . Name .

Going conditions Distance Prize money .

Select items, then select an action.

Opt	Entry number	Horse code	Horse name	Finishing position	Entry Status	H'cap
-			000000000000000000000000		-	
	Jockey:		000000000000000000000000		-	
-			000000000000000000000000		-	
	Jockey:		000000000000000000000000		-	
-			000000000000000000000000		-	
	Jockey:		000000000000000000000000		-	
-			000000000000000000000000		-	
	Jockey:		000000000000000000000000		-	

F3=ExitF5=ResetF9=OpenF10=Actions

Press F7.

This displays the relations for the subfile record format. You specify optional relations by typing **O** against the relations.

Type **O** against the relations.

EDIT SCREEN FORMAT RELATIONS		My model	
File name . . . . .	Race	Attribute . . .	REF
Access path name. . . . .	Races and Entries	Type. . . . .	SPN
Format text . . . . .	Race Entry		
Based on. . . . .	Race Entry	Format No . . .	2
? Verb	File/for	Access path/Function	Check
<input checked="" type="checkbox"/> Owned by	Race	Retrieval index	REQUIRED
<input type="checkbox"/> Known by	Entry number		REQUIRED
<input type="checkbox"/> Refers to	Horse	Retrieval index	REQUIRED
<input type="checkbox"/> Refers to	Jockey	Retrieval index	REQUIRED
<input type="checkbox"/> Has	Finishing position		REQUIRED
<input type="checkbox"/> Has	Handicap		REQUIRED
+			
R-Required, O-Optional, N-No error, U-User, S-Select F4, T-Default F4 F3=Exit			

Press Roll Up to view the remaining relations listed. Type **O** against the relation.

EDIT SCREEN FORMAT RELATIONS		My model	
File name . . . . .	Race	Attribute . . .	REF
Access path name. . . . .	Races and Entries	Type. . . . .	SPN
Format text . . . . .	Race Entry		
Based on. . . . .	Race Entry	Format No . . .	2
? Verb	File/for	Access path/Function	Check
<input type="checkbox"/> Has	Entry Status		REQUIRED
R-Required, O-Optional, N-No error, U-User, S-Select F4, T-Default F4 F3=Exit			

Press Enter to confirm the changes to the format relations. Note that the values in the Check column changed from REQUIRED to OPTIONAL.

Press Roll Down to view the beginning of the list of relations.

EDIT SCREEN FORMAT RELATIONS		My model	
File name . . . . .	: Race	Attribute . . .	REF
Access path name . . . . .	: Races and Entries	Type . . . . .	SPN
Format text . . . . .	: Race Entry		
Based on . . . . .	: Race Entry	Format No . . .	2
? Verb	File/for	Access path/Function	Check
■ Owned by	Race	Retrieval index	REQUIRED
Known by	Entry number		REQUIRED
_ Refers to	Horse	Retrieval index	OPTIONAL
_ Refers to	Jockey	Retrieval index	OPTIONAL
_ Has	Finishing position		OPTIONAL
_ Has	Handicap		OPTIONAL
+			
R-Required, O-Optional, N-No error, U-User, S-Select F4, T-Default F4			
F3=Exit			

Press F3 return to the device design.

## Introduction to Function Fields

You will now add function fields to your device design.

New terms introduced

- Function field
- New panels introduced
- Edit Device Function Field

## Objectives

Add a new field to the subfile control format of the Edit Transaction panel that shows the number of horses that finished the race. To do so you will define a function field containing the total number of race entries having an Entry Status of Finished.

## Overview of Function Fields

*Function fields* are non-database fields that may be used in device designs and action diagrams. The same function field can be used in many different functions.

There are six types, or usages, of function fields. Four provide standard field-level functions: SUM, MIN, MAX, and CNT; two are user-defined and let you define your own function fields: USR and DRV.

Function field parameters specify which field values are to be passed into the function field at execution time and which field is to be returned from the function field as the result. All function fields with the exception of USR fields have one output parameter, the function field itself. The system-defined function fields have a single input parameter. Derived function fields, of usage DRV, are specified in the action diagram and can have up to nine input parameters.

Adding Function Fields

In this step, you will define a function field of usage DRV to evaluate whether or not a given race entry finished a race. The DRV field will contain a value of 1 if the horse finished and 0 if it did not. You will also define a function field with a usage of SUM to calculate the number of finishers. The sum of the values in the DRV function field gives the count of the number of finishers.

Position the cursor on the field that is to be displayed on the panel before the function field, in this case Handicap.

FilefUnctionSelectorHelp

\*PROGRAM\*PGMMOD

DD/MM/YY HH:MM:SS

-----

Edit Race and Entries

Course code . . . 000000000000000000000000

Race date . . . Name

Going conditions Distance Prize money

Select items, then select an action.

Opt	Entry number	Horse code	Horse name	Finishing position	Entry Status	H'cap
-	Jockey:		000000000000000000000000		-	
-	Jockey:		000000000000000000000000		-	
-	Jockey:		000000000000000000000000		-	
-	Jockey:		000000000000000000000000		-	
-	Jockey:		000000000000000000000000		-	

F3=ExitF5=ResetF9=OpenF10=Actions

Press F19 to display the Edit Device Function Field panel.

Naming the Function Field

If the function field already exists and you know its name, you can fill in the name on the Edit Device Function Field panel. Otherwise, type a ? to display existing fields.

Type ? for the Field name.

```

EDIT DEVICE FUNCTION FIELD          My model
Format. . . . . : Subfile record.
Field name . . . : ? _____ ('?' to select)

F3=Exit  F5=Parameters

```

Press Enter.

## Displaying Existing Fields

The ? displays a list of all existing fields. You can select an existing function field or you can create a new one. In this tutorial, you will create two new function fields.

## Defining Two New Function Fields

To add the function fields you require, press F10 to display the Define Objects panel. You can define both function fields at this point.

```

DISPLAY FIELDS                      My model
? Field name                        Type REF  (*ZERO) (*BLANK)
                                     Length  Field name  Field usage
- Course code                       CDE    6      ABCD      CDE
- Course name                       TXT    25      ABTX      ATR
- Dam Date of birth                 DT#   REF   10      ADD2      ATR
- Dam Horse code                    CDE   REF    6      ACD      CDE
- Dam name                          TXT   REF   25      AFTX      ATR
- Date of birth                     DT#    10      ACD2      ATR
- Distance                          QTY    5.0     AAQT      ATR
- Entry number                      CDE    6      ACCD      CDE
- Entry Status                      STS    1      ACST      ATR
- Finishing position                NBR    5.0     ABNB      ATR
- Going conditions                  STS    1      ABST      ATR
- Handicap                          QTY    5.0     ABOT      ATR
- Horse code                        CDE    6      ADCD      CDE      +

SEL: P-Parameters, N-Narrative
      X-Select.
F3=Exit  F5=Reload  F10=Define field

```

Here are the details for the two function fields:

- Type the new function fields.

Press Enter.

Since you specified **Y** for Edit field, you will be shown the field details for the Finished race field. You can now change the internal field length to 3 from the Edit Field Details panel. Type **3** for internal length and press Field Exit.

```

EDIT FIELD DETAILS                               My model
Field name . . . . : Finished race               Document'n seq. :
Type . . . . . : NBR                               Field usage: DRV
Internal length. . . : 3      Data type : P      GEN name: ACNB
                                   K'bd shift: -
Headings. . . . . :-                               Old DDS name:
Text . . . . . : Finished race
Left hand side text. : Finished race
Right hand side text : Number
Column headings. . . : Finished
                                   race
Control . . . . . :-
Default condition : *NONE
Check condition . . : *NONE
Modulus 10/11. . . . :
Edit codes. . . . . :-      Mask input edit code (Y, ' ')
Screen input . . . . : 4      '
Screen output. . . . : 3      '      0'
Report . . . . . : 3      '      0'
F3=Exit no update  F8=Change name/type  F9=Conditions  F10=Appearance  F20=Var

```



Press Enter. Press F3 to return to the Display Fields panel.

## Displaying the Function Fields

The Display Fields panel now has your new function fields displayed. Note that you need to press Roll Up to view the No. of finishers function field.

Type **P** against Finished race as shown to edit its parameters.

DISPLAY FIELDS			My model			
?	Field name	Type	REF	(*ZERO) Length	(*BLANK) Field name	Field usage
-	Course code	CDE		6	ABCD	CDE
-	Course name	TXT		25	ABTX	ATR
-	Dam Date of birth	DT#	REF	10	ADD2	ATR
-	Dam Horse code	CDE	REF	6	AFCO	CDE
-	Dam name	TXT	REF	25	AFTX	ATR
-	Date of birth	DT#		10	ACD2	ATR
-	Distance	QTY		5.0	AAOT	ATR
-	Entry number	CDE		6	ACCD	CDE
-	Entry Status	STS		1	ACST	ATR
P	Finished race	NBR		3.0	ACNB	DRV
■	Finishing position	NBR		5.0	ABNB	ATR
-	Going conditions	STS		1	ABST	ATR
-	Handicap	QTY		5.0	ABOT	ATR

SEL: P=Parameters, N=Narrative  
X=Select.  
F3=Exit F5=Reload F10=Define field

Press Enter to display the Edit Function Parameters panel.

## Defining Function Field Parameters

Next you are going to specify that Entry Status is an input parameter to the Finished race function field. You will use the same Edit Function Parameters panel you used to specify function parameters earlier in this tutorial.

Type the details for the Entry Status parameter as shown. The usage of this parameter defaults to input. Check the parameter usage by also typing **Z** against the field.

```

EDIT FUNCTION PARAMETERS          My Model
Function name. . : Finished race   Type : Derived function field
Received by file : *FIELD         Acpth:
                                     Passed
? File/*FIELD                     Access path/Field      as Seq
- *FIELD                          Finished race         FLD
Z *FIELD                          Entry Status           FLD █
-                                _____
-                                _____
-                                _____
-                                _____
-                                _____
-                                _____
-                                _____
-                                _____
                                     Values
                                     FLD: One parameter per field
                                     RCD: One parameter for all fields
                                     KEY: One parameter for key fields only

SEL: Z-Details (field selection).
F3=Exit F5=Reload

```

Press Enter. The Edit Function Parameter Details panel displays showing the Usage for Entry Status as I (input).

```

EDIT FUNCTION PARAMETER DETAILS    My model
Function name. . : Finished race   Type : Derived function field
Received by file : *FIELD         Acpth: Entry Status
Parameter (file) : *FIELD         Passed as: FLD
? Field                               Usage  Role
  Entry Status                       I
                                     _____
                                     _____
                                     _____
                                     _____
                                     _____
                                     _____
                                     _____
                                     _____

SEL: Usage: I-Input, O-Output, B-Both, N-Neither, D-Drop.
      Role: R-Restrict, M-Map, V-Vary length, P-Position.
F3=Exit

```

Press F3 to return to the Edit Function Parameters panel.

The derived (DRV) function field itself, Finished race, is automatically defined as an output parameter. Check this by displaying the parameter details. Type **Z** against the Finished race field to display the Edit Function Parameter Details panel. Note that the Usage for this parameter is O (output).

Press F3 twice to exit and return to the Display Fields panel.

## Selecting the Function Field

You have just finished declaring the function fields in CA 2E. After returning to the Display Fields panel, you need to select the Finished race function field for inclusion on the device design. First, type **Finished** in the positioner line and press Enter to position to the new function fields.

Type **X** next to Finished race.

DISPLAY FIELDS			My model			
			(*ZERO)		(*BLANK)	
?	Field name	Type	REF	Length	Field name	Field usage
	<b>Finished</b>					
X	Finished race	NBR		3.0	ACNB	DRV
	Finishing position	NBR		5.0	ABNB	ATR
	Going conditions	STS		1	ABST	ATR
-	Handicap	QTY		5.0	ABQT	ATR
-	Horse code	CDE		6	ADCD	CDE
-	Horse gender	STS		1	ADST	ATR
-	Horse name	TXT		25	ADTX	ATR
-	Horse value	VAL		11.2	ABVA	ATR
-	Jockey code	CDE		6	AECD	CDE
-	Jockey gender	STS		1	AEST	ATR
-	Jockey name	TXT		25	AETX	ATR
-	No. of finishers	NBR	REF	3.0	ADNB	SUM
-	Prize money	VAL		11.2	AAVA	ATR
						+

SEL: P=Parameters, N=Narrative  
X=Select.  
F3=Exit F5=Reload F10=Define field

Press Enter.

## Accepting the Parameters

The Edit Action - Function Details window is displayed showing the parameters for the Finished race function field. Note that the input parameter is Entry Status, the output parameter is Finished race, and both parameters have the RCD (subfile record) context.

```

DISPLAY FIELDS                                My model

                                           (*ZERO)  (*BLANK)

? F -----
F : EDIT ACTION - FUNCTION DETAILS
X F : Function file : *FIELD
F : Function. . . : Finished race
- G : Obj
- H : IOB Parameter Use Typ Ctx Object Name
- H : 0 Finished race FLD RCD Finished race
- H : I Entry Status FLD RCD Entry Status
- H :
- H :
- J :
- J :
- J :
- H : F3=Exit F9=Edit parameters F10=Default parms F12=Previous
- P : Some parameters have been defaulted. Press ENTER to accept

SEL: -----
X-Select.
F3=Exit F5=Reload F10=Define field

```

Press Enter to accept the default parameters and return to the device design.

The Modified Device Design

After returning to the device design, note that the Finished race function field has been added to the subfile record.

FilefUnctionSelectorHelp

\*PROGRAM\*PGMMOD

DD/MM/YY HH:MM:SS

Edit Race and Entries

Course code . . . 00000000000000000000000000000000

Race date . . . . . Name .

Going conditions \_ Distance . Prize money .

Select items, then select an action.

Opt	Entry number	Horse code	Horse name	Finishing position	Entry Status	H'cap	Fin race
			00000000000000000000000000000000		-		666
	Jockey:		00000000000000000000000000000000		-		666
-	Jockey:		00000000000000000000000000000000		-		666
-	Jockey:		00000000000000000000000000000000		-		666
-	Jockey:		00000000000000000000000000000000		-		666

F3=Exit F5=Reset F9=Open F10=Actions

Design exceeds device size limits for function Edit Race and Entries

Note that since the device design exceeds the limits of the panel, you will need to adjust this device design later in the tutorial.

Press F3 to exit and display the Edit Function Devices panel.

Defining a Function Field Action Diagram

You will now define the action for the user-defined (DRV) function field, Finished race. This will require entering logic into the action diagram for the Edit Race and Entries function.

Press F5 to view the action diagram for the Edit Race and Entries function.

EDIT FUNCTION DEVICES		My model
Function name. . .	Edit Race and Entries	Type : Edit transaction
Received by file :	Race	Acph: Races and Entries
Screen title..... ? Title		
█ Edit Race and Entries		
SEL: Z=Scr/rpt design, N=Narrative, A=Animate		
F3=Exit F5=Action diagram F15=Open Functions		

Press F5 to view the Action Diagram User Points.

EDIT ACTION DIAGRAM		Edit	MYMDL	Race
FIND=>				Edit Race and Entries
█	> Edit Race and Entries			
---	.---			
---	. ...Initialize			<--
---	. > Conduct program process			
---	. .REPEAT WHILE			
---	. .-*ALWAYS			
---	. ...Initialize screen			<--
---	. > Conduct screen conversation			
---	. .REPEAT WHILE			
---	. .-*Transaction continues			
---	. Display screen			
---	. ...Process response			<--
---	. .-ENDWHILE			
---	. .-ENDWHILE			
---	. ...Closedown			<--
---	.---			
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward				
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys				

Recall that the user points are the places in the action diagram that you can change.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Race
FIND=>                        Edit Race and Entries

___ > Edit      ACTION DIAGRAM EXIT POINTS      F3=Exit      SEL:X,Z-Select.
___ .--        USER: Validate header non-key fields      -
___ ...I        USER: Validate header non-key relations  <--
___ > Co        USER: Validate subfile record fields
___ .RE        USER: Validate subfile record relations
___ .-#A        Z CALC: Subfile record function fields      <<<
___ .          █ CALC: Header function fields              <--
___ .          > USER: Validate totals
___ .          = USER: Create header DBF record            <<<
___ .          - USER: Delete header DBF record            + <<<
___ .          .Process response                          <--
___ .          -ENDWHILE
___ .          -ENDWHILE
___ .          ...Closedown                                <--
___ .          .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Roll Up to see the next set of user points.

Type **Z** against the CALC: Subfile record function fields user point.

Press Enter. Note that CA 2E automatically inserted the Finished Race function field in the action diagram for the Edit Race and Entries function.

Type **Z** against Finished Race as shown to insert logic for the function field.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Race
FIND=>                        Edit Race and Entries

___ > CALC: Subfile record function fields
___ .--
___ Z . Finished race          *FIELD              <<<
___ .--                          <<<

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

## Adding a CASE Construct

The action diagram for the function field Finished race will be displayed. Note that you are now editing the Finished race function as shown in the upper right corner of the panel. You will use the Action Diagram Editor to insert the following conditions and actions for the Finished race function field.

```

CASE
  Condition 1  Does Entry Status = Finished?
  Action 1    Yes, set Finished race to 1
  Condition 2  *OTHERWISE
  Action 2    Set Finished race to 0
ENDCASE

```

Type **IC**.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      *FIELD
FIND=>                   Finished race

> Finished race
IC :--
--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Adding an Action

Insert an action into the action diagram. Type **IA**.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      *FIELD
FIND=>                   Finished race

> Finished race
-- :--
-- :--CASE
IA :--!!! New condition
-- :--ENDCASE
-- :--

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Adding a Second Condition

Add another condition to the CASE construct by typing **IX** against the first !!! New condition as shown. Recall that IX inserts a new condition within a CASE construct.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      *FIELD
FIND=>                   Finished race

___ > Finished race
___ .--
___ .-CASE                                     <<<
IX .-!!! New condition                         <<<
___ .-!!! Undetermined action                 <<<
___ .-ENDCASE                                <<<
___ .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

## Adding a Second Action

Add another action to the CASE construct by typing **IA** against the second !!! New condition.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      *FIELD
FIND=>                   Finished race

___ > Finished race
___ .--
___ .-CASE                                     <<<
___ .-!!! New condition                         <<<
IA .-!!! Undetermined action                 <<<
___ .-!!! New condition                         <<<
___ .-ENDCASE                                <<<
___ .--

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.



## Defining the Conditions and Actions

To define the conditions and actions, type **F** against them all, and press Enter. Recall that F lets you edit the action or condition details for a line in the action diagram.

Type **F** against the conditions and actions.

```

EDIT ACTION DIAGRAM      Edit      MYMDL      *FIELD
FIND=>                   Finished race

--- > Finished race
--- .-.-
F .-.-CASE <<<
F .-.-!!! New condition <<<
F .-.-!!! Undetermined action <<<
F .-.-!!! New condition <<<
F .-.-!!! Undetermined action <<<
F .-.-ENDCASE <<<
--- .-.-

F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

## Specifying the First Condition

The first condition specifies that the action is only to be executed if the Entry Status is Finished.

Type the details as shown. Type **PAR** (parameter) for the context, **Entry Status** for the Field, and **?** in the condition field to view a list of the available conditions.

EDIT ACTION DIAGRAM		Edit	MYMDL	*FIELD
FIND=>				Finished race
____ > Finished race ____ ____ F ____ F ____ F ____ F ____ ____ ____		<b>EDIT ACTION - CONDITION</b> Title. : <b>!!! New condition</b> Context.Field . . . : <b>PAR Entry status</b> Condition . . . . . : <b>?</b> OR Comparison. . . . . : Context.Field . . . : F3=Exit F7=Edit Compound Condition		
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys				

Press Enter.

## Defining Field Conditions for the Status Field

If you completed the earlier exercise of defining conditions for the Entry Status field, the Edit Field - Condition window panel will include the list of conditions shown here. If you have not already done so, you can add them at this point.

Type **X** against the condition Finished.

EDIT FIELD CONDITIONS		My model	
Field name. . . . . : <b>Entry Status</b>		Attr. : STS	
Enter condition . . . : _____		and type to add new condition.	
type . . . : _____ (Type: LST, VAL)			
? Condition	Type Op File/From value	Display/To value	MN
- *ALL values	LST **		
- Disqualified	VAL D	D	
<b>X Finished</b>	VAL F	F	
- Not yet run	VAL W	W	
- Scratched	VAL S	S	
SEL: Z-Details, D-Delete, X-Select, U-Where used, N-Narrative. F3=Exit			

Press Enter twice.



Press Enter.

## Specifying the Second Condition

Because you typed **F** against the second new condition in the action diagram CASE construct, CA 2E displays the Edit Action - Condition window. The second condition specifies the action to be executed in all cases other than that specified in the first condition. A special value of **\*OTHERWISE** is used for this case.

Remove all the ? and specify **\*OTHERWISE** for the condition.

EDIT ACTION DIAGRAM	Edit	MYMDL	*FIELD
FIND=>			Finished race
> Finished race	EDIT ACTION - CONDITION		
---	Title. : !!! New condition		
---	Context.Field . . . :		
F	---	---	---
F	---	---	---
F	---	---	---
F	---	---	---
---	Comparison. . . . :		
---	Context.Field . . . :		
---	F3=Exit F7=Edit Compound Condition		
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward			
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys			

Press Enter.

## Defining the Second Action

Because you typed **F** against the second action, CA 2E displays the Edit Action - Function Name window. For the second action, you will specify the action to take when the first condition is not true. The action is to return a value of zero in the Finished race function field. You will again use the **\*MOVE** built-in function.

Leave the Function file field blank and specify the **\*MOVE** function.

Press Enter to enter parameter details for the \*MOVE function.

Type the details for the \*MOVE parameters.

Press Enter to accept the parameters and return to the Action Diagram Editor.

The completed action diagram should look like this.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      *FIELD
FIND=>                        Finished race

___ > Finished race
___ .-
___ .-CASE                      <<<
___ .-PAR.Entry Status is Finished <<<
___ .-PAR.Finished race = COH.1   <<<
___ .-*OTHERWISE                  <<<
___ .-PAR.Finished race = COH.*ZERO <<<
___ .-ENDCASE                     <<<
___ .-

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press F13 to exit the action diagram for the Finished race function field.

Accept the default of **Y** to save the Finished race function field.

```

EXIT FUNCTION DEFINITION      My model

Type choices, press Enter.

Change/create function. . . . Y                Y=Yes, N=No
  Function name . . . . . Finished race          Name
  Access path name. . . . . *H/A                 Name
  File name . . . . . *FIELD                     Name
  Function type . . . . . Derived function field

Print function. . . . . N                Y=Yes, N=No
Return to editing . . . . . N                Y=Yes, N=No

F5=Refresh  F12=Cancel  F15=Open Functions

```

Press Enter to return to the action diagram for the Edit Race and Entries function.

EDIT ACTION DIAGRAM		Edit	MVMDL	Race
FIND=>				
> CALC: Subfile record function fields				
---		---	---	<<<
---		Finished race	*FIELD	<<<
---		---	---	
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys Function 'Finished race' has been saved.				

Press F13 to exit the action diagram for the Edit Race and Entries function.

Accept the default of **Y** to save the function definition.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<b>Y</b>	Y=Yes, N=No
Function name . . . . .	<b>Edit Race and Entries</b>	Name
Access path name. . . . .	<b>Races and Entries</b>	Name
File name . . . . .	<b>Race</b>	Name
Function type . . . . .	Edit transaction	
Print function. . . . .	<b>N</b>	Y=Yes, N=No
Return to editing . . . . .	<b>N</b>	Y=Yes, N=No
Submit generation . . . . .	<b>N</b>	Y=Yes, N=No
<b>F5=Refresh F12=Cancel F15=Open Functions</b>		

Press Enter.

## Returning to the Device Design

You have now finished defining the Finished race function field. Return to the device design to define the No. of finishers function field.

Type **S** against the Edit Race and Entries function.

[illegible]

Press Enter.

## Adding the Second Function Field

Add the second function field to the header format (subfile control). Position the cursor on the field before the desired location for the function field, in this case Prize money, and press F19.

```

File      Function      Selector      Help
-----
*PROGRAM      *PGMMOD

                                Edit Race and Entries
                                DD/MM/YY  HH:MM:SS

Course code . . .      _____  000000000000000000000000
Race date . . .      _____  Name . _____
Going conditions      _ Distance . _____ Prize money . _____

Select items, then select an action.

Opt  Entry  Horse  Horse  Finishing  Entry  H'cap  Fini
     number code name      position  Status      race
-    -      -      -      -          -      -      -
Jockey: _____  00000000000000000000000000000000  _____  -  _____  666
-    Jockey: _____  00000000000000000000000000000000  _____  -  _____  666
-    Jockey: _____  00000000000000000000000000000000  _____  -  _____  666
-    Jockey: _____  00000000000000000000000000000000  _____  -  _____  666
Jockey: _____  00000000000000000000000000000000  _____  -  _____  666

F3=Exit  F5=Reset  F9=Open  F10=Actions
Design exceeds device size limits for function Edit Race and Entries

```

## Specifying the Second Function Field

The required field already exists; as a result, you can type the name No. of finishers directly. Alternatively you could type ? to obtain a selection list.



Type the function field name No. of Finishers.

```

EDIT DEVICE FUNCTION FIELD          My model
Format. . . . . : Subfile control.
Field name . . . : No. of finishers ( '?' to select)

F3=Exit  F5=Parameters

```

Press Enter.

## Defining Parameters for the No. of finishers Function Field

The parameters for the No. of finishers function field are defaulted. The input parameter is Finished race from the RCD (subfile record) context; the output parameter is the No. of finishers function field itself in the CTL (subfile control) context. Recall that you defined the No. of finishers function field with the SUM field usage.

This causes the values in the input parameter field (Finished race) for each detail record to automatically be added together and the sum to be placed in the output parameter (No. of finishers).

```

EDIT DEVICE FUNCTION FIELD          My model
Format. . . . . : Subfile control.
Fiel : EDIT ACTION - FUNCTION DETAILS
      : Function file : *FIELD
      : Function. . . : No. of finishers
      : IOB Parameter      Use Typ Ctx Object Name
      : 0 No. of finishers      FLD CTL No. of finishers
      : I Finished race        FLD RCD Finished race
      :
      : F3=Exit  F9=Edit parameters  F10=Default parms  F12=Previous
      : Some parameters have been defaulted. Press ENTER to accept
      :
F3=Exit  F5=Parameters

```

Press Enter to accept the default parameters as shown.

## The Added Function Field

The No. of Finishers field will be added to the subfile control format.

File	fUnction	Selector	Help				
*PROGRAM	*PGMMOD		DD/MM/YY HH:MM:SS				
<b>Edit Race and Entries</b>							
Course code . . . .	_____	00000000000000000000000000000000					
Race date . . . . .	_____	Name . _____					
Going conditions . .	_____	Distance . _____	Prize money . _____ No. of _____				
Select items, then select an action.							
Opt	Entry number	Horse code	Horse name	Finishing position	Entry Status	H'cap	Fin. race
			00000000000000000000000000000000				666
	Jockey: _____		00000000000000000000000000000000				
-	Jockey: _____		00000000000000000000000000000000				666
-	Jockey: _____		00000000000000000000000000000000				666
-	Jockey: _____		00000000000000000000000000000000				666
-	Jockey: _____		00000000000000000000000000000000				666
-	Jockey: _____		00000000000000000000000000000000				666
F3=Exit F5=Reset F9=Open F10=Actions							
<b>Design exceeds device size limits for function Edit Race and Entries</b>							

## Readjusting the Device Design

Adjust the device design so that all fields fit on the panel. Move the No. of finishers function field to a new line and hide the Finished race function field so that it is not displayed.

To do the latter, position the cursor on the field and press Enter. The Edit Screen Entry Details panel displays. Change the I/O usage for the field to hidden (H).

EDIT SCREEN ENTRY DETAILS		My model
Field name . . . . .	Finished race	Display length . . . : 3
GEN name . . . . .	ACHB	
Label location . . . .	C (Above,Before,Column,blank)	Label spacing. : _
Lines before . . . . .	_____	
Spaces before . . . . .	2	Screen text. . . . : E (M, L, F)
Column Headings. . . .	Finished race	
Left hand side text. :	Finished race	
Right hand side text :	Number	
Display RHS text . . .	RHS spaces . . . . : _____	Fill LHS text. . . . : Y
I/O Usage. . . . .	H Edit codes Output. . : 3	Input: 4
Mask input edit code :	N	
Check condition . . .	*NONE	
Allow zero . . . . .	_____	Field exit option. . : 2
F3=Exit, no update F7=Relations F18=Screen attributes		

Press Enter. The final device design should look like this:

```

File  fFunction  Selector  Help
-----
*PROGRAM  *PGMMOD                                DD/MM/YY HH:MM:SS

Edit Race and Entries
Course code . . .      _____ 000000000000000000000000
Race date . . . .      _____ Name . _____
Going conditions _____ Distance . _____ Prize money . _____
No. of finishers 666

Select items, then select an action.

Opt Entry Horse Horse Finishing Entry H'cap
number code name position Status
- _____ 000000000000000000000000 _____ - _____
  Jockey: _____ 000000000000000000000000 _____ - _____
- _____ 000000000000000000000000 _____ - _____
  Jockey: _____ 000000000000000000000000 _____ - _____
- _____ 000000000000000000000000 _____ - _____
  Jockey: _____ 000000000000000000000000 _____ - _____
- _____ 000000000000000000000000 _____ - _____
  Jockey: _____ 000000000000000000000000 _____ - _____

F3=Exit  F5=Reset  F9=Open  F10=Actions

```

Press F3 to go to the Edit Function Devices panel.

## Exiting the Device Design

You have finished creating the function Edit Race and Entries and designed an appropriate panel layout.

```

EDIT FUNCTION DEVICES                                My model

Function name. . : Edit Race and Entries      Type : Edit transaction
Received by file : Race                        Acpth: Races and Entries

? Title
Screen title..... Edit Race and Entries

SEL: Z-Scr/rpt design, N-Narrative, A-Animate
F3=Exit  F5=Action diagram  F15=Open Functions

```

Exit from the Edit Function Devices panel by pressing F3. The Exit Function Definition panel displays.

## Submitting the Function for Generation

In addition to saving changes to the function, you can use the Exit Function Definition panel to submit the function and display file for generation by setting the Submit generation field to Y.

Accept the default of Y for Change/create function to save your changes and change the Submit generation field to **Y**.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<b>Y</b>	Y=Yes, N=No
Function name . . . . .	<b>Edit Race and Entries</b>	Name
Access path name. . . . .	<b>Races and Entries</b>	Name
File name . . . . .	<b>Race</b>	Name
Function type . . . . .	Edit transaction	
Print function. . . . .	<b>N</b>	Y=Yes, N=No
Return to editing . . . . .	<b>N</b>	Y=Yes, N=No
Submit generation . . . . .	<b>Y</b>	Y=Yes, N=No
F5=Refresh F12=Cancel F15=Open Functions		

Press Enter.

You will be returned to the Edit Functions panel. Messages indicating that the function has been saved and submitted for generation will appear at the bottom of the panel.

EDIT FUNCTIONS		My model
File name. . . : Race		
** 1ST LEVEL **		
? Function	Function type	Access path
- Change Race	Change object	Update index
- Create Race	Create object	Update index
- Delete Race	Delete object	Update index
- Edit Race	Edit file	Retrieval index
- Edit Race and Entries	Edit transaction	Races and Entries
- Select Race	Select record	Retrieval index
-		
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Marr, O-Open,		
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.		
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		
Function 'Edit Race and Entries' has been saved.		

You have now created a program that allows the end user to update the two files, RACE and RACE ENTRY, simultaneously.

## Exercises

Do the following exercises:

1. Submit the request for generation and compilation for the Races and Entries SPN access path and the Edit Races and Entries function. You do this by selecting the Submit model create request (YSBMMDLCRT) option from the Display Services Menu (F17). For a program to compile successfully, all files and access paths used by the program must have been generated and compiled first.

**Note:** Be sure to write down the implementation name for the Edit Races and Entries function. You will need this name for testing the program using the CALL command.

2. Call the Edit Races and Entries function and add a few races to the database and some entries for each race. When you call the program, remember to specify the null or blank parameter for the return code. For example,

call myaretr ''

Try pressing F4 on fields like *Horse code* and *Jockey code* to view a selection list of available entries.

**Note:** CA 2E automatically provides this prompting capability for file-to-file relations, such as, RACE ENTRY Refers to HORSE. When the end user types a ? or user prompt (F4) for a key or foreign key field, by default CA 2E calls the related Select record function, such as, Select Horse. A function used in this way is known as a *prompt function*.

3. Call the Edit Horse program and test the link you defined between the Edit Horse program and the Display Racing Results program. You should be able to select a horse and display its racing history.

For example, the steps required to call Display Racing results for Faithful Dobbin are:

- a. Type **/** in Subfile selector for Faithful Dobbin.
- b. Press F10 to access the action bar.
- c. Type **S** to access the Selector Choice menu.
- d. Type **1** beside the Display Racing Results action.
- e. Press Enter.
- f. Press F3 to return to the Edit Horse function.



# Chapter 8: Report Functions

---

This chapter introduces the two functions used to produce reports. These are:

- The Print File (PRTFIL) Function
- The Print Object (PRTOBJ) Function

You will use these functions to define the two following reports:

- A simple report of all the horses recorded in the database.
- A more complex report showing all the horses in the database and for each horse, a list of the races in which the horse competed.

## Introduction to the Print File Function

In this topic you will learn how to define a Print File function to produce a simple report of all the horses recorded in the database.

New terms introduced

- PRTFIL (Print File) function
- CNT (count) function field
- CUR (current report format) report context
- NXT (next report format) report context

New displays introduced

- Edit Report Design
- Display Report Formats
- Edit Report Format Details

## Objectives

You will be producing a report that lists all the horses recorded in your database. In addition, the report will give the total number of horses recorded and the total value of all the horses. The following steps are involved:

1. Create a Print Horses function based on the HORSE file. The function will require the creation of a RSQ (resequence) access path to retrieve records in *Horse name* order.
2. Specify the *Total horse value* and *Total number of horses* function fields on the report design.
3. Modify the layout of the report, as required.
4. Generate and compile the source for the access path and the report.
5. Test the implemented function.

## Defining the Print File Function

Starting from the Edit Functions panel for the HORSE file, specify the Print Horses function. A Print File (PRTFIL) function is defined the same way as other functions. Type **Print Horses** for the Function name, **PRTFIL** for the Function type, and **?** in the access path field to view a list of existing access paths.

EDIT FUNCTIONS		My model
File name. . . : Horse		** 1ST LEVEL **
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
- Edit Horse	Edit file	Retrieval index
- Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
- <b>Print Horses</b>	<b>PRTFIL</b>	<b>?</b>
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Harr, O-Open,		
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.		
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press Enter. The Edit File Details panel displays as a result of entering **?** for the Access path.

## Specifying a New Access Path

Use the Edit File Details panel for the HORSE file to create a new access path that will cause the report to print horse details in alphabetical order by Horse name, rather than in Horse code order. To achieve this, specify a new access path of type RSQ called Horse name order.

Type **RSQ** for the access path type, **Horse name order** for the Access path name, and **Z** to zoom into the access path.



EDIT FILE DETAILS

My model

File name : Horse

Attribute : REF

Documentation sequence :

GEN format prefix : AE

Assimilated physical :

Record not found message : Horse

Record exists message : Horse

Field reference file : \*NONE

Source library : MYGEN

Distributed : N (Y,N)

HF Msgid : USR0007

EX Msgid : USR0008

? Typ Access path

PHY Physical file

UPD Update index

RTV Mares

RTV Retrieval index

RTV Stallions

RSQ Horse name order

Source mbr

Key

Index options

Auto add

MYAREP NONE

MYAREL0 UNIQUE IMMED

MYAREL2 UNIQUE IMMED DYNHLT

MYAREL1 UNIQUE IMMED

MYAREL3 UNIQUE IMMED DYNHLT

ATR ONLY

ATR ONLY

ATR ONLY

ATR ONLY

ATR ONLY

SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock

H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., H-Narrative

F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative

Press Enter.

Adding Virtual Entries to the Access Path

Remember that virtual entries are not automatically added to an access path. As a result, you need to add the Dam name, Dam Date of birth, Sire name, and Sire Date of birth virtual fields explicitly so they will appear on the report you are designing.

Type **V** against the access path format.

EDIT ACCESS PATH DETAILS

My model

File name : Horse

Access path name : Horse name order

Unique or duplicate order : F (U-Unique,F-FIFO,L-LIFO,C-FCFO, ' '-Undefined)

Index maintenance option : I (I-IMMED, D-DLY, R-REBLD)

Alternate collating table :

Allow select/omit : (S-Static, D-Dynamic, ' '-None)

Generation mode : M (M-MDLVAL, D-DDS, S-SQL, X-UNX)

Source member name : MYAREL4

Source member text : Horse

Attribute : REF

Type : RSQ

Format

GEN

Format text

Associated

? Seq name

pxf

(Based on file)

Retrieval access path

V 1 VAEREAG AE Horse Retrieval index

SEL: Z-Entries, R-Relations, S-Select/omit, A-Assoc.acps, T-Trim, V-Virtualize

F3=Exit F8=Rename F20=Narrative

Press Enter to display the Virtualize Access Path panel.

VIRTUALIZE ACCESS PATH		My model					
File name . . . . .	: Horse						Attribute . : REF
Access path . . . . .	: Horse name order						Type . . . : RSQ
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Dam name	TXT REF		V		AFTX	25	Y
Dam Date of birth	DT# REF		V		ADD2	10	Y
Sire name	TXT REF		V		AGTX	25	Y
Sire Date of birth	DT# REF		V		AED2	10	Y
F3=Exit, no update ENTER=Validate							

Press Enter twice to validate and confirm the virtual entries displayed. Notice the message at the bottom of the Edit Access Path Details panel.

## Edit Access Path Details

You used the Edit Access Path Details panel earlier in this tutorial to specify a RSQ access path. You will use the same process here to specify a new key order on the access path format.

Type **Z** against the access path format.

EDIT ACCESS PATH DETAILS		My model	
File name . . . . .	: Horse	Attribute . : REF	
Access path name . . . . .	: Horse name order	Type . . . : RSQ	
Unique or duplicate order :	(U-Unique,F-FIFO,L-LIFO,C-FCFO, ' '-Undefined)		
Index maintenance option :	I (I-IMMED, D-DLV, R-REBLD)		
Alternate collating table :	(S-Static, D-Dynamic, ' '-None)		
Allow select/omit . . . . .	(M-MDLVAL, D-DDS, S-SQL, X-UNX)		
Generation mode . . . . .	M (M-MDLVAL, D-DDS, S-SQL, X-UNX)		
Source member name . . .	HYAEREL4		
Source member text . . .	Horse	Horse name order	
Format	GEN	Format text	Associated
? Seq name	px	(Based on file)	Retrieval access path
Z 1 VAEREAG	AE	Horse	Retrieval index
SEL: Z=Entries, R=Relations, S=Select/omit, A=Assoc.acps, T=Trim, V=Virtualize			
F3=Exit F8=Rename F20=Narrative			
Format 'Horse' of 'Horse name order' updated with all virtuals.			

Press Enter.

### Specifying Access Path Details

On the Edit Access Path Format Entries panel, specify the new key order for the RSQ access path. The initial key order default is the same as for the default Retrieval index access path.

### The New Key Order

Specify an alternative key order. For this type of access path, you can choose any field except a virtual field as a key field. Use Horse name as the key.

In the Key no. column, clear the 1 next to Horse code and type **1** against Horse name.

EDIT ACCESS PATH FORMAT ENTRIESMy model

File name . . . . . : Horse

Access path name. . . . . : Horse name order

Format text . . . . . : Horse

Based on. . . . . : Horse

Attribute . . : REF

Type. . . . : RSQ

Format No . . : 1

? Field		GEN	Name	Type	Key no.	Altcol	Ref
					no.	seq	cnt
- Horse code	CDE		ABCD	K			1
- Horse name	TXT		ADTX	A	1		1
- Horse gender	STS		ADST	A			1
- Horse value	VAL		ABVA	A			1
- Date of birth	DT#		ACDZ	A			1
- Dam Horse code	CDE REF		AFCO	A			1
- Dam name	TXT REF		AFTX	V			1
- Dam Date of birth	DT# REF		ADDZ	V			1
- Sire Horse code	CDE REF		AGCO	A			1
- Sire name	TXT REF		AGTX	V			1
- Sire Date of birth	DT# REF		AEDZ	V			1

SEL: Z=Field details, L=Locks.

F3=Exit F7=Relations

Press Enter. Press F3 twice to exit.

### Generating and Compiling the New Access Path

At the Edit File Details panel, request batch generation and compilation for the newly defined access path using the J option.

Type **J** next to the Horse name order RSQ access path.

EDIT FILE DETAILS		My model																																				
File name . . . . .	Horse	Field reference file. : *	NONE																																			
Attribute . . . . .	REF	Source library. . . . .	MYGEN																																			
Documentation sequence. . .		Distributed . . . . .	N (Y,N)																																			
GEN format prefix . . . .	AE																																					
Assimilated physical. . . .																																						
Record not found message. :	Horse	NF Msgid. :	USR0007																																			
Record exists message . . :	Horse	EX Msgid. :	USR0008																																			
<table border="1"> <thead> <tr> <th>? Typ Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>PHY Physical file</td> <td>MYAEREP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>UPD Update index</td> <td>MYAEREL0</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Mares</td> <td>MYAEREL2</td> <td>UNIQUE IMMED</td> <td>DYNSLT</td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Retrieval index</td> <td>MYAEREL1</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Stallions</td> <td>MYAEREL3</td> <td>UNIQUE IMMED</td> <td>DYNSLT</td> <td>ATR ONLY</td> </tr> <tr> <td>X RSQ Horse name order</td> <td>MYAEREL4</td> <td>UNDEFN IMMED</td> <td></td> <td>ATR ONLY</td> </tr> </tbody> </table>				? Typ Access path	Source mbr	Key	Index options	Auto add	PHY Physical file	MYAEREP	NONE		ATR ONLY	UPD Update index	MYAEREL0	UNIQUE IMMED		ATR ONLY	RTV Mares	MYAEREL2	UNIQUE IMMED	DYNSLT	ATR ONLY	RTV Retrieval index	MYAEREL1	UNIQUE IMMED		ATR ONLY	RTV Stallions	MYAEREL3	UNIQUE IMMED	DYNSLT	ATR ONLY	X RSQ Horse name order	MYAEREL4	UNDEFN IMMED		ATR ONLY
? Typ Access path	Source mbr	Key	Index options	Auto add																																		
PHY Physical file	MYAEREP	NONE		ATR ONLY																																		
UPD Update index	MYAEREL0	UNIQUE IMMED		ATR ONLY																																		
RTV Mares	MYAEREL2	UNIQUE IMMED	DYNSLT	ATR ONLY																																		
RTV Retrieval index	MYAEREL1	UNIQUE IMMED		ATR ONLY																																		
RTV Stallions	MYAEREL3	UNIQUE IMMED	DYNSLT	ATR ONLY																																		
X RSQ Horse name order	MYAEREL4	UNDEFN IMMED		ATR ONLY																																		
SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative																																						

Press Enter to request generation.

## Selecting the Access Path

Now that you have created the required access path, select it for use with the Print Horses function.

Type **X** to select the Horse name order access path.

EDIT FILE DETAILS		My model																																				
File name . . . . .	Horse	Field reference file. : *	NONE																																			
Attribute . . . . .	REF	Source library. . . . .	MYGEN																																			
Documentation sequence. . .		Distributed . . . . .	N (Y,N)																																			
GEN format prefix . . . .	AE																																					
Assimilated physical. . . .																																						
Record not found message. :	Horse	NF Msgid. :	USR0007																																			
Record exists message . . :	Horse	EX Msgid. :	USR0008																																			
<table border="1"> <thead> <tr> <th>? Typ Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>PHY Physical file</td> <td>MYAEREP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>UPD Update index</td> <td>MYAEREL0</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Mares</td> <td>MYAEREL2</td> <td>UNIQUE IMMED</td> <td>DYNSLT</td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Retrieval index</td> <td>MYAEREL1</td> <td>UNIQUE IMMED</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Stallions</td> <td>MYAEREL3</td> <td>UNIQUE IMMED</td> <td>DYNSLT</td> <td>ATR ONLY</td> </tr> <tr> <td>X RSQ Horse name order</td> <td>MYAEREL4</td> <td>UNDEFN IMMED</td> <td></td> <td>ATR ONLY</td> </tr> </tbody> </table>				? Typ Access path	Source mbr	Key	Index options	Auto add	PHY Physical file	MYAEREP	NONE		ATR ONLY	UPD Update index	MYAEREL0	UNIQUE IMMED		ATR ONLY	RTV Mares	MYAEREL2	UNIQUE IMMED	DYNSLT	ATR ONLY	RTV Retrieval index	MYAEREL1	UNIQUE IMMED		ATR ONLY	RTV Stallions	MYAEREL3	UNIQUE IMMED	DYNSLT	ATR ONLY	X RSQ Horse name order	MYAEREL4	UNDEFN IMMED		ATR ONLY
? Typ Access path	Source mbr	Key	Index options	Auto add																																		
PHY Physical file	MYAEREP	NONE		ATR ONLY																																		
UPD Update index	MYAEREL0	UNIQUE IMMED		ATR ONLY																																		
RTV Mares	MYAEREL2	UNIQUE IMMED	DYNSLT	ATR ONLY																																		
RTV Retrieval index	MYAEREL1	UNIQUE IMMED		ATR ONLY																																		
RTV Stallions	MYAEREL3	UNIQUE IMMED	DYNSLT	ATR ONLY																																		
X RSQ Horse name order	MYAEREL4	UNDEFN IMMED		ATR ONLY																																		
SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative Source generation request for MYAEREL4 accepted.																																						

Press Enter.

## The Report Design

After you select the access path, CA 2E creates the Print Horses function and returns to the Edit Functions panel. From here you can display the report layout. To do so, type **S** against the function.

Press Enter.

The default layout is made up of the following formats:

A heading format and corresponding total format is provided for each Level break. For example, for each key of the based-on access path there is a heading and total format. If the access path has a unique key, the number of level breaks is one less than the number of keys. In this case, the access path does not have a unique key. Since there is one key level, one heading format is provided that, by default, has Horse name as an output field.

The detail format contains the subfile records from the based-on access path. It is shown on the Print Horses report layout between the Horse name heading and Horse name total formats. All the fields from the access path are available in this format. The key field (Horse name) is hidden by default because it already appears on the heading format.

You can modify the report layout the same way you modified the device designs of the earlier functions. The editors for the report layout and the panel design are very similar. However, a report is by default 132 characters across and extends vertically past the bottom of the panel. You can press the Roll Up key to roll the panel by half a panel at a time.

## Displaying the Report Formats

The report formats are shown on the Display Report Formats panel. To access this panel, press F17 from the device design (report layout).

DISPLAY REPORT FORMATS		My model					
? Format	Type	Hide	Start	Space	New	Ovflw	Function
		fmt	Line	before	page	print	indent
■ Standard report header	HDR		1				
- Top of Page	TOP				1		
- First Page Format	1PG				1		
- Horse name	1HD				1		
- Detail line.	RCD				1		
- Horse name	2TL				1		
- Final totals	2TL				1		
- End of report	FTR				1		

SEL: Z-Details, H-Hide, I-Indent, '-'-Drop, '+'-Reinstate format.  
F3=Exit

- The Top of page format (TOP) enables you to define fields to be printed at the top of each page in addition to the Standard report header (HDR). This format is empty by default.
- The First page format (1PG) can be used to add title information to the beginning of the report. This format is empty by default.
- The Level-heading format (1HD) contains fields appropriate to the key level; in this case, Horse name. By default, this format prints before the first detail within a level break.

- The Detail line format (RCD) contains the records from the access path on which the Print File function is based. By default, this format is printed for every record read. This format cannot be dropped, only hidden.
- The Level total format (2TL) contains fields appropriate to the key level. By default, this format is printed after the last detail record or subtotal within a level break.
- The Final total format (ZTL) contains totals of all previous total levels. By default, this format contains only the constant Final totals; you can add your own function fields to provide totals.

## Dropping Formats from a Report Design

In this step you will remove the Horse name heading and total formats. To remove a format, you can either hide it or drop it completely. If you hide a format it will still be logically present. In this example, you will drop both formats completely.

Type '-' against the two Horse name formats.

DISPLAY REPORT FORMATS		My model						
? Format	Type	Hide	Start	Space	New	Qvflw	Function	
		fmt	Line	before	page	print	indent	
- Standard report header	HDR		1					
- Top of Page	TOP			1				
- First Page Format	1PG			1				
- Horse name	1HD			1				
- Detail line.	RCD			1				
- Horse name	2TL			1				
- Final totals	ZTL			1				
- End of report	FTR			1				

SEL: Z-Details, H-Hide, I-Indent, '-'-Drop, '+'-Reinstate format.  
F3=Exit

Press Enter. Notice the D in the Hide fmt column for the two formats you dropped.

## Adding Function Fields

In this step, you will add two function fields to the Final totals format (ZTL).

Type **Z** against Final totals format.

DISPLAY REPORT FORMATS		My model					
? Format	Type	Hide	Start	Space	New	Ovflw	Function
		fmt	Line	before	page	print	indent
- Standard report header	HDR		1				
- Top of Page	TOP			1			
- First Page Format	IPG			1			
- Horse name	1HD	D		1			
- Detail line.	RCD			1			
- Horse name	2TL	D		1			
- Final totals	ZTL			1			
- End of report	FTR			1			

SEL: Z-Details, H-Hide, I-Indent, '-'-Drop, '+'-Reinstate format.  
F3=Exit

Press Enter to display the Edit Report Format Details screen for the Final totals format.

EDIT REPORT FORMAT DETAILS		My model	
Format . . . . .	: <u>Final totals</u>	Type:	ZTL
Skip to new page . . . . .	: <u>1</u>	Print on forms overflow. . . . .	: -
Blank lines before fmt . . . . .	: <u>1</u>	or Fixed start line no. . . . .	: -
Indentation type . . . . .	: -	Relative . . . . .	: -
		Absolute . . . . .	: -

SEL: Z-Details, A,B,C,D-Text position, I,O,H,'-'-Field usage.  
F3=Exit F7=Fmt rel F10=Sequence F19=Add function field F23=Add constant

Press F19 to add a function field to this format using the Edit Device Function Fields panel.

## Displaying Existing Function Fields

To display existing function fields, type ? for Function field.



EDIT DEVICE FUNCTION FIELD

My model

Format. . . . . : Final totals

Field name . . . . : ? ( '?' to select)

F3=Exit F5=Parameters

Press Enter.

All fields defined in your model are displayed, including the two function fields you defined for the Edit Race and Entries function (EDTTRN).

DISPLAY FIELDS

My model

?	Field name	Type	REF	(*ZERO) Length	(*BLANK) Field name	Field usage
-	Course code	CDE		6	ABCD	CDE
-	Course name	TXT		25	ABTX	ATR
-	Dam Date of birth	DT#	REF	10	ADD2	ATR
-	Dam Horse code	CDE	REF	6	AACD	CDE
-	Dam name	TXT	REF	25	AFTX	ATR
-	Date of birth	DT#		10	ACD2	ATR
-	Distance	QTY		5.0	AAQT	ATR
-	Entry number	CDE		6	ACCD	CDE
-	Entry Status	STS		1	ACST	ATR
-	Finished race	NBR		3.0	ACHB	DRV
-	Finishing position	NBR		5.0	ABNB	ATR
-	Going conditions	STS		1	ABST	ATR
-	Handicap	QTY		5.0	ABQT	ATR

SEL: P-Parameters, N-Narrative  
X-Select.  
F3=Exit F5=Reload F10=Define field

To define two new function fields, press F10.

Defining New Function Fields

The first new function field, Total horse value, will provide a total of all the Horse value fields on the detail format. A function field of type SUM is used for this purpose. Recall that you used a function field of this type previously to calculate the Total Number of finishers in a race.

There are two ways to define a function field:

- You will use the REF field method to define the Total horse value function field. However, you cannot use this method for the Total number of horses function field since the CNT field must be numeric and the field you are counting is alphanumeric (CDE).

[illegible]

## Specifying Function Parameters

The Total horse value function field is now fully defined. However, you must still specify the field used to arrive at the Total number of horses function. Type **P** next to this function field as shown to specify its parameters.

DISPLAY FIELDS

My model

?	Field name	Type	REF	(*ZERO) Length	(*BLANK) Field name	Field usage
-	Sire Horse code	CDE	REF	6	AGCD	CDE
-	Sire name	TXT	REF	25	AGTX	ATR
-	Total horse value	VAL	REF	11.2	ACVA	SUM
P	Total number of horses	NBR		5.0	AENB	CNT

SEL: P-Parameters, N-Narrative  
X-Select.  
F3=Exit F5=Reload F10=Define field

Press Enter.

The Default Parameters

As mentioned previously, the function field itself is the output parameter. In this step, you need to specify an input parameter, which is the field to be counted. In this case, the Horse code field is the input parameter.

You can specify function parameters in two different ways. These can either be individual fields or an access path from which you select individual fields. In this example, you will specify the parameter as an individual field.

Type the parameter details.

EDIT FUNCTION PARAMETERS

My model

Function name. . : Total number of horses      Type : Count function field

Received by file : \*FIELD      Acpth:

?	File/*FIELD	Access path/Field	Passed as Seq
-	*FIELD	Total number of horses	FLD
-	*FIELD	Horse code	FLD

Values  
FLD: One parameter per field  
RCD: One parameter for all fields  
KEY: One parameter for key fields only

SEL: Z-Details (field selection).  
F3=Exit F5=Reload

Press Enter.

The parameter usage defaults to input. You can check this for yourself by typing **Z** against Horse code on the Edit Function Parameters panel.

Press F3 to exit.

## Selecting the Function Fields

You have finished defining the two new function fields. You must now select them to appear on the Final totals format.

Type **X** against Total Horse value.

DISPLAY FIELDS			My model		
?	Field name	Type	REF	(*ZERO) Length	(*BLANK) Field name
-	Sire Horse code	CDE	REF	6	AGCD
-	Sire name	TXT	REF	25	AGTX
X	Total horse value	VAL	REF	11.2	ACVA
-	Total number of horses	HBR		5.0	AENB
SEL: P-Parameters, N-Narrative X-Select. F3=Exit F5=Reload F10=Define field					

Press Enter.

## Confirming the Function Details

The Edit Action - Function Details window will prompt you to confirm details of the function parameters. This panel shows the contexts from which the values of the parameters will be taken: NXT (next report format) and CUR (current report format).

The values for the Horse value parameter are taken from the CUR context. This context corresponds to the format upon which the processing of the function field takes place (the Detail line format). The NXT context corresponds to the format that follows the Detail line. In this case, the NXT context is the Final totals format.

```

DISPLAY FIELDS                                My model
                                           (*ZERO)  (*BLANK)

? F
: EDIT ACTION - FUNCTION DETAILS
S : Function file : *FIELD
S : Function. . . : Total horse value
I X I T : Obj
T : IOB Parameter      Use Typ Ctx Object Name
: 0 Total horse value  FLD HXT Total horse value
: 1 Horse value        FLD OUR Horse value
:
:
:
: F3=Exit F9=Edit parameters F10=Default parms F12=Previous
: Some parameters have been defaulted. Press ENTER to accept
SEL:
X>Select.
F3=Exit F5=Reload F10=Define field

```

Press Enter to confirm the parameters. Press F13 to return to the report device design.

## Adding the Total Number of Horses Function Field

The Total horse value function field is now present on the Final totals format. To add the other function field, position the cursor on the Final totals format.

```
0000000000000000000000000000000000 Print Horses

Horse   Horse   Horse value      Date of   Dam Horse   Dam name                      Da
code    gender                                     birth    code       of

000000 0          666666666.66CR 00000000 000000     00000000000000000000000000 00
000000 0          666666666.66CR 00000000 000000     00000000000000000000000000 00
000000 0          666666666.66CR 00000000 000000     00000000000000000000000000 00

Final total15 Total horse value 666666666.66CR

** END OF REPORT **

Design exceeds device size limits
```

Press F19 to add the Total number of horses function field.

Type **Total Number of horses** in the Field name field.

```

EDIT DEVICE FUNCTION FIELD          My model
Format. . . . . : Final totals
Field name . . . : Total number of horses ('?' to select)

F3=Exit  F5=Parameters

```

Press Enter.

## Confirming the Parameters

CA 2E automatically provides defaults for the parameters and contexts. In this case you need only accept them.

```

EDIT DEVICE FUNCTION FIELD          My model
Format. . . . . : Final totals
Fiel : EDIT ACTION - FUNCTION DETAILS
      Function file : *FIELD
      Function. . . : Total number of horses
      Obj
      IOB Parameter      Use Typ Ctx Object Name
      0 Total number of horses    FLD HXT Total number of horses
      1 Horse code              FLD HOR Horse code

      F3=Exit  F9=Edit parameters  F10=Default parms  F12=Previous
      Some parameters have been defaulted.  Press ENTER to accept

F3=Exit  F5=Parameters

```

Press Enter to accept the parameters and return to the report device design.

## Exercise

Both function fields are now present on the Edit Report Format Details panel of the Final totals format. Check this yourself by pressing F17 from the report device design and typing **Z** against the Final totals format on the Display Report Formats panel. Press F13 to return to the report device design.



Update the report device design by moving fields and changing labels and label spacing until the design meets your needs. Here's an example.



```
00000000000000000000000000000000 Print Horses
```

Horse	M/F	Value	D of B
00000000000000000000000000000000	0	666666666.66CR	00000000
		Dam:	00000000000000000000000000000000 00000000
		Sire:	00000000000000000000000000000000 00000000
00000000000000000000000000000000	0	666666666.66CR	00000000
		Dam:	00000000000000000000000000000000 00000000
		Sire:	00000000000000000000000000000000 00000000
00000000000000000000000000000000	0	666666666.66CR	00000000
		Dam:	00000000000000000000000000000000 00000000
		Sire:	00000000000000000000000000000000 00000000
Final totals			
Total number of horses :		66666	
Total horse value . . . :		666666666.66CR	
** END OF REPORT **			

Note that to produce this design you will need to drop the Top of Page format (TOP) and the First Page Format (1PG) using the method you used to drop the 1HD and 2TL formats previously; hint, press F17.

When you have finished the report layout, press F3 to exit the device design.

```

EDIT FUNCTION DEVICES                                My model
Function name. . : Print Horses                      Type : Print file
Received by file : Horse                             Acpth: Horse name order

? Title
Report title. . . ■ Print Horses

```

---

```

SEL: Z=Scr/rpt design, N=Narrative, A=Animate
F3=Exit  F5=Action diagram  F15=Open Functions

```

Press F3.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<u>Y</u>	Y=Yes, N=No
Function name . . . . .	<u>Print Horses</u>	Name
Access path name. . . . .	<u>Horse name order</u>	Name
File name . . . . .	<u>Horse</u>	Name
Function type . . . . .	Print file	
Print function. . . . .	<u>N</u>	Y=Yes, N=No
Return to editing . . . . .	<u>N</u>	Y=Yes, N=No
Submit generation . . . . .	<u>N</u>	Y=Yes, N=No
F5=Refresh   F12=Cancel   F15=Open Functions		

## Saving the Report Device Design

Save the changes by accepting the defaults and pressing Enter.

## Generating and Compiling the Function

You have now finished specifying the Print Horses function. Type **J** next to this function on the Edit Functions panel and press Enter to request batch generation of the Print File and program that will implement the function.

Finally, compile the generated source for the function and the access path. Press F17 to access the Display Services Menu and select the Submit model create request (YSBMMDLCRT) option. Be sure to note the implementation name (member) of the Print Horses function so you can call it in the next step.

## Running Your Program

When the Print Horses function has successfully compiled, try it out by calling the program from a command line. For example,

```
call myaspfr ''
```

Following is an example of the resulting report showing data you entered earlier using the Edit Horse program.

Your company name here		Print Horses	
Horse	M/F	Value	D of B
Bonfire	M	5000.00	2/01/83
		Dam:	
		Sire:	
Faithful Dobbin	F	3500.00	5/31/86
		Dam:	
		Sire:	
Pegasus	M	3000.00	6/23/92
		Dam: Faithful Dobbin	5/31/86
		Sire: Bonfire	2/01/88
Final totals			
Total number of horses	:	3	
Total horse value . . .	:	11500.00	
** END OF REPORT **			

Use the i OS Work with Spooled Files (WRKSPLF) command to view or print your report.

## Introduction to the Print Object Function

You will learn how to embed a Print Object function into the existing Print File function to produce a more detailed report.

New terms introduced

- PRTOBJ (Print Object) function
- QRY (Query) access path

New displays introduced

- Edit Device Structure

## Objectives

At this point, you have created a function to print a list of all the horses stored in the database. A more useful report might also list the races for which each horse has been entered. To produce such a report, you need to combine records from the RACE ENTRY file with the existing Print Horses function. You can achieve this using a Print Object function. Use the following steps to do this.

1. Specify a Print Object function based on the RACE ENTRY file.
2. Define a Query access path to be used with the RACE ENTRY file.
3. Combine the Print Object function with the Print File function and specify the parameters to be passed between them.
4. Generate and compile the new Print Horses function.

5. Test the program.

## Defining the Print Object Function

To define a Print Object function, start from the Edit Database Relations panel and go to the Edit Functions panel on the RACE ENTRY file.

Type **F** against RACE ENTRY.

EDIT DATABASE RELATIONS				My model			
=>	Rel	lvl:		Seq	Typ	Referenced object	
? Type	Object	Relation					
— FIL	Race*	Owned by			FIL	Course	
— FIL	Race	Known by			FLD	Race date	
— FIL	Race	Known by			FLD	Race time	
— FIL	Race	Has			FLD	Race name	
— FIL	Race	Has			FLD	Going conditions	
— FIL	Race	Has			FLD	Distance	
— FIL	Race	Has			FLD	Prize money	
<b>F</b> FIL	Race Entry	Owned by			FIL	Race	
— FIL	Race Entry	Known by			FLD	Entry number	
— FIL	Race Entry	Refers to			FIL	Horse	
— FIL	Race Entry	Refers to			FIL	Jockey	
— FIL	Race Entry	Has			FLD	Finishing position	
— FIL	Race Entry	Has			FLD	Handicap	
— FIL	Race Entry	Has			FLD	Entry Status	
Bottom							
Z(n)=Details F=Functions E(n)=Entries S(n)=Select F23=More options							
F3=Exit F5=Reload F6=Hide/Show F7=Fields F9=Add/Change F24=More keys							

Press Enter.

## Adding a Print Race Entries Function

Specify the Print Object function by typing **Print Race Entries** for the function name, **PRTOBJ** for the function type, and **?** to display the available access paths.

EDIT FUNCTIONS			My model		** 1ST LEVEL **	
File name. . . : Race Entry						
? Function	Function type	Access path				
— Change Race Entry	Change object	Update index				
— Create Race Entry	Create object	Update index				
— Delete Race Entry	Delete object	Update index				
— Display Racing results	Display file	Races for a Horse				
— <b>Print Race Entries</b>	<b>PRTOBJ</b>	<b>?</b>				
—						
—						
—						
—						
—						
—						
—						
—						
More...						
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, H-Harr, O-Open,						
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.						
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services						

Press Enter to display the Edit File Details panel.

## Specifying a Query Access Path

The Edit File Details panel shows the existing access paths for the RACE ENTRY file. You will create a new access path for the Print Race Entries function.

To produce the desired report, the records in the RACE ENTRY file need to be processed in order by Horse name. To do this, you need to create an access path keyed on the Horse name field; however, Horse name is a virtual field on the RACE ENTRY file. In order to use a virtual field as a key, you need to create a query (QRY) access path. In addition, since virtual entries are not automatically included on an access path, you need to add them explicitly.

Specify a QRY access path named **Entries by Horse name** and type **V** against it.

EDIT FILE DETAILS		My model	
File name . . . . .	Race Entry	Field reference file. :	*NONE
Attribute . . . . .	CPT	Source library. . . . :	MYGEN
Documentation sequence. . .		Distributed . . . . .	N (Y,N)
GEN format prefix . . . .	AD		
Assimilated physical. . . .			
Record not found message. :	Race Entry	HF Msgid. :	USR0005
Record exists message . . :	Race Entry	EX Msgid. :	USR0006
? Typ Access path			
PHV Physical file	MYADCPP	NONE	ATR ONLY
UPD Update index	MYADCPL0	UNIQUE IMMED	ATR ONLY
RTV Retrieval index	MYADCPL1	UNIQUE IMMED	ATR ONLY
RSQ Races for a Horse	MYADCPL2	FIFO IMMED	ATR ONLY
V QRY Entries by Horse name			
SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock			
H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative			
F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative			

Press Enter to view the virtual entries that are available for inclusion on the access path; namely, Horse name and Jockey name.

VIRTUALIZE ACCESS PATH		My model					
File name . . . . .	Race Entry					Attribute . . .	CPT
Access path . . . . .	Entries by Horse name					Type . . . . .	GRY
Field	Type	Ocr	Et	Ksq	GEN name	Length	Renamed
Horse name	TXT		V		AOTX	25	
Jockey name	TXT		V		AETX	25	
F3=Exit, no update ENTER=Validate							

Press Enter twice to validate and confirm the inclusion of the two virtual entries and return to the Edit File Details panel. Note the message at the bottom of the panel indicating that the access path has been updated with all virtuals.

Type **Z** against the Entries by Horse name access path to begin the process of specifying Horse name as the key.

EDIT FILE DETAILS		My model			
File name . . . . .	Race Entry		Field reference file. : *NONE		
Attribute . . . . .	CPT		Source library. . . . : MYGEN		
Documentation sequence. . .			Distributed . . . . . : N (Y,N)		
GEN format prefix . . . . .	AD				
Assimilated physical. . . .					
Record not found message. .	Race Entry		NF	Msgid. :	USR0005
Record exists message . . .	Race Entry		EX	Msgid. :	USR0006
? Typ Access path	Source mbr	Key	Index options	Auto add	
PHY Physical file	MYADCPP	NONE		ATR ONLY	
UPD Update index	MYADCPL0	UNIQUE	IMMED	ATR ONLY	
RTV Retrieval index	MYADCPL1	UNIQUE	IMMED	ATR ONLY	
RSQ Races for a Horse	MYADCPL2	FIFO	IMMED	ATR ONLY	
<b>Z</b> GRY Entries by Horse name	MYADCPL3	FIFO		ATR ONLY	
SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative Access path 'Entries by Horse name' updated with all virtuals.					

## Displaying the Access Path Format Entries

Press Enter to display the Edit Access Path Details panel. Next, type **Z** against the displayed format and press Enter to access the Edit Access Path Format Entries panel.

```

EDIT ACCESS PATH FORMAT ENTRIES      My model
File name . . . . . : Race Entry      Attribute . : CPT
Access path name. . . : Entries by Horse name  Type. . . : QRY
Format text . . . . . : Race Entry
Based on. . . . . : Race Entry          Format No . : 1

? Field                                GEN      Key   Altcol Ref
                                Name      no.  Dsc seq  cnt
- Course code                     CDE      ABCD   K    1    -    1
- Race date                       DT#      ABDZ   K    2    -    1
- Race time                       TM#      ABT2   K    3    -    1
- Entry number                     CDE      ACCD   K    4    -    1
- Horse code                       CDE      ADCD   A    -    -    1
- Horse name                       TXT      ADTX   V    -    -    1
- Jockey code                      CDE      AECD   A    -    -    1
- Jockey name                      TXT      AETX   V    -    -    1
- Finishing position              HBR      ABNB   A    -    -    1
- Handicap                        QTY      ABQT   A    -    -    1
- Entry Status                     STS      ACST   A    -    -    1

SEL: 2-Field details, L-Locks.
F3=Exit F7=Relations

```

On the Edit Access Path Format Entries panel, the key order defaults to that of the RTV access path. Note that all fields, including virtual fields, are available as key fields.

## Specifying an Alternative Key

Specify Horse name and Race date as the alternate keys of the access path. You can assume that each horse runs in no more than one race per day; as a result, Race time is not required as a key.

Define the keys by changing entries in the Key no. column as shown; namely, blank the Key no. field for Course code, Race time, and Entry number, and type **1** for Horse name.

```

EDIT ACCESS PATH FORMAT ENTRIES      My model
File name . . . . . : Race Entry      Attribute . : CPT
Access path name. . . : Entries by Horse name  Type. . . : QRY
Format text . . . . . : Race Entry
Based on. . . . . : Race Entry          Format No . : 1

? Field                                GEN      Key   Altcol Ref
                                Name      no.  Dsc seq  cnt
- Course code                     CDE      ABCD   K    -    -    1
- Race date                       DT#      ABDZ   K    2    -    1
- Race time                       TM#      ABT2   K    -    -    1
- Entry number                     CDE      ACCD   K    -    -    1
- Horse code                       CDE      ADCD   A    -    -    1
- Horse name                       TXT      ADTX   V    1    -    1
- Jockey code                      CDE      AECD   A    -    -    1
- Jockey name                      TXT      AETX   V    -    -    1
- Finishing position              HBR      ABNB   A    -    -    1
- Handicap                        QTY      ABQT   A    -    -    1
- Entry Status                     STS      ACST   A    -    -    1

SEL: 2-Field details, L-Locks.
F3=Exit F7=Relations

```

Press Enter. Press F3 twice to exit to the Edit File Details panel.

## Compiling the QRY Access Path

The access path you have just defined must be generated and compiled before running the application. You have the option to add it to the job list now. To do so, type **J** next to the QRY access path.

EDIT FILE DETAILS		My model																															
File name . . . . .	Race Entry	Field reference file. . . . .	*NONE																														
Attribute . . . . .	CPT	Source library. . . . .	MYGEN																														
Documentation sequence. . . . .		Distributed . . . . .	N (Y,N)																														
GEN format prefix . . . . .	AD																																
Assimilated physical. . . . .																																	
Record not found message. . . . .	Race Entry	HF Msgid. . . . .	USR0005																														
Record exists message . . . . .	Race Entry	EX Msgid. . . . .	USR0006																														
<table border="1"> <thead> <tr> <th>? Typ Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>PHY Physical file</td> <td>MYADCPP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>UPD Update index</td> <td>MYADCPL0</td> <td>UNIQUE</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Retrieval index</td> <td>MYADCPL1</td> <td>UNIQUE</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td>RSQ Races for a Horse</td> <td>MYADCPL2</td> <td>FIFO</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td><b>J</b> QRY Entries by Horse name</td> <td>MYADCPL3</td> <td>FIFO</td> <td></td> <td>ATR ONLY</td> </tr> </tbody> </table>				? Typ Access path	Source mbr	Key	Index options	Auto add	PHY Physical file	MYADCPP	NONE		ATR ONLY	UPD Update index	MYADCPL0	UNIQUE	IMMED	ATR ONLY	RTV Retrieval index	MYADCPL1	UNIQUE	IMMED	ATR ONLY	RSQ Races for a Horse	MYADCPL2	FIFO	IMMED	ATR ONLY	<b>J</b> QRY Entries by Horse name	MYADCPL3	FIFO		ATR ONLY
? Typ Access path	Source mbr	Key	Index options	Auto add																													
PHY Physical file	MYADCPP	NONE		ATR ONLY																													
UPD Update index	MYADCPL0	UNIQUE	IMMED	ATR ONLY																													
RTV Retrieval index	MYADCPL1	UNIQUE	IMMED	ATR ONLY																													
RSQ Races for a Horse	MYADCPL2	FIFO	IMMED	ATR ONLY																													
<b>J</b> QRY Entries by Horse name	MYADCPL3	FIFO		ATR ONLY																													
SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative																																	

Press Enter.

## Selecting the Access Path for the Function

Now that you have defined the QRY access path, you need to select it for the Print Race Entries function. To do so, type **X** next to the QRY access path.

EDIT FILE DETAILS		My model																															
File name . . . . .	Race Entry	Field reference file. . . . .	*NONE																														
Attribute . . . . .	CPT	Source library. . . . .	MYGEN																														
Documentation sequence. . . . .		Distributed . . . . .	N (Y,N)																														
GEN format prefix . . . . .	AD																																
Assimilated physical. . . . .																																	
Record not found message. . . . .	Race Entry	HF Msgid. . . . .	USR0005																														
Record exists message . . . . .	Race Entry	EX Msgid. . . . .	USR0006																														
<table border="1"> <thead> <tr> <th>? Typ Access path</th> <th>Source mbr</th> <th>Key</th> <th>Index options</th> <th>Auto add</th> </tr> </thead> <tbody> <tr> <td>PHY Physical file</td> <td>MYADCPP</td> <td>NONE</td> <td></td> <td>ATR ONLY</td> </tr> <tr> <td>UPD Update index</td> <td>MYADCPL0</td> <td>UNIQUE</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td>RTV Retrieval index</td> <td>MYADCPL1</td> <td>UNIQUE</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td>RSQ Races for a Horse</td> <td>MYADCPL2</td> <td>FIFO</td> <td>IMMED</td> <td>ATR ONLY</td> </tr> <tr> <td><b>X</b> QRY Entries by Horse name</td> <td>MYADCPL3</td> <td>FIFO</td> <td></td> <td>ATR ONLY</td> </tr> </tbody> </table>				? Typ Access path	Source mbr	Key	Index options	Auto add	PHY Physical file	MYADCPP	NONE		ATR ONLY	UPD Update index	MYADCPL0	UNIQUE	IMMED	ATR ONLY	RTV Retrieval index	MYADCPL1	UNIQUE	IMMED	ATR ONLY	RSQ Races for a Horse	MYADCPL2	FIFO	IMMED	ATR ONLY	<b>X</b> QRY Entries by Horse name	MYADCPL3	FIFO		ATR ONLY
? Typ Access path	Source mbr	Key	Index options	Auto add																													
PHY Physical file	MYADCPP	NONE		ATR ONLY																													
UPD Update index	MYADCPL0	UNIQUE	IMMED	ATR ONLY																													
RTV Retrieval index	MYADCPL1	UNIQUE	IMMED	ATR ONLY																													
RSQ Races for a Horse	MYADCPL2	FIFO	IMMED	ATR ONLY																													
<b>X</b> QRY Entries by Horse name	MYADCPL3	FIFO		ATR ONLY																													
SEL: X-Select, Z-Details, G/J-Generate, E-STRSEU, D-Delete, O-Override, L-Lock H-Hold/Release, T-Trim, V-Virtualize, U-Usage, F-Func refs., N-Narrative F3=Exit F5=Reload F8=Change name F17=Services F20=Narrative																																	

Press Enter.





The default layout is obtained in a similar way to the previous Print File function. The access path has a non-unique key. A heading and total format are provided for each key of the access path. All the fields from the access path format are output on the detail line records, except for the key fields, which are hidden.

A PRTOBJ function is an internal function. It does not represent a complete report and must be embedded within an existing Print File function. The embedding Print File defines the standard header format, page size, and additional details for the combined report.

## Exercise - Updating the Report Layout

Update the report layout using the same techniques as those in the device design updates you completed earlier in this tutorial. Perform the following steps.

1. Display the report formats (F17) and drop the following formats: Race date header, Final totals, Horse name total, and Race date total.
2. On the Detail line format, change the usage of the Race date field from hidden to output.
3. Hide all fields that are not required and adjust labels and label spacing until the design resembles the one shown.

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<u>Y</u>	Y=Yes, N=No
Function name . . . . .	<u>Print Race Entries</u>	Name
Access path name. . . . .	<u>Entries by Horse name</u>	Name
File name . . . . .	<u>Race Entry</u>	Name
Function type . . . . .	Print object	
Print function. . . . .	<u>N</u>	Y=Yes, N=No
Return to editing . . . . .	<u>N</u>	Y=Yes, N=No
F5=Refresh F12=Cancel F15=Open Functions		

4. Press F3 twice to exit.

## Exiting the Device Design

Save your design by accepting the defaults on the Exit Function Definition panel.

EDIT FUNCTIONS		
File name. . . : Horse		
	My model	** 1ST LEVEL **
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
- Edit Horse	Edit file	Retrieval index
<b>T</b> Print Horses	Print file	Horse name order
- Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, N-Harr, O-Open,		
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.		
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		

Press Enter to return to the Edit Functions panel.

## Combining the Report Functions

You have just finished defining the two report functions. Press F3 to exit the Edit Functions panel for the RACE ENTRY file and return to the Edit Database Relations panel.

Go to the Edit Functions panel for the HORSE file. To do this, type **Horse\*** on the selection line and press Enter. Then type **F** next to any HORSE relation and press Enter.

The next step is to combine the two functions so that the records from the Print Object function (each race entry) will be printed after each detail record in the Print File function (after the horse to which it applies). To do this, you will use the Edit Device Structure panel.

Type **T** against the Print Horses function.

```
EDIT DEVICE STRUCTURE                               My model
Function name : Print Horses
█ Page Headings
  — First Page
  — KEY.Horse name Header
  — Details
  — KEY.Horse name Totals
  — Final Totals
  — End of report

SEL: IA/IB/D/Z M/C/B/A
F3=Exit F6=Cancel pending moves F14=Map
```

Press Enter.

```
EDIT DEVICE STRUCTURE                               My model
Function name : Print Horses
— Page Headings
— First Page
— KEY.Horse name Header
IA Details
— KEY.Horse name Totals
— Final Totals
— End of report

SEL: IA/IB/D/Z M/C/B/A
F3=Exit F6=Cancel pending moves F14=Map
```

## Embedding the Print Race Entries Function

The Edit Device Structure panel represents the layout of the report in terms of the formats from which it is constructed. This display is only available for functions of type PRTFIL or PRTOBJ. It has facilities for embedding PRTOBJ functions within the report and moving and copying them.

Many Print Object functions can be embedded within the same Print File function. A Print Object function can contain other embedded Print Object functions. This makes it possible to construct complex reports. In this tutorial, you will be embedding one Print Object function in the previously defined Print File function.

Insert a Print Object function after the Details format by typing **IA** (Insert After) in the Subfile selector.

```

DISPLAY PRINT FUNCTIONS                                My model
_____<= Position
File          Function
X Race Entry    Print Race Entries

SEL: X=Select
F3=Exit  F9=Create function

```

Press Enter.

## Selecting the PRTOBJ Function

The Display Print Functions panel is now shown. This panel displays a list of all the Print Object functions that are available for inclusion in the Print File function. In this case there is just one, Print Race Entries. Type **X** to select the Print Race Entries function.

```

EDIT DEVICE STRUCTURE                               My model

Function name : Print Horses

█_ Page Headings
  _ First Page
    _ KEY.Horse name Header
    _ Details
    _ \FUN.Print Race Entries
    _ KEY.Horse name Totals
    _ Final Totals
  _ End of report

SEL: IA/IB/D/2 M/C/B/A
F3=Exit  F6=Cancel pending moves  F14=Map

```

Press Enter.

## The Modified Device Structure

The modified device structure is shown below. The backslash character indicates that the Print Race Entries function is attached to the preceding Details format and will be printed after each detail record.

```

EDIT DEVICE STRUCTURE                               My model
Function name : Print Horses

___ Page Headings
___ First
___ KEY.
___ De
___ \F
___ KEY.
___ Final
___ End of r

Exit Device Structure
Select one of the following:
1. Exit without update
2. Exit and save device structure
3. Return to editing

Option: 2
F12=Cancel

SEL: IA/IB/D/2 M/C/B/A
F3=Exit F6=Cancel pending moves F14=Map

```

Press F3 to exit.

## Saving the Device Structure

Save the modified device structure by selecting option **2** from the Exit Device Structure panel.

```

EDIT FUNCTIONS                               My model
File name. . . : Horse                               ** 1ST LEVEL **

? Function      Function type      Access path
- Change Horse  Change object      Update index
- Create Horse  Create object      Update index
- Delete Horse  Delete object      Update index
- Edit Horse    Edit file          Retrieval index
S Print Horses  Print file          Horse name order
- Select Horse  Select record      Retrieval index
- Select Mares  Select record      Mares
- Select Stallions Select record    Stallions

SEL: 2-Details, P-Parms, F-Action diagram, S-Device design, N-Harr, O-Open,
T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock.
F3=Exit F5=Reload F7=File details F9=Add functions F17=Services

```

Press Enter to return to the Edit Functions panel.

## Displaying the Report Device Design

Examine the effect of the new device structure on the layout of the Print Horses report device design. Type **S** next to the Print Horses function on the Edit Functions panel.

```
00000000000000000000000000000000 Print Horses
```

Horse	M/F	Value	D of B
00000000000000000000000000000000	0	666666666.66CR 00000000	
		Dam:	00000000000000000000000000000000 00000000
		Sire:	00000000000000000000000000000000 00000000
00000000000000000000000000000000	0	666666666.66CR 00000000	
		Dam:	00000000000000000000000000000000 00000000
		Sire:	00000000000000000000000000000000 00000000
00000000000000000000000000000000	0	666666666.66CR 00000000	
		Dam:	00000000000000000000000000000000 00000000
		Sire:	00000000000000000000000000000000 00000000

Race Entries for 000000000000000000000000

Course Date	Time	Entry number	Jockey	Finishing H'c position
-------------	------	--------------	--------	------------------------

Press Enter.

## The New Report Device Design

The modified report device design is shown with the fields resulting from the embedded Print Object function.

## View the Rest of the Report Device Design

You can display the rest of the report device design by pressing Page Down (or Roll Up depending upon your keyboard).

EXIT FUNCTION DEFINITION		My model
Type choices, press Enter.		
Change/create function. . . . .	<u>Y</u>	Y=Yes, N=No
Function name . . . . .	<u>Print Horses</u>	Name
Access path name. . . . .	<u>Horse name order</u>	Name
File name . . . . .	<u>Horse</u>	Name
Function type . . . . .	Print file	
Print function. . . . .	<u>N</u>	Y=Yes, N=No
Return to editing . . . . .	<u>N</u>	Y=Yes, N=No
Submit generation . . . . .	<u>N</u>	Y=Yes, N=No
F5=Refresh   F12=Cancel   F15=Open Functions		

**Note:** If your report device design does not look like that shown above, adjust it until it looks similar. Do not be concerned with exceeding the panel limits. For reports, CA 2E allows 132 characters per line.

## Modifying the Combined Report Device Design

You are restricted in the placement of the fields from the Print Object function. Place the cursor on any PRTOBJ field and press F10. All the PRTOBJ fields are moved in a single block. In other words, you cannot manipulate the PRTOBJ fields individually from this panel. You are only allowed to alter the indentation of the whole PRTOBJ function relative to the left-hand margin of the PRTFIL function in which it is embedded. It is possible to specify individual formats of the PRTOBJ. You can specify an absolute indentation that is fixed relative to the left-hand margin of the report, regardless of any adjustments made to the indentation of the PRTOBJ as a whole.

In the current report design, the PRTOBJ fields are displayed after all the detail line records. This implies that all the horses will be printed followed by all the race entries. The advantage of the report as shown in the above panel is that it enables you to easily distinguish between the different formats. However, to meet the original requirements, you need to change the report functions to print one horse at a time followed by the corresponding race entries. You will do this in the next step.

Press F3 twice to exit the Device Design Editor. Save your design by accepting the default options.



EDIT FUNCTIONS		
File name. . . : Horse		
My model		
** 1ST LEVEL **		
? Function	Function type	Access path
- Change Horse	Change object	Update index
- Create Horse	Create object	Update index
- Delete Horse	Delete object	Update index
- Edit Horse	Edit file	Retrieval index
F Print Horses	Print file	Horse name order
■ Select Horse	Select record	Retrieval index
- Select Mares	Select record	Mares
- Select Stallions	Select record	Stallions
-		
-		
-		
-		
-		
More...		
SEL: Z-Details, P-Parms, F-Action diagram, S-Device design, H-Harr, O-Open, T-Structure, A-Access path, U-Usage, G/J-Generate, D-Delete, C-Copy, L-Lock. F3=Exit F5=Reload F7=File details F9=Add functions F17=Services		
Function 'Print Horses' has been saved.		

Press Enter.

### Specifying Parameters

You have now combined the two report functions and have a suitable report design. However, the function is not yet fully defined. If you generate and compile the program now, it would print all the race entries in the RACE ENTRY file after every horse. You need to pass a restrictor parameter from the Print Horses function to the Print Race Entries function so that only the race entries that apply to each horse are printed after each horse’s details.

There are two places where you can specify the parameters of the Print Race Entries function.

- From the Edit Functions panel of the RACE ENTRY file by typing **P** against Print Race Entries.
- From the action diagram of the Print Horses function.

In this example, you will use the second method. Go to the Action Diagram Editor for the Print Horses function. To do so, type **F** next to the Print Horses function on the Edit Functions panel.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Print Horses

___ > Print      ACTION DIAGRAM EXIT POINTS      F3=Exit  SEL:X,2=Select.
___ .--          USER: Initialize program
___ . ...I       USER: Record selection processing      <--
___ . Posi       USER: Process top of page
___ . ...R       Print first page                      <<< <--
___ . ...P       USER: Null report processing           <--
___ . ...P       USER: Print required level headings     <<< <--
___ . -CA       2 Print details                          <<<
___ . -DB       USER: Print required level totals        <<<
___ . .         Print final totals                      + <<< <--
___ . -#0
___ . ...Process report body                             <--
___ . ...Print final totals                             <--
___ . -ENDCASE
___ . ...Print end of report                             <--
___ . --

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Editing the Action Diagram

Display the user points of the action diagram by pressing F5. To specify the parameters of the Print Race Entries function, locate the point in the action diagram at which the function is called. Since the function was attached to the Detail format, you will zoom into the Print details user point.

Type **Z** against the Print details user point.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Print Horses

___ > Print details
___ .--
___ . Process before print of detail format
___ . ...USER: Process before print of detail format      <--
___ . ...PRTOBJ calls before print of detail format
___ . On print of detail format
___ . > USER: On print of detail format
___ . .--
___ . .--
___ . Print detail line format
___ . Process after print of detail format
___ . ...PRTOBJ calls after print of detail format        <--
___ . ...USER: Process after print of detail format        <--
___ . --

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## The Print Details Construct

You embedded the PRTOBJ function after the detail line format. To reach the equivalent point in the action diagram, type **Z** against the hidden construct, ...PRTOBJ calls after print of detail format.

```

EDIT ACTION DIAGRAM          Edit    MYMDL    Horse
FIND=>                        Print Horses

___ > PRTOBJ calls after print of detail format
Z  : ...Embedded PRTOBJ : Print Race Entries      <--
___

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## Editing the Print Race Entries Function

You have now reached the point in the action diagram where the PRTOBJ is embedded. Type **Z** against the hidden construct, ...Embedded PRTOBJ : Print Race Entries.

```

EDIT ACTION DIAGRAM          Edit    MYMDL    Horse
FIND=>                        Print Horses

___ > Embedded PRTOBJ : Print Race Entries
Z  : ...
   : Print Race Entries - Race Entry *          <<<
___ : ...                                         <<<

F3=Exit  F5=User points  F6=Cancel pending moves  F7=Forward  F8=Backward
F9=Edit parameters  F15=Open Functions  F16=Toggle Change Date  F24=More keys

```

Press Enter.

## The Action Diagram Editor Subfile Selector Values

You can type **?** in the Subfile selector of the Action Diagram Editor to display a list of allowed values. Type **?**.

```

DISPLAY ALLOWED VALUES                                My model
Field name. . . . . : *AD SFLSEL
List name . . . . . : Line Commands

? Value          Description
- *              Activate/Inactivate construct (Comment out)
- **             Place block Activate/Inactivate boundary
- A              Place copied or moved construct after this line
- B              Place copied or moved construct before this line
- C              Copy construct to a point indicated by 'A' or 'B'
- CC             Place block Copy boundary
- D              Delete this construct
- DD             Place block Delete boundary
- F              Edit action or condition details for line
- FF            Edit action parameters
- X              Hide construct
- H              Insert *ADD built-in function
- I+             Insert and Prompt *ADD built-in function
- I*             Insert Comment
- I*F            Insert and Prompt Comment
SEL: X-Select value.
F3=Exit, no selection

```

Press Enter.

## Editing the Function

Since you are in the process of specifying a restrictor parameter, you should select FF, Edit action parameters.

Type **X** as shown to select the FF value.

```

EDIT ACTION DIAGRAM                                Edit      MYMDL      Horse
FIND=>                                              Print Horses

___ > Embedded PRTOBJ : Print Race Entries
? : EDIT ACTION - FUNCTION DETAILS
___ : Function file : Race Entry
   : Function. . . : Print Race Entries
   :
   :
   :
   :
   : F3=Exit F9=Edit parameters F10=Default parms F12=Previous
   :
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

Press Enter.

## Displaying the Function Parameters

The Edit Action - Function Details window is now displayed.

```

EDIT ACTION DIAGRAM          Edit      MYMDL      Horse
FIND=>                        Print      Print
                                Horses

> Embedded PRTOBJ : Print Race Entries

? EDIT ACTION - FUNCTION DETAILS
Function file : Race Entry
Function. . . : Print Race Entries

:
:
:
:
: F3=Exit F9=Edit parameters F10=Default parms F12=Previous
:
:
:
:
F3=Exit F5=User points F6=Cancel pending moves F7=Forward F8=Backward
F9=Edit parameters F15=Open Functions F16=Toggle Change Date F24=More keys

```

To edit the function parameters, press F9. The Edit Function Parameters panel lets you edit parameters for the Print Race Entries function.

## Editing the Function Parameters

To define the Horse name field as a restrictor parameter, type **Z** in the Subfile selector to specify additional details, and type **\*Field, Horse name, and FLD**.

```

EDIT FUNCTION PARAMETERS
Function name. . : Print Race Entries      Type : Print object
Received by file : Race Entry              Acpth: Entries by Horse name
                                           Passed
? File/*FIELD      Access path/Field      as Seq
Z *Field           Horse name             FLD  █
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
-                 -                       -   -
                                           Values
                                           FLD: One parameter per field
                                           RCD: One parameter for all fields
                                           KEY: One parameter for key fields only

SEL: Z-Details (field selection).
F3=Exit F5=Reload

```

Press Enter.

## Specifying a Restrictor Parameter

On the Edit Functions Parameter Details panel you will specify the usage and role of the Horse name parameter. In this case, the parameter defaults to Input (I). You will specify that the parameter has a restrictor role by typing **R** in the Subfile selector. This will cause the Print Race Entries program to process only those records on the RACE ENTRY file for the horse identified by the Horse name parameter.

Type **R** against the Horse name parameter.

```

EDIT FUNCTION PARAMETER DETAILS
Function name. . : Print Race Entries      Type : Print object
Received by file : Race Entry              Acpth: Horse name
Parameter (file) : *FIELD                  Passed as: FLD
? Field                               Usage  Role
  R Horse name                          I

```

SEL: Usage: I-Input, O-Output, B-Both, N-Neither, D-Drop.  
 Role: R-Restrict, M-Map, V-Vary length, P-Position.  
 F3=Exit

Press Enter.

CA 2E confirms that the parameter is a restrictor parameter by showing RST in the Role column.

```

EDIT FUNCTION PARAMETER DETAILS
Function name. . : Print Race Entries      Type : Print object
Received by file : Race Entry              Acpth: Horse name
Parameter (file) : *FIELD                  Passed as: FLD
? Field                               Usage  Role
  R Horse name                          I      RST

```

SEL: Usage: I-Input, O-Output, B-Both, N-Neither, D-Drop.  
 Role: R-Restrict, M-Map, V-Vary length, P-Position.  
 F3=Exit

Press F3 twice to exit and return to the Edit Action - Function Details window.



## Testing the Program

Once the program has compiled, call it from any command line. Various messages relating to the execution of the Query access path (via the i OS Open Query File (OPNQRYF)) command are displayed while the report is produced. A sample report listing data entered with the Edit Horse and Edit Race and Entries programs follows.

Your company name here				Print Horses			
Horse			M/F	Value	D of B		
Bonfire			M	5000.00	2/01/88		
			Dam:				
			Sire:				
Race Entries for Bonfire							
Course	Date	Time	Entry number	Jockey	Finishing position	H'cap	
CAMDEN	5/01/95	1:11:00	2	M Brown	3	0	
GOLDEN	6/12/95	1:30:00	3	M Brown	3	0	
Faithful Dobbin			F	3500.00	5/31/86		
			Dam:				
			Sire:				
Race Entries for Faithful Dobbin							
Course	Date	Time	Entry number	Jockey	Finishing position	H'cap	
CAMDEN	5/01/95	1:05:00	1	A Miles	2	0	
GOLDEN	6/12/95	1:00:00	2	M Brown	1	0	
Pegasus			M	3000.00	6/23/92		
			Dam:	Faithful Dobbin	5/31/86		
			Sire:	Bonfire	2/01/88		
Race Entries for Pegasus							
Course	Date	Time	Entry number	Jockey	Finishing position	H'cap	
CAMDEN	5/01/95	1:00:00	3	M Brown	1	0	
GOLDEN	6/12/95	1:02:00	1	M Brown	2	0	
Final totals							
Total number of horses			:	3			
Total horse value			:	11500.00			
** END OF REPORT **							



# Glossary

---

**access path**

A view of the data in a physical file, in a given key sequence, implemented as an i OS physical or logical file.

**access path format entry**

A process that establishes which fields are present in the access path, which of those fields are key fields for the access path, and the order of those key fields.

**access path relation**

The set or subset of a file's relations that apply to a particular access path.

**access path types**

Types which differentiate the functionality of the various access paths. The types of access paths include Physical, Update, Retrieval, Resequencing, Span, and Query.

**action bar**

A construct that appears at the top of an end user panel. It provides a set of choices which define the actions available to the end user.

**action bar choice**

An option that allows application users access to the actions available for the panel.

**action diagram**

A notation for defining and showing procedural processing logic. It is made up of a number of basic logic constructs: Iterative, Sequential, and Conditional. Most CA 2E functions have an action diagram associated with them.

**action diagram editor**

A CA 2E facility that lets you modify the default processing logic automatically supplied for a function by adding action diagram constructs at specified user points.

**All Objects Model Object List**

A unique list within a CA 2E model that contains an up-to-date detailed model object description for each model object in the model. A model object's description is updated automatically whenever it changes. It is also referred to as \*ALLOBJ.

**animation**

A process that converts a CA 2E device design to a Toolkit panel design that you can use to interactively simulate (prototype) a system design. This is useful for demonstrating a design to end users to ensure that the design meets operational requirements before you begin programming.

**application**

A set of implementation objects, primarily programs and files that together satisfy business needs. For example, a payroll application or an order entry application.

---

**attribute**

A property of an entity. It is any detail that qualifies, identifies, classifies, quantifies, or expresses the state of an entity. Attributes are usually implemented as fields.

**based-on file**

The file to which a given CA 2E object is attached.

**built-in function**

Type of function which provides commonly required low level operations such as addition and multiplication.

**capture file**

One of the two types of CA 2E database files. Capture files generally contain transaction data.

**change type**

The change type indicates the impact a change to a model object has on other model objects. The change type indicates whether impacted model objects require editing and/or regeneration to ensure the integrity of the model and the application.

**component change processing**

An automated impact analysis tool that determines how a change to a model object affects other objects in the model and records whether the affected objects need to be edited or regenerated.

**condition**

A value that specifies the circumstances under which an action or set of actions are to be executed. It also defines the particular value for a field.

**conditional construct**

A construct that allows steps to be conditionally executed within the action diagram logic and generally conforms to nested IF THEN ELSE logic statements or SELECT sets.

**construct**

The system of notation in an action diagram to define a boundary for procedural logic.

**context**

The source of a field within a function. A particular field name can be present several times in a function. The context of the field differentiates between these instances of the field. For example, a field may appear on a panel and also be part of a database file.

**cross reference facility**

The operations CA 2E provides for identifying dependencies between objects. This is also referred to as the model object cross reference facility.

**current version**

This refers only to versions of functions or messages. A version is current when it is the version that is active in the model. In other words, the current version is the one used by and referred to by other model objects and is the one shown on CA 2E panels that are not specific to processing versions. Only one version in a group of versions can be current at a time.

**database function**

An internal function which reads or updates the database. The database functions are: Change Object (CHGOBJ), Retrieve Object (RTVOBJ), Create Object (CRTOBJ), and Delete Object (DLTOBJ).

**data model**

A representation of the information used by a business. A data model contains groups of data called entities, attributes of the entities, and the relationships between those entities or between an entity and an attribute.

**design model**

The library that holds the i OS database files that compose the model. The model library has an associated library, the generation library, which contains the source code CA 2E generates for the model.

**design object**

See model object.

**designer**

The user type that can change any aspect of the design model including data relationships and functional specifications. In general when a designer is using a model, no other user can access that model. A designer also has the authority to lock and unlock objects. Also known as \*DSNR.

**developer**

A term used to designate either a designer (\*DSNR) or a programmer (\*PGMR) when a distinction between the two user types is not necessary. Designers and programmers are often referred to collectively as “developers.”

**device design**

The layout of a panel or report associated with a function.

**device function**

A function which has a device design associated with it, such as Edit Record (EDTRCD), Display File (DSPFIL), Edit Transaction (EDTTRN), and Print File (PRTFIL).

**domain**

A set of possible values an attribute can take, along with its definition of length and data type, such as numeric or alpha.

**dynamic selection**

An option that specifies that all records are included in the access path regardless of any select/omit criteria. The records are then filtered by the system as they are read by a program.

**end user**

The person who uses the generated application once it is in production.

**entity**

An object which is significant or relevant to the business it represents; for example, a company, a person, a product. An entity is a potential file to be created in the data model of your application system.

---

**entity-relationship diagram (ERD)**

A graphic chart used to record the entities and entity relationships using boxes, lines, and notation.

**expansion**

The process of resolving all relations for a file or for all files in a model. For example, synchronizing a model causes the expansion of all relations to rebuild the file entries for all files in the model.

**external function**

A function which is implemented as a separate HLL program rather than as a subroutine. All functions are either external or internal.

**field (CA 2E)**

A description of an item of data. It must have a name and a field type. Field names may be up to 25 characters long. The names of fields must be unique within the design model.

**field type**

An assigned type which defines the type of data that the field represents; for example, a number or a date.

**file (CA 2E)**

A list of relations that describe an entity. CA 2E can automatically resolve the relations into a list of file entries or fields. File names may be up to 25 characters long. The names of CA 2E files must be unique within the model.

**file entry**

A field that arises as a result of resolving CA 2E relations. An entry indicates the presence of a field on a file. The three types, depending on the type of relation from which the field is resolved, are: key field entry, attribute entry, and virtual field entry.

**file type**

The description of how the file will be used. Each file must be one of the allowed file types: structure (STR), reference (REF), or capture (CPT).

**first normal form (1NF)**

The normalization rule that eliminates repeating groups of data from an entity.

**foreign key**

A file entry that is the result of a relationship between two entities. The relationship is defined by a non-key attribute of one entity that exists as the primary key of the other entity.

**function**

A template process that operates on a file or field. A function serves as a building block with which to design applications.

**function field**

A field whose value is not physically stored in the database but is derived from other fields or files. Function fields can be placed on device designs and used in action diagrams.

**function options**

The optional features of the standard functions.

**function parameter**

A field whose value is passed into or out of a function when it is executed. Each parameter can have a parameter role and a parameter usage.

**function parameter role**

The parameter role that specifies how a parameter is to be used in a function into which it is passed. Standard parameter roles include Restrictor, Positioner, Map, and Vary.

**function parameter usage**

The parameter attribute that specifies whether a function parameter is to be passed to and/or returned from a CA 2E function. Function parameter usage includes Input, Output, Both, and Neither.

**function redirection**

A process whereby the using objects of a given function (FUN) or message (MSG) are redirected to use an alternative FUN or MSG. This occurs when you make a version of a function or message current.

**function type**

One of a group of process templates CA 2E provides such as EDTFIL.

**generalization**

The process during data modeling of combining two entities of the same class into one entity and renaming it.

**generatable objects;defined generation**

The creation of application source code from a CA 2E function definition and device design.

**generation library**

The library that holds CA 2E generated source code, compiled objects, and help text for a model.

**group**

This term relates to versions of functions or messages. Versions of the same model object form a group of versions.

**impact analysis**

The process of determining the impact of a proposed or actual change to model objects in order to ensure the integrity of a set of changes by identifying and including dependent objects. CA 2E impact analysis tools include, cross reference commands, interactive panels to determine usages and references, simulation, and automated component change processing.

**implementation**

The process of setting up your application and moving it into production once you have generated and compiled source objects.

---

**implementation objects;defined implementation object**

Objects such as generated source and compiled objects as opposed to model objects. They are also known as traditional or 3GL (third generation language) objects.

**internal function**

A function which is generated as in-line code within the calling function.

**involution**

A process that occurs when an entity refers to itself. This is known as an involuted relation. Also called a self referencing relation.

**iterative construct**

A programming construct that represents repetitive statements or loops that will be implemented as REPEAT WHILE logic within the main action diagram logic.

**key**

The field(s) used to uniquely identify a record in a file.

**list entry**

This is another term for model object list entry.

**logical file (i OS)**

A logical view of the data in the physical file based on key sequence. Access paths can be implemented as logical files.

**message function**

Function types which allow you to send error, status, or information messages to the invoking program or to send information to other destinations such as submitting a job to batch.

**model object**

CA 2E design objects such as, access paths, functions, and fields, as opposed to implementation objects such as, generated source or a compiled object.

**model object description**

A reference to a model object in the All Objects list (\*ALLOBJ). It contains detailed information about the actual model object. A model object's description is automatically updated whenever the model object changes, and as a result, always reflects the current state of the object.

**model object list entry**

A reference to a model object in a named model object list. It reflects the state of the model object at the time the list entry was created or refreshed, and as a result, provides an historical record of the model object.

**model object list**

A list of references to model objects.

**model profile**

A user profile associated with a model where you can define defaults for various processes and file specifications for an interactive session.

**model value**

A specific value for a model that controls particular features of the interactive use of CA 2E, code generation, or implementation.

**normalization**

The process of eliminating data redundancy using specific standards and rules in data modeling.

**notepad**

A temporary action diagram which can be used to copy constructs between action diagrams.

**object (CA 2E)**

A design element of a CA 2E model; for instance, a field or function. See also model object.

**object type**

Each CA 2E object must belong to one of the CA 2E object types: file, field, condition, message, function, access path, or application area.

**open functions**

All functions a single user currently has open for editing.

**i OS**

The IBM i operating system.

**owning model object**

Certain object types exist associated with an owning type. For example, functions (FUN) and access paths (ACP) are owned by a file (FIL) object type.

**panel**

A panel is displayed by a device function. Panels are composed of a header, a footer, and a body.

**panel entry**

The appearance of a field on a panel device design. Panel entries arise either from the resolution of access path relations, from function parameters, or from explicit specification by the application user.

**parameter**

See function parameter.

**primary key**

A chosen attribute or group of attributes assigned to a particular entity to uniquely define it within an entity relationship model.

**program**

The result of generating and compiling a CA 2E function.

**programmer**

A user type who can create, change, and delete any help text, access paths, functions, and arrays, including working with action diagrams and field conditions for database and function fields. A programmer cannot alter the database files or fields. In general, programmers cannot use a model while a designer is using it. Also known as \*PGMR.

---

**prototyping**

The presentation of a realistic mock up of a system used to verify that the design meets the operational requirements.

**redirection**

See function redirection.

**reference file**

One of the two types of database files: Reference files contain static data, in contrast to capture files which contain volatile transactional data.

**references**

The references for a model object are all objects the model object refers to, or contains. For example, an internal function is a reference of the external function that contains it. In other words, references are the model objects the referring model object requires in order to be complete or to exist.

**referential integrity checking**

The process that ensures that proper relations between files are maintained. This usually means making certain foreign keys contain valid values in the referenced files. CA 2E functions generate source code by default to provide referential integrity checking.

**relation**

A connection between one entity and another or between an entity and an attribute. When specified in CA 2E, a relation defines the connection between files or between a file and a field in the file.

**relation type**

A relation must be one of a fixed number of types; the type is indicated by the verb used to represent the relation such as Has or Known by.

**restrictor parameter**

One of the allowed function parameter roles. If a function has a restrictor parameter, it can only process database records whose keys match the restrictor. Any device design is modified automatically to meet restrictions. Restrictor parameters must be key field entries on the access path to which the function is attached.

**second normal form (2NF)**

The normalization rule that eliminates attributes that are not dependent on the primary key. In 2NF, non-key attributes are fully functionally dependent on the primary key.

**select/omit set**

A choice that defines criteria for the inclusion or exclusion of records from the physical file.

**sequence construct**

The simplest action diagram construct which specifies a list of actions or other constructs that are to be executed sequentially.

**session list**

A model object list to which all objects you change during a session are logged.



**simulation of a change**

A process that lets you see the impact of a proposed change to a model object on other objects in your model before you actually make the change.

**source file (i OS)**

A file specially formatted to contain program source code. It is usually multi-member with each member containing a different set of source.

**standard function**

A function which specifies entire programs (external functions) or subroutines (internal functions). There are three main classes of standard functions: device functions, database functions, and user functions.

**static selection**

An option that specifies that only those records that satisfy select/omit criteria are included in the access path.

**subfile**

The part of a panel that contains a repeating detail format such as that needed for a list longer than one page.

**subfile selector**

A field on the left hand side of a subfile used for specifying an action to be taken against a particular subfile record or records. For device designs there will normally be a subfile selection value explanation text field to explain the allowed values for the subfile selector.

**synchronize a model**

The expansion of relations to rebuild the file entries for all files in the model.

**third normal form (3NF)**

The normalization rule that eliminates non-key dependencies between attributes of an entity. In 3NF, non-key attributes are mutually independent.

**unique key**

A file key that ensures each record in the file is unique.

**usages**

The usages for a model object are all objects that use the model object. They are also called using objects.

**user**

A user type that is limited to viewing the model and cannot change it. Also known as \*USER.

**user function**

A function that is entirely user specified.

**user point**

A point in an action diagram at which you can insert logic. A user point is indicated by an arrow on the far right in the action diagram editor.

---

**version**

A model object that originated as a copy of either a function (FUN) or a message (MSG). Versions of the same originating model object are linked together to form a group of objects.

**virtual field**

A field that is logically, but not physically, present on an access path. Virtual fields are the result of specifying that fields in a related file are to be included on an access path.

**virtual field entry**

An entry which describes the presence of a field on a file or access path as a result of a join with another file. Virtual fields can only be specified for the Owned by, Refers to, and Extended by relations.

**window**

A panel on which information appears that is not necessarily a full panel. Windows can be varying sizes and can be positioned in varying locations.

# Index

---

## A

---

- Access Path Details, 3-57, 3-61
- Access Paths, 3-49
- Access Paths and Functions, 5-1
- Accessing Session List, 6-42
- Action
  - Diagrams, 4-40
  - Text, 4-103
- Action Bar Navigation, 6-12
- Adding
  - Details of Extended Relations, 3-35
  - Field Condition Detail, 3-25
  - Field Conditions, 3-24
  - Function Fields, 8-9
  - Functions, 4-6
  - More Relations, 3-32
  - New Message Function, 4-60
  - Print Race Entries, 8-22
  - Text to the First Message, 4-63
  - Validation Procedure, 4-47
  - Virtual Fields, 3-39, 7-7
- Animate Functions Panel, 6-4
- Application
  - Entities, 3-2
  - Maintenance, 6-1
- Attributes for Each Entity, 3-3

## B

---

- Batch Generation, 7-11

## C

---

- Change Type, 6-75
- Changing

- Referenced Path, 3-65
- Window Dimensions, 4-38

- Completing Requests, 5-6
- Condition Values, 5-11
- Confirm Prompts, 4-96
- Creating
  - New Functions, 4-7
  - Span Access Path, 7-3

## D

---

- Data Model, 3-1
- Declaring More Files, 3-19
- Default
  - Access Paths, 3-50
  - Device Design Formats, 4-10
  - External Functions, 4-3
  - Functions, 4-3
  - Report Layout, 8-7
- Defining
  - Objects, 3-14
  - Parameter Details, 4-93
  - Print Function, 8-2
  - Requirements, 2-1
  - Second Action, 4-69
- Deleting a Model List Entry, 6-34
- Development Life Cycle, 1-4
- Device Designs, 4-9
- Display Functions for Selection, 5-4
- Displaying
  - Function Details, 4-8
  - Selected Relations, 3-29
- Documenting Relations, 3-21

## E

---

- Edit

---

- Action Function Name Panel, 4-66
- Database Relations, 3-11
- Screen Format Details, 4-17
- Transaction, 7-1

#### Editing

- Action Diagram, 4-48
- Default Device Design, 4-13
- Model Objects, 6-31
- New Function, 6-47
- Panel Format, 4-17
- Session List, 6-22

Editing Message Text, 6-50

Embedding Print Race Entries, 8-30

#### Entering

- New Field Name, 3-44
- Relation Statements, 3-12

Entity Unique Identifier, 3-4

Exiting the Action Diagram, 4-56

Extending Relations, 3-30, 3-33

## F

---

Field Details and Conditions, 3-21

File Attributes, 3-20

#### Function

- Fields, 7-17
- Options, 4-76
- Parameters, 4-90
- Versioning, 6-40

Functions Introduction, 4-1, 6-10, 6-25

## G

---

Generating and Creating Objects, 5-7

## H

---

Hidden Fields in Subfile Record Format, 4-35

## I

---

Impact Analysis, 6-74

#### Inserting

- Condition, 4-51
- Single Action, 4-51

Interactive Device Design, 6-2

Introduction, 1-1

Invoking the Device Design, 4-34

## J

---

Job List Commands, 6-55

## L

---

Linking Functions, 4-79, 4-80

## M

---

#### Message

- Function Details, 4-62
- Function Text, 4-73
- Functions, 4-58

#### Model

- List of References, 6-73
- Object
  - Cross References, 6-65
- Lists, 6-20
- Usages Panel, 6-67

Modifying an Action, 4-102

## N

---

#### Naming

- Function, 4-85
- Select/Omit Set, 3-62
- Versions, 6-46

New Case Construct, 4-82

New Function, 4-94

---

Normalize Entities, 3-7

## O

---

Open Functions Panel, 6-7

## P

---

Path Access Formats, 7-4

Positioning Model Object List, 6-25

Print

- File Function, 8-1

- Object Function, 8-21

## R

---

Reassigning Command Keys, 6-10

Reduce Width of Device Design Layout, 4-14

Relations Panel, 4-105

Reload Subfile, 4-98

Removing Field Labels, 4-22

Renaming Fields, 3-43

Resynchronizing Design Model, 5-14

Retrieval Access Path, 3-53

## S

---

Saving Modified Panels, 4-104

Sequence of Relations, 3-32

Shortening Field Labels, 4-19

Specifying

- Check Condition, 3-27

- Function Parameters, 4-91

- Object Attributes, 3-15

- Virtual Fields, 3-38

Static and Dynamic Selections, 3-58

Submitting Function for Generation, 7-39

Substitution Variable, 6-51

## T

---

Testing Compiled Program, 5-15

## U

---

Understanding Function Details, 4-7

Using

- CA 2E, 1-5

- Function Keys, 4-26

## V

---

Validation of Relation Statements, 3-13

Viewing a Subset, 6-27

Virtual Field Entries, 3-40

Virtual Fields, 3-37

## W

---

Windows Device Design, 4-33