

# **STARWARP<sup>®</sup>**

## **User Guide**

**Version 6.1.0**

**Order Number: STWUG610**

---

The logo for Serena Software, featuring the word "serena" in a bold, lowercase, sans-serif font. The letter "e" is stylized with a curved underline that loops back to touch the top of the letter.

## STARWARP User Guide

### STARWARP<sup>®</sup> 6.1.0 User Guide

First Edition – January 1, 2000

© SERENA Software, Inc., 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000. All rights reserved. This manual, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. The content of this manual is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by SERENA Software, Inc. SERENA Software, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this book. Except as permitted by such license, no part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of SERENA Software, Inc.

### STARWARP<sup>®</sup> Publications

Additional copies of the following STARWARP publications may be purchased from SERENA:

*STARWARP 6.1.0 User Guide* is a guide to STARWARP concepts and facilities.

- The User Guide section explains how StarWarp can help with Year 2000 conversions and is a guide to StarWarp functions and features.
- The Messages section explains both online and batch messages for StarWarp. This section also describes how to deal with error situations.
- The StarBat section contains extensive information on batch facilities. These facilities are designed to execute as background programs and to process control statements. Programmers debugging or designing batch facilities background jobs use the StarBat section.

*StarWarp 6.1.0 Reference* is designed for professional applications and systems programmers who have experience with programming, file structures, utilities, and testing practices within the mainframe environment.

- The Reference section is an alphabetic guide to the functions and commands of StarWarp.
- The Installation section explains procedures for setting up StarWarp, including installation tailoring, execution of the program, and installation verification.
- The StarWarp Appendixes section provides a history of the various upgrades of StarWarp.

Change Man, Comparex, StarWarp, and StarTool are registered trademarks of SERENA Software, Inc. CDF (Concurrent Development Facility), Detect+Resolve, FULL.CYCLE, Merge+Reconcile, SERNET, SER<sup>POWER</sup>, and X:Change are trademarks of SERENA Software, Inc. Change Transfer is a trademark of SERENA Software, Inc. and High Power Software, Inc. Detect+Resolve Mainframe is jointly owned and developed by SERENA Software, Inc. and High Power Software, Inc. SyncTrac is a registered trademark of SERENA Software, Inc. and High Power Software, Inc.

CA-ACF2, CA-Librarian, CA-Panvalet, CA-Top Secret, and CA-Endevor are registered trademarks of Computer Associates International, Inc. DB2, DFSMS, DR/RACF, ESA/390, IMS, ISPF, MVS/DFP, MVS/ESA, MVS/XA, RACF, and VTAM are trademarks and registered trademarks of the International Business Machines Corporation (IBM). Microsoft Windows NT and Microsoft SNA Server are registered trademarks of Microsoft Corporation. All other products or company names are used for identification purposes only, and may be trademarks of their respective owners.

## Special Notices

### About SERENA Software

SERENA Software specializes in providing integrated products and services to support enterprise software change management. SERENA focuses on providing a total solution to the problems of locating, fixing, and managing changes to applications and systems software. SERENA Software's products have become industry-recognized, and are currently used in more than 2,000 data centers worldwide.

SERENA is headquartered in Burlingame, California, and has offices in the United Kingdom, Germany, and Canada.

### FULL.CYCLE™ Mainframe

#### Change Man®

*For Software Life Cycle Management*

A comprehensive software change management system that automates the entire software change life cycle.

#### Change Transfer™

*For Efficient VSAM Backups*

Through its innovative fingerprinting process, identifies and backs up only those VSAM records that have been changed, reducing the need for daily backups.

#### Comparex®

*For Intelligent Comparison*

The industry-leading tool for efficient application testing and software quality assurance. It is used for regression testing, auditing, reconciliation of entire production libraries or of single members, and for data extraction for testing.

#### Detect+Resolve™ Mainframe (SyncTrac®)

*For Intelligent Change Detection and Synchronization*

A synchronization and data set change tracking product for the MVS software environment. Primary applications include centralized migration of systems maintenance, systems audit and change control, disaster recovery, and data center consolidation or migration.

#### Merge+Reconcile™ Mainframe (CDF™)

*For Intelligent Version Merging and Reconciliation*

An MVS product that consolidates multiple versions of application programs, primarily for collaborative development. It quickly compares and intelligently merges ongoing internal development and third party software code into one consolidated file.

#### SERNET™ Connect (X:Change™)

*For Enterprise Connectivity*

A state-of-the-art tool that provides cross-platform file management support for programmers involved in desktop-based software development for mainframe and client/server applications. SERNET Connect comprises powerful features that enable users to synchronise, transfer, and distribute data and software quickly and easily.

#### SERNET™ Mainframe

*The Core Enabling Infrastructure for FULL.CYCLE Mainframe*

## STARWARP User Guide

### StarTool®

*For File and Data Management*

An integrated programmer's environment comprising a workbench of productivity and diagnostic tools for managing sequential, partitioned, extended partitioned data sets, VSAM data sets, load modules, and other files.

### StarWarp®

*For Data Conversion*

A programmer's workbench for managing software conversions, manipulating dates, and managing data in most file types. StarWarp is a critical tool for use in Year 2000 and European Monetary Union (EMU) projects.

## FULL.CYCLE™ Desktop

### Detect+Resolve™

*For Intelligent Software  
Configuration Problem Detection  
and Resolution*

A product that provides sophisticated and automated problem detection and resolution of software configuration problems for the enterprise desktop.

## Getting Support

### Support for the United States and Canada

<b>Support Hours</b>	Monday through Friday 5:00 A.M. to 5:00 P.M. Pacific time
<b>Email</b>	support@serena.com
<b>URL</b>	<a href="http://www.serena.com/html/support.htm">http://www.serena.com/html/support.htm</a>
<b>Phone</b>	650 696-1850
<b>Facsimile</b>	650 696-1817
<b>Mailing Address</b>	SERENA Software, Inc. 500 Airport Boulevard, Second Floor Burlingame, CA 94010-1904

### Support for the United Kingdom, Germany, Switzerland, and Austria

<b>Support Hours</b>	Monday through Friday 8:30 A.M. to 7:00 P.M. Central European time
<b>Email</b>	eurotech@serena.com
<b>URL</b>	<a href="http://www.serena.com/html/support.htm">http://www.serena.com/html/support.htm</a>
<b>Phone</b>	44 1923 222233
<b>Facsimile</b>	44 1923 224433
<b>Mailing Address</b>	SERENA Software International, (UK) Ltd. 33 Metro Centre, Dwight Road Watford, Hertfordshire WD1 8SB, England

### Support for All Other Countries

Please contact your local distributor.

### **Getting Support on the Internet**

SERENA Software provides extensive support on the Internet through out “Customers Only” section on the Web and through our FTP server. To obtain an ID and password for access to support on the Internet, send a request by email to [support@serena.com](mailto:support@serena.com).

Once you have an ID and password, you can

- Report new issues.
- Search our problem tracking system for information about existing problems.
- View a knowledge base of frequently asked questions and helpful product hints.
- Query the call tracking database to obtain the current status of an open issue.
- Access our FTP server to download product fixes and online documentation in PDF format.

Also, as a SERENA customer, you can get the latest product information via email by subscribing to one of our mailing list servers (LISTSERV). For information on how to subscribe to the StarTool mailing list server, send an email message to [support@serena.com](mailto:support@serena.com).



# Contents

**Preface ..... i**

- Software Environment.....i
- Version, Release, Modification .....i
- What is STARWARP? .....ii
- What Types of Data Sets can be Processed? .....ii
- STARWARP Initialization .....ii

**Introduction..... 1**

- Data Aging With STARWARP .....1
- STARWARP DATA AGING CONTROLS .....3
- General Overview .....5
- Environment .....6
- ISPMODE Execution.....6
  - Standard Users .....6
- Batch Mode.....10
- Invoking STARWARP.....10
- Single Subcommand Mode .....10
- StarWarp Subset of StarTool.....11
- Restricted Subcommands .....11
- Using Subcommands .....12

**The Menu System ..... 13**

**Guide to Data Aging ..... 17**

- Objective ..... 17
- Basics ..... 17
- Scrolling..... 18
- Option 4 -- Warp.....20
- Option 4A -- WARPCOPY: Create a Job from a Copybook.....21
- Option 4B -- WARPSAVE: Create a Job from a Saved Table .....29
- Option 4C -- PROFMAN: Manipulate STARWARP Saved Tables.....30
- Option 4D -- DATECALC: Calculate Dates.....31
- Option 4E -- EUROCALC: Monetary Conversion Calculator .....32
- Option 4F -- DATEDIFF: Date Difference Calculator .....35
- Option 4G -- HOLIDAYC: Create a Holiday Calendar .....36
- Option 4H -- IMPACT: Load to Source Reconciliation .....38
- Option 4I -- COMPCHK: Load Library Summary .....44
- Option 4J -- CSECTCHK: Load Library Component Check.....46
- Option 4K -- LANGTYPE: Classify Source Modules.....48

**Advanced Topics ..... 53**

- Warp Commands .....53
- Expanding a Date Field .....54
- Warping Multiple Date Fields.....56
- Warping Occur Depending On Fields .....58
- Saving a STARWARP Table.....61
- WARP Table Views .....63
- Modeling the Input Data Set .....66
- The STARBAT Program.....68

Optional DD Statements.....	68
Required DD Statements .....	68
Control Statements.....	69
Control Statement Syntax .....	69
Frequently Used Functions .....	69
Frequently Used Parameters .....	70
Examples.....	70
<b>Change Management and STARWARP.....</b>	<b>71</b>
MEMLIST Option for Change Man.....	72
MAP LCT -- Produce Linkage Control Table Statements .....	73
Load Module HISTORY Preload .....	75
<b>Batch Processing .....</b>	<b>77</b>
<b>ISPMODE in Batch.....</b>	<b>78</b>
F Command .....	79
EXCLUDE Facility .....	80
SORT Command.....	80
L Command.....	80
X Command .....	80
OUTPUT Command.....	80
<b>Quick Start to Using STARWARP.....</b>	<b>81</b>
Allocating a Data Set.....	82
Building a Member List.....	82
Checkpointing the Log.....	83
Comparing Members.....	83
Editing Multiple Members .....	83
Excluding Multiple Members .....	84
Formatting Member Data .....	84
Listing Module History .....	84
Locating Data in Dialog Tables .....	85
Managing Data Set Lists .....	85
Managing Member Lists.....	85
Modifying Member Statistics .....	86
Obtaining Assistance.....	86
Processing Multiple Data Sets .....	88
Saving a LISTC table .....	88
Saving a MEMLIST Table .....	88
Searching a Data Set .....	88
Searching Multiple Data Sets .....	89
Selecting Members.....	89
Specifying a Member Group.....	90
Trimming Dialog Tables .....	90
Updating a Data Set .....	91
Updating Multiple Data Sets.....	91
Validating Data .....	92
<b>Edit Macros .....</b>	<b>93</b>
ATTRIB Edit Macro .....	93
CUT and PASTE Edit Macros .....	93
EQUAL Edit Macro and REEQUAL CLIST.....	95
PACKOFF Edit Macro .....	96
PACKON Edit Macro .....	96
PDSEDSN Edit Macro .....	96

TRAP Edit Macro.....96  
VOLUME Edit Macro .....97  
**Appendix A. IBM Batch Utilities Comparison..... 99**  
**Appendix B. ISPF Utilities Comparison ..... 101**  
**GLOSSARY ..... 103**  
**Index ..... 105**



## Preface

---

This manual contains information on the use of STARWARP, with emphasis on those features most useful to application programmers and operation analysts responsible for the development and maintenance of MVS production systems especially in year 2000 conversion projects.

STARWARP is a programmer's workbench that provides tools for analyzing the scope of year 2000 projects and expediting the conversion process. STARWARP enables programmers to create test data for year 2000 conversion projects by automating the process of increasing, decreasing, or specifying default values for date fields.

With STARWARP, users can simulate future dates without the effort of writing specialized batch programs for each test scenario. STARWARP can also address business rules by assuring that certain activities will occur on specified days, such as the last day of a month or the first Monday of a quarter.

STARWARP addresses the issue of data aging by providing a method of warping, shifting, or aging the dates in existing files, thus providing the ability to perform necessary testing without the need to write one-time specialized programs for each test scenario. For examples of how to warp dates, see **Guide to Data Aging** on page 17.

### Software Environment

STARWARP runs under the two major IBM MVS operating systems:

- MVS/ESA (*any release*)
- OS/390 (*any release*)

In addition, the following environment should be available:

- ISPF and ISPF/PDF (*Version 4.1 or above*)
- TSO/E (*any release or any version*)

### Version, Release, Modification

All software created and maintained by **SERENA Software** will have a Version, Release, and Modification level associated with it. Only when the Version or Release number changes (*usually annually*) will the full customer base (*those who are up to date with maintenance*) be issued new tapes and documentation; other releases can be requested if desired.

This manual describes:

#### STARWARP

**Version** ..... 6  
**Release** ..... 1  
**Modification** ..... 0  
**Julian release date** ... 2000.001

### ***What is STARWARP?***

STARWARP is a multi-purpose utility program for data processing professionals, particularly application programmers working on year 2000 tasks. Programmers using STARWARP have a new, more efficient environment in which to perform their everyday tasks; but they also have their previous environment available as well.

STARWARP is a subset of the STARTOOL program and generally includes all facilities required to investigate existing data sets and program libraries for year 2000 projects. STARTOOL subcommands such as FIXPDS or COPY which modify data sets and their members are not available in STARWARP.

STARWARP is a program designed to run as a normal ISPF dialog manager. It also operates as a normal TSO command processor and in the background under the terminal monitor program.

ISPF dialog facilities are also available in batch executions of STARWARP if an ISPF environment is available. Even though line commands are not supported (except with the global APPLY command) and STARWARP can not prompt for input data, all of the major ISPMODE functions can be established. A batch STARWARP application can save MEMLIST, LISTC/LISTF or WORKPAD tables for later interactive use or process multiple data sets in a table with global commands.

STARWARP also creates batch jobs which utilize STARBAT to process data sets in many different ways. For more information, please see the **STARBAT for STARWARP Guide**.

### ***What Types of Data Sets can be Processed?***

STARWARP can process disk files with partitioned, partitioned extended, sequential, VSAM or direct organization. Any record format is acceptable; STARWARP assumes that any partitioned data set with record format U is a load library and other libraries are usually called "source" libraries.

### ***STARWARP Initialization***

When STARWARP is invoked, license information is normally displayed as shown below to identify the program, release number and release date.

```
PDS100I PDSE/StarWarp -- Version 6.1.0 2000.000

Proprietary software product of SERENA Software Intl.
Phone (650)696-1800 OR FAX (650)696-1776
LICENSED TO: your corporate name/trial offer expires ...
             your city, state, zip/agent to contact for license ...
All other rights reserved - use of this software
product by unauthorized persons is prohibited.
```

**Figure 1. Sample License Information**

---

---

## Introduction

---

---

### ***Data Aging With STARWARP***

Data aging (or "data warping") is a process that really should be a normal part of application development. The impending Year 2000 situation has brought the concept of data aging into public focus, but it is actually a valuable procedure that should be properly understood. In many environments, data aging should become part of the normal testing process. But nowadays, before and after the turn of the century, it is especially important to understand the way in which data aging helps you test how your applications will work in the future.

Almost every business application relies on data files, such as customer information files and transaction information files. These files almost always contain dates: for example, the date of the transaction, the date that billing is due, the projected and actual dates of delivery, and so forth. Computer application programs create and update these files, often as part of a long job stream which processes an entire application's data. CICS and database programs update the data files as well, record by record, supplying additional date information in the process. As the Year 2000 approaches and passes by, how can we know in advance, that the computer date processing in our application programs will continue to work properly? After the year 2000 has passed by, how can we do regression testing to simulate earlier data, and fix programs which haven't been fixed yet? Data aging can be done in reverse, too.

One way of testing our applications, is to pretend that they are executing at a later date, some time in the future. If we have a separate computer (or LPAR) for date testing, we can IPL (boot up) that computer using a future date, say December 29, 1999, or April 15, 2000. That's all well and good, but by itself, it won't do the testing job. We need to simulate business data that would have been acquired at an appropriate time, shortly before the date at which the "future" system is supposed to be running.

For example, if the testing system date is December 29, 1999, we'd like to look at data that was acquired right before that date--for example, in November and December of 1999. Our business processing might create some dates that are after that time, say, into the years 2000 and 2001, and we want to know if this processing is working properly. However, we haven't done the business yet, if it's now some time considerably before then. So how do we get the appropriate data to test with?

We can get the future or past data, using data we have now.

We only need an accurate way to add (or subtract) the proper number of days, months, or years, to every date in our data files, under careful control, so that today's data appears as though it were created "tomorrow", or "yesterday". Holidays and weekend days need to be adjusted too, so the business doesn't appear as though it were done on those days. This is the concept of "aging", or "warping" your data, and this is what STARWARP can do.

## STARWARP User Guide

STARWARP has the capability of accurately "aging dates", or adding (subtracting) a fixed number of days, months, or years, to existing data in data files. It could be, that the "aged date", which was obtained after adding an increment to a date in the current file, comes out on a weekend or holiday when business is not normally transacted. STARWARP can make corrective date adjustments with many options and choices. STARWARP provides the programmer with the specific ability to go to the previous, or next business day after any date has been incremented, and it takes the holidays into account also.

How can STARWARP find the date fields within a record in a data file? STARWARP actions can be controlled in two ways, interactively, and in batch. Interactively, using STARWARP ISPF panels, a special table called a "copybook table" can be created. This copybook table is a data description which is made from a COBOL or PL/I data description or "copybook". The copybook table tells STARWARP the proper displacement and data format of all the fields in the file. You go into the table, and pick which fields you want to manipulate. One or more conditional actions can be associated with each data field in the copybook table, so you can change (and "warp") each data field according to rules, independently of any other data field.

STARWARP can then use the copybook table to BUILD a STARBAT batch job which is run, to actually change the data in the file itself, or to copy some or all of the records to a new file. STARBAT control statements, in the generated STARBAT batch job, can themselves be changed by the programmer directly, to control the date processing. You have the option to skip the interactive part, and write a STARBAT job directly, if you want. So STARWARP controls are twofold, either through ISPF copybook tables, which are created interactively, or by STARBAT batch control statements that can be altered by editing the JCL.

STARWARP copybook tables can be saved, recalled, and altered as often as you desire. It just takes a second more, to BUILD a new STARBAT batch job to run.

You, the programmer, construct the copybook table, and you, yourself, save and submit the STARBAT JCL. So you yourself, have complete control to tell STARWARP which date fields to age, and how to age them. Once you construct a copybook table for STARWARP to use, and build and run a STARBAT batch job, STARWARP will accurately age all of the dates, so that the data file will appear as though it were created in a different time period.

Now you have test files in your possession which appear to have been created just before a different IPL date, so you can run your normal production jobstream and test it. You'll see if the different dates run properly in your production jobstream, or if they give rise to errors. You can now run your "year 2000" tests much more confidently, using data which looks exactly like production data.

## STARWARP DATA AGING CONTROLS

To better grasp the power of STARWARP, we'll talk about some of the actions you can do to dates, and we'll begin to show you the kind of practical control you can have.

As we said before, records in a file are defined to STARWARP by importing COBOL or PL/I data descriptions into a copybook table under ISPF. Each field in a copybook table can be dealt with separately--STRW provides the capability of issuing a line command against each data field.

For example, if we have a data field called **02 TRANSACTION-DATE PIC X(8)** in CCYYMMDD format, we can age this field by "20 business days", not counting weekend days and holidays. The designation would be, **20B** as opposed to **20D** (which would refer to "20 days, regardless"). Then, we need to tell STARWARP about our holidays by specifying the Holiday Exit Name. If we don't enter a Holiday Exit Name, STARWARP just skips the weekend days, without any extra holidays counted.

The actions against one field, can depend conditionally on the state of another field. For example, suppose there is one branch office whose system date was accidentally IPL'ed to a date that's a week early. No one noticed the bad date for a while, and all the records from that branch office (for several days) have corrupted a combined master file composed of records from all the branch offices. To fix the combined file, you need to execute logic in STARWARP that only operates against records from the offending branch office.

Also, you need to execute only against records whose transaction time was in a certain range. And then, you must adjust certain of the other dates accordingly (but not all of them). STARWARP can handle this job and can fix your master file, either by making a copy that's corrected, by removing the bad records, or by actually adjusting the entries in the original file. All of this is done under programmer and management control.

STARWARP can handle data files of many types. VSAM files and large varieties of non-VSAM files can easily be dealt with. This is because STARWARP can draw on the enormous power built within the STARTOOL product, for detecting the properties of the data set it is pointing at. STARWARP "knows", in tremendous detail, the internal structure of that data set.

For instance, if STARWARP is pointing at a VSAM data file, it can generate another VSAM data set that's just like the original one, through its **MODEL** subcommand. **MODEL** internally generates IDCAMS statements that can define a new VSAM file identical to the old one. With **MODEL**'s panel controls, these can be selectively adjusted under user supervision. You see, when STARWARP is looking at a VSAM file, it has enormous knowledge of the structure of that file. This kind of internal file knowledge extends to the individual records, too. Thus, when STARWARP is instructed to make a copy of a data file, you can be assured that the structure of the copy is going to be accurate.

Now, let's see a very simple example, which shows some of the conditional logic that STARWARP has. For learning purposes, it's best to look at this from the STARWARP ISPF interface, assuming that we've imported a copybook which describes the data in a file. Let's say that the branch office field is 2 characters starting in column 1, and there's a date field of 8 characters starting in column 12. We know that the date field is in CCYYMMDD format, but the copybook has it only as PIC X(8), the way the COBOL program had it.

## STARWARP User Guide

We'd like to pick only records with 'NC' in the branch office field in column 1, and we'd like to age all the dates in column 12, by 7 days, just for the records from that branch office. What do we do? First, from the STARWARP main menu, we select option **4A**, or WARPCOPY, to select the copybook that will allow us to make our specifications.

From the **4A** selection panel, we specify where the table has been saved as a file, and a few other things. Then, when we press ENTER, the copybook table appears on our screen, field by field, with places on the left side to enter line commands.

There are 2 kinds of line commands we'll use, **S** to select conditions, and **W** to warp dates. For our example, first we'll select the condition, and enter an **S** next to the 2-byte field in column 1. The screen to select conditions will appear. We enter 2 things for this field: the conditional verb, **EQ** (for "equals" in this case), and the value of the field to be tested against. In our case, the value is a "character field" consisting of the letters **NC**. We enter this as **C'NC'**. After we press enter, STARWARP takes us back to the copybook table again.

Now we have to enter the instructions to warp the date. Next to the line in the copybook table that describes the 8-byte date field starting in column 12, we place a **W**, and press ENTER. A "warp" screen will appear. We enter the format of the date (CCYYMMDD - the copybook doesn't know as much about date fields as STARWARP does, only about data field formats). Then we select the action, which is **ADD**. Then we tell STARWARP what to add, which is **7D**, or 7 days. When we've finished entering data on this screen, we press ENTER, and we go back to the copybook table screen.

Since this is a simple example and we've completed entering all the conditions, we now enter a primary command at the top of the table: **BUILD**. STARWARP takes us into the BUILD screen, where we'll generate a STARBAT batch job to run against our data file.

In the BUILD screen, we specify jobcard JCL, where the input data file is, where the output data file will be, and what actions we want STARBAT to take. Among the possible actions are: **COPYSOME**, which will copy only the records that fit the conditions specified for change, **COPYALL**, which will copy all the records to a new file, and will "warp" the ones which fit the conditions, and **UPDATEALL**, which will update the "warped" records in place. You decide what to do. **COPYSOME** is the default.

After you've pressed ENTER in the BUILD screen, STARWARP will get you into an EDIT session on the generated STARBAT batch job. You can then change the job and/or SUBMIT it, or CREATE a file to save all of the STARBAT JCL. After you run this batch job, the data will be copied and warped, or updated and warped, depending on what you had specified.

We've just shown an example which displays the most primitive of STARWARP's capabilities. In actuality, you can run extremely complicated manipulations to a data file, adjusting the dates the way you want to. The key thing to remember is: "STARWARP knows dates. Application programs don't." Application programs only know the picture formats of date fields. STARWARP knows "dates as dates". And STARWARP knows how to manipulate dates in a dazzling array of different formats, and it can change dates consistently, in many different ways. STARWARP can meet the date processing needs, for your business data.

So we've completed our introduction. Now you've entered the great new frontier of modern date processing with business data. Your mission--to go where no one has gone before. You may now proceed, "steady as she goes," at **WARP** speed.

## General Overview

Data centers are more difficult to manage today than ever before. The expanding volume and distribution of data, coupled with tight budgets, downsized staffs, and the pending century change create many new challenges. STARWARP meets these challenges by providing a single workbench that can be used across the entire data center, by both applications and year 2000 change specialists.

STARWARP provides a comprehensive and easy-to-use environment for editing and managing PDS, PDSE, sequential, direct access and VSAM files, and their contents (source, copybooks, JCL, data and load modules). With STARWARP, complex data and file management tasks are simplified and the need for multiple tools, batch utilities and homegrown routines is eliminated.

STARWARP is a comprehensive programmer's workbench, with over 100 integrated facilities that perform data set and member management for IBM data sets. STARWARP offers powerful facilities to manipulate data sets, individual members and groups of members using IBM's strategic ISPF dialog manager, native TSO (**READY** mode), or batch processing.

In the familiar ISPF environment, programmers have access to the latest technology. STARWARP implements pull-down menus, pop-up informational windows, scrollable panels, entry assist panels, and field-level help to make its facilities easier to use.

STARWARP provides programmers an edge in managing data sets by providing facilities to logically group data sets for easy access and it provides global edit and search facilities for operations on individual data sets or groups of data sets. File maintenance activities (which are usually lengthy and error prone), can be performed very quickly by STARWARP. With STARWARP' data set integrity checking, trouble-shooting tools and performance enhancements, programmers gain an extra edge in meeting the technical demands in the data center.

STARWARP must always have a current data set; the **CHANGE** subcommand is used to specify a different current data set. STARWARP supports multiple active data sets in parallel sessions; sessions are activated and selected with the **GO** function and they may be terminated with the **DROP** command.

ISPF programmers no longer have to execute numerous data set and member utilities to perform daily maintenance activities. STARWARP eliminates this work by organizing their desktop with multiple utilities. Data set lists, an application similar to ISPF option 3.4, may be built using numerous high-level qualifiers and can be saved and recalled across ISPF sessions with the STARWARP LISTC/LISTF application. Also, many batch utilities like AMBLIST, IDCAMS and IEHLIST are incorporated into STARWARP. STARWARP can also display the linkage-edit date of load modules within a member list, map the structure of load modules and list the compile options for COBOL programs.

For a comparison of STARWARP and the IBM batch utilities, see **Appendix A. IBM Batch Utilities Comparison** on page 99; to compare STARWARP and different ISPF utilities, see **Appendix B. ISPF Utilities Comparison** on page 101.

### ***Environment***

STARWARP operates in three distinct environments:

- as an interactive dialog under ISPF
- as a TSO command processor in "READY" mode
- as a batch dialog under the TSO Terminal Monitor Program

Each mode has various advantages, but ISPF mode processing (or "ISPMODE") offers the largest programmer productivity advantages.

As an interactive dialog under ISPF, STARWARP combines its powerful data set and member manipulation features with the usability of ISPF. STARWARP takes full advantage of the facilities available in ISPF 3.3 and above; STARWARP is a CUA-compliant application, with pull-down menus, pop-up informational windows, scrollable panels and field-level help features designed to improve its usability. Multilevel menuing with overtypable data entry panels, comprehensive tutorials and command syntax-assist panels provide the interfaces to quickly learn the power available in STARWARP.

### ***ISPMODE Execution***

As programmers become more familiar with STARWARP, they can become more effective and use ever more powerful techniques. STARWARP provides intuitive, easy to use panels for standard users but some time must be spent traversing these panels and replying to panel prompts. STARWARP is less intuitive for advanced users who select assistance panels as needed but they gain in overall effectiveness by reducing prompting and increasing parallel function usage. Power users can use STARWARP in the most direct, effective fashion by entering subcommands directly and by using STARWARP in all environments.

These different STARWARP user modes are actually more descriptive than restrictive. A new STARWARP user can choose to enter certain subcommands directly, thereby becoming a power user for those subcommands. A very experienced STARWARP user can choose to obtain any level of prompting or subcommand assistance for facilities of STARWARP where additional prompting is desired.

### **Standard Users**

This manual is oriented to standard mode users. If you wish to utilize advanced or power modes after learning to use STARWARP, you should refer to tutorial panels for a full discussion of all modes of STARWARP operation.

Standard mode programmers use the menu system extensively. These users expect hierarchical behavior due to their ISPF background and they do not mind multiple screen prompts to establish functions.

The menu system is used to initiate member lists (from menu options 1 or 2) and data set lists (from menu option 3.4); line commands are used in each of these functions to manipulate individual members and data sets. Standard mode users generally understand the concepts of member groups and they quickly learn how to select members by contents or attributes and perform other actions on them.

Some programmers choose to enter STARWARP to perform a specific task and exit back to their starting point; others utilize STARWARP for almost all of their work. STARWARP can accommodate many working styles.

When invoking STARWARP for a data set in ISPMODE, the first screen is called the primary menu panel.

```

----- StarWarp Version 6.1.0 Primary Options -----
OPTION  ==> 10

Current data set ==> LIB.CNTL

Basic Applications:
 0 Parameters - Specify StarTool parameters      USERID - WSER50
 1 Browse    - Display source data or output listings PREFIX - WSER50
 2 Edit      - Create or change source data      TIME   - 16:20
 3 Utilities - Perform utility functions         DATE   - 2000/01/0
 4 Warp      - StarWarp Year 2000 options        JUL DATE - 2000.001
                                           TERMINAL - 3278
                                           LOG PROC - ISPFPROC
                                           MVSID   - P390
                                           APPLID  - ISR
                                           TRIAL   - 48 days

Data Warping Applications:
4A WARPCOPY - Create a job from a copybook
4B WARPSAVE - Create a job from a saved table
4C PROFMAN  - Manipulate StarWarp saved tables
4D DATECALC - StarWarp date calculator
4E EUROCALC - StarWarp Euro conversions
4F DATEDIFF - StarWarp date difference calculator
4G HOLIDAYC - Create a StarWarp holiday calendar

Analysis Applications:
4H IMPACT   - Reconcile load members with source
4I COMPCHK  - Check for compiler usage
4J CSECTCHK - Check like CSECTS consistent uses
4K LANGTYPE - Mark languages used in a MEMLIST
4L HISTORY  - Display member CSECT history
4M MAP      - Display member CSECT structure
4N XREF     - Display inter-module references

Intermediate Applications:
 5 Batchjcl - Create a background StarTool job
DE Demo     - Demonstrate StarTool facilities
I# ISPF     - Stack an ISPF session (like I3.4)
IN Index    - Display StarTool tutorial index
 N New      - Summarize changes for this release
 X Exit     - Terminate StarTool

Advanced Applications:
 6 All      - Create a MEMLIST with all members
 7 Output   - Output LOG Table
 8 Log      - View StarTool session log
 9 Activate - Activate data sets and GO sessions
10 Search   - Search and Replace functions

```

**Figure 2. Sample STARWARP Primary Menu**

This primary panel is scrollable so most of these options do not appear until you scroll forward. As an example, if you enter **10** as shown above on the primary menu panel to perform "**Search and Replace Functions**" and press **ENTER**, the following search selection panel is displayed. This panel is subordinate to the primary menu panel; you may back up in the panel hierarchy with an **END** command; however, you will usually make a further selection and press **ENTER**.

```

----- StarTool Search and Replace Selections -----
OPTION  ==> 9

Choose one of the following:
 1 - MODULE - search system for module name
 2 - ADDRESS - search system for module using memory address
 3 - SEEKDS - search group of libraries for a member
 4 - FINDDS - search group of libraries for members containing a string
 5 - REPLDS - update members in a group of libraries
 6 - SEEKCON - search data set concatenation for a member
 7 - FINDCON - search data set concatenation for members containing a string
 8 - REPLCON - update members in a data set concatenation
 9 - FIND - search current source data set for members containing a string
10 - REPL - update current source data set members which contain a string
11 - IF - search current source data set for members with given attributes
12 - VERIFY - search current source data set for members with errors
14 - IMPACT - reconcile load library members with associated source libraries
15 - COMPCHK - check a load library for compiler usage
16 - CSECTCHK - check a library for like CSECTS with different compile dates
17 - LANGTYPE - check a source library for languages used and mark a MEMLIST

Current data set is WSER07.LIB.CNTL,VOL=SER=SER005 MEM=ABC*

```

Figure 3. Sample Search and Replace Selections

Then, if you enter **9** on the above panel to "search current data set" and press **ENTER**, an entry assist panel like the following example will be displayed. An entry assist panel is normally used to specify command parameters; this information will be used by the panel to format a STARWARP subcommand which will perform the requested function. Entry assist panels usually have some default information preinitialized in the panel; you may change any input field as desired by typing over any existing information. If this panel is displayed later in this same STARWARP session, previously changed values should be remembered.

```

----- FIND: Search Source Members -----
OPTION  ==>

- DSN=WSER07.LIB.CNTL,VOL=SER=SER005 MEM=ABC* -----
Member or member group ==> copy/ (*, start*, start:end or part/)
Find string ==> dsn= (may be 'quoted')
Format for listing ==> NUM (NUM, SNUM, NONUM, LBLOCK, LDUMP, BLOCK, DUMP)
Find conditions ==> (WORD, PREFIX or SUFFIX)
List all data after find==> NO (YES or NO)
Maximum records input ==> (1 to 99999999)
Maximum records output ==> (1 to 99999999)

```

Figure 4. Sample Entry Assist Panel

On color terminals, input fields (just after each ==> marker) are displayed in red by default. To position the cursor to any field on an entry assist panel, the tab down key is most effective; however, the arrow keys can also be used. Most fields in an entry assist panel can be blanked with the erase-EOF key and if a choice of keywords is provided, you need only overtype enough to make your choice unambiguous. For example, if **LBLOCK** is displayed and your choices include NUM, SNUM and NONUM, changing the entry to **noLOCK** is sufficient to specify NONUM.

For additional information while in an entry assistance panel, press **HELP** for tutorials on that subcommand. Some fields on entry assist panels (like the "Member or member group" line above) have their input arrow (==>) marked in a special color (green by default) to indicate that field-level or pop-up help is available for that field; it may be obtained by placing the cursor in the field and pressing **HELP**. If the cursor is in another field which does not have field-level help, the **HELP** command will provide the normal subcommand tutorials.

After pressing **ENTER** on an entry assist panel, you will normally transit to the log where the formatted subcommand is displayed prefixed by **>----->** (this demonstrates STARWARP subcommand syntax), followed by the output of the subcommand.

After you have examined the output of the subcommand in the log, you may press **END** to return to the entry assist panel and make corrections and retry the subcommand; each **END** command will back up one additional level in the panel hierarchy. If you want to go to the menu primary panel directly, you may enter the **MENU** command or you may enter a *chained* command such as **M.3.4** or its equivalent, **3.4** (numbered commands are assumed to be references to the menu system when operating in the log or any other table panel).

```

FUNCTIONS CONTROL DSN CMDS MEM CMDS A-M MEM CMDS N-Z DEFAULTS FEATURES
-----
----- ISPMODE Session# 1 Log# 1 -- ROW 162 TO 179 OF 245
COMMAND ===> SCROLL ===> CSR
- DSN=SER07.LIB.CNTL,VOL=SER=SER006 MEM=COPY/ -----
>----->Find copy/ 'dsn=' NUM CAPS

** FIND IEBCOPY
000900 //SYSUT1 DD DSN=PDSE.MAPS,DISP=OLD,UNIT=3480,
001100 //SYSUT2 DD DSN=SER09.MODS2.ASM,DISP=SHR
PDS142I 10 lines in this member
PDS146I 2 strings found

** FIND IEBCOPYA
000270 //SYSUT1 DD DSN=QUICKREF,DISP=OLD,UNIT=3480,
000290 //SYSUT2 DD DSN=SER07.QUICKREF.JCL,DISP=(,CATLG),
PDS142I 14 lines in this member
PDS146I 2 strings found

** FIND IEBCOPYB
000700 //SYSUT1 DD DSN=OXTS207.PDSTOOLS.STOW.COPYDAT.TAPECPY,UNIT=3480,
000800 //SYSUT2 DD DSN=SER07.LIB.STOW,DISP=(,CATLG,DELETE),UNIT=SERDA,
PDS142I 14 lines in this member

```

**Figure 5. Sample Log Panel**

In the figure above, the data from “>----->Find...” through the last line in the display is log data. The log is used to track subcommands entered and the output from subcommands in a scrollable ISPF table. An ISPF command such as **HELP** would not be tracked in the log but a STARWARP subcommand such as **FIND** and the results of the **FIND** would be placed in the log.

Subcommands are identified in the log by a **>----->** string. Even though the log supports up to 159 characters in each log line, if an echoed subcommand does not fit within the first 79 characters of a log line, continuation lines will be added as needed; each continuation line is identified with an initial - character.

Several facilities are available for manipulating the log.

- The output from the last subcommand can be manipulated in an edit session with the **EDITLOG** (or **EL**) command.
- The log may be directed to a data set or a system printer with the **OUTPUT** command.
- The entire log can be manipulated in an edit session with the **EDITTBL** (or **ET**) command.
- Any character string can be searched for in the log with the **F** command; you may press **RFIND** or **PF5** for repeat searches.
- Other views of the log can be selected by pressing **LEFT** or **PF10** and **RIGHT** or **PF11**.

If the cursor is positioned over a STARWARP message (**PDSnnn**) in the log, you may press **RCHANGE** or **PF6** to obtain an explanation of the message.

### **Batch Mode**

STARWARP can operate under the batch Terminal Monitor Program (TMP). Execution is very similar to TSO command processor mode except that user interaction is not supported. To execute a script in batch, several subcommands may be placed in the input stream or in CLIST or REXX command lists. This mode of execution is very effective for performing regular maintenance functions such as validating libraries and searching for unusual members or data sets. ISPMODE functions are available in batch mode; however, panel prompting is not supported and line commands are supported only by the global APPLY command. Multiple data sets are easily processed using ISPMODE facilities in batch through the use of global commands.

For more information on batch execution, see **Batch Processing** on page 77. Also, see the **STARBAT Guide** for more information on the STARWARP batch control statement emulation program.

### **Invoking STARWARP**

STARWARP must always point to a data set. This data set is referred to as the current data set of the STARWARP session; you may have more than one current data set with the STARWARP **GO** function, but there must be at least one current data set pointed to by STARWARP. The current data set can quickly be changed using the **CHANGE** subcommand. If the data set name is omitted, the last data set you used in a STARWARP will be used again..

STARWARP is normally invoked as an ISPF command processor as shown below:

```
TSO STARWARP 'data.set.name'
```

STARWARP can also be invoked directly through a primary panel option, and the introductory dialog panel will appear to prompt you for data set names in what is called ISPMODE operation.

ISPMODE operation of STARWARP consists of a sequence of panel displays that are invoked one from another, depending on choices presented, and from which you decide where to go. STARWARP ISPMODE dialog operation is therefore no different from any other ISPF dialog. The normal rules and functions of ISPF apply. STARWARP in ISPMODE uses the facilities of ISPF to accomplish its aims.

STARWARP can also be invoked from an ISPF data list (ISPF option 3.4) by typing **STARWARP** or **PDSE** as a line command for a data set and STARWARP will honor the data set volume name. Normally, ISPF line commands do not honor the volume name. In general, ISPF and STARWARP help each other and do not interfere with each other.

### **Single Subcommand Mode**

STARWARP can also be invoked in “single subcommand mode” by including a subcommand and any required operands after the data set name and STARWARP will only perform the single subcommand and terminate.

ISPMODE subcommands such as **MEMLIST**, **ISPMODE** and **ISPXEQ** are exceptions; they invoke STARWARP in an ISPF dialog mode as expected. Following is an example of invoking STARWARP in single subcommand mode from an ISPF environment (note: from **READY** mode, leave off the **TSO**).

```
TSO STARWARP data.set.name history : notrans
```

This is a special subcommand mode for performing only a single subcommand; ISPF services are not available and STARWARP operates only in line mode. Also, no YES/NO prompting is provided; YES responses are assumed. Be aware that when STARWARP is invoked in single subcommand mode, the return code is set to the numeric value of the first warning or error message encountered. For example, if the subcommand encountered a PDS820E message, the return code would be set to 820 in decimal.

## **StarWarp Subset of StarTool**

STARWARP is a subset of STAR TOOL which provides many resources for installations converting programs and data files for the year 2000. STARWARP is an information tool; while the full STAR TOOL product adds program and data modification facilities.

In general, those subcommands which inspect and report on programs and data are in the STARWARP subset but the following subcommands which modify programs and data can not be used in a STARWARP environment.

### **Restricted Subcommands**

<b>Subcommand</b>	<b>Abbreviation</b>	<b>Function</b>
ACFCOMP	ACF	Compile source CA-ACF2 rules
ALIAS	AL	Add an alias name for a member
COMBINE	COMB	Join several members together
COMPRESS	COMPR	Compress data sets
COPY	COP	Copy data selectively
DELETE	DEL	Delete members
DUP	DU	Copy data selectively
FIXPDS	FIX	Modify a data set
LLA	LLA	Refresh LLA directory entries
OUTCOPY	OUTC	Format utility control statements
PEDIT	PE	Edit data with Parallel Edit
PVIEW	PV	Edit data in VIEW mode of Parallel Edit
RENAME	REN	Change a member's name
REPRO	REPR	Rebuild or create members
RESTORE	RES	Resurrect deleted members
SEPARATE	SEP	Split into members
SMPGEN	SM	Generate SMP/E sysmods
VMAP	VM	Display a disk volume map
ZAP	Z	Modify load module

## Using Subcommands

If you attempt to use a restricted subcommand, STARWARP will prompt for input as usual and format subcommands and options; however, STARWARP will not permit the formatted subcommand to execute.

For example, DELETE is one of the restricted subcommands.

```
----- ISPMODE Session# 1 Log# 1 ----- Row 74 to 77 of 77
COMMAND ==> delete                               SCROLL ==> CSR
- DSN=WSER07.LIB.CNTL,VOL=SER=SER001 MEM= -----
PDS200I DISP UNIT OPT RECFM LRECL BLKSIZE  ALLOCTRK FREETRK  SECONDARY FREEDIR
PDS200I SHR  3380 C   FB           80   11440   1X  1000     655   200 TRK     67
***** Bottom of data *****
```

**Figure 6. Sample Restricted Subcommand**

STARWARP prompts for DELETE input in the following panel.

```
----- DELETE: Delete Members -----
OPTION ==>
- DSN=WSER07.LIB.CNTL,VOL=SER=SER001 MEM= -----
Member or member group ==> abc*           (*, start*, start:end or part/)
Delete associates also ==> NO             (YES or NO)
Delete even if under EDIT==> NO          (YES or NO)
```

**Figure 7. Sample Subcommand Prompting**

Next, STARWARP issues an error message and shows which major subcommands are available.

```
----- ISPMODE Session# 1 Log# 1 ----- Row 78 to 87 of 87
COMMAND ==>
- DSN=WSER07.LIB.CNTL,VOL=SER=SER001 MEM= -----
>----->Delete ABC* NOALIAS
PDS771E DELETE has been disabled for StarWarp; the following are valid
subcommands: ATTRIB  BROWSE  CHANGE  COMPARE  COMPDIR  CONTROL  CREATE
                DISPLAY  DISASM  DIRENTRY  DSNAME  DCF      EDIT     EDREC
                END     EXEC   FIND    FSE     HELP    HISTORY  IF
                ISPF    ISPMODE  ISPXEQ  ISPXEQS  LIST    MAP      MEMBERS
                MEMLIST  OPTIONS  PATTERN  PRINT   RECALL  REVIEW   SUBMIT
                SUBLIST  SVCMAP  TSO      TSOEDIT  TSOLIST  USAGE   VERIFY
                VPRINT  XREF
***** Bottom of data *****
```

**Figure 8. Sample Subcommand Error Message**

## The Menu System

The **MENU** (or **M**) system provides multi-level menuing with overtypable data entry (or entry assist) panels. These panels usually format and enter STARWARP subcommands or invoke STARWARP functions.

The menu system was designed to be similar to ISPF. The STARWARP Primary Options panel is normally displayed on entry to STARWARP. Note that this panel is scrollable for even more options.

```

----- StarWarp Version 6.1.0 Primary Options -----
OPTION  ==>

Current data set ==> LIB.CNTL

Basic Applications:
 0 Parameters - Specify StarTool parameters          USERID - WSER50
 1 Browse     - Display source data or output listings PREFIX - WSER50
 2 Edit       - Create or change source data          TIME   - 16:20
 3 Utilities  - Perform utility functions             DATE   - 2000/01/0
 4 Warp       - StarWarp Year 2000 options            JUL DATE - 2000.001
                                                    TERMINAL - 3278
                                                    LOG PROC - ISPFPROC
Data Warping Applications:
4A WARPCOPY  - Create a job from a copybook          MVSID  - P390
4B WARPSAVE  - Create a job from a saved table       APPLID - ISR
4C PROFMAN   - Manipulate StarWarp saved tables      TRIAL  - 48 days
4D DATECALC  - StarWarp date calculator
4E EUROCALC  - StarWarp Euro conversions
4F DATEDIFF  - StarWarp date difference calculator
4G HOLIDAYC  - Create a StarWarp holiday calendar

Analysis Applications:
4H IMPACT    - Reconcile load members with source
4I COMPCHK   - Check for compiler usage
4J CSECTCHK  - Check like CSECTS consistent uses
4K LANGTYPE  - Mark languages used in a MEMMLIST
4L HISTORY   - Display member CSECT history
4M MAP       - Display member CSECT structure
4N XREF      - Display inter-module references

Intermediate Applications:
 5 Batchjcl  - Create a background StarTool job

```

**Figure 9. Sample Primary Menu Panel**

From anywhere in the STARWARP menu system, you can invoke the primary menu panel by entering **M.** followed by any data for the menu system. For example, you could enter **M.FIND** to transit to an entry assist panel for the FIND subcommand.

From any function in STARWARP, numbers are assumed to reference the menu system. There are many short-hand conventions for entering commands as shown in the examples below:

**3.4** is equivalent to **M.3.4**

**=3.4** is normally equivalent to **M.3.4** (see "Internal Jump" in SETALL)

**M.FIND** is equivalent to **M.12.FIND** because option 12 is assumed for a subcommand reference.

**FIND** is equivalent to **M.12.FIND** as most subcommands entered without operands invoke the menu system.

## STARWARP User Guide

**M** (for MENU) is also available as a line command; from a source or load member list, you may enter **M** as a line command for a member to get a directory of available line commands. When you select one of these items, you will get the entry assist panel for that subcommand. You may also enter a chained line command such as **M.FI** (for FIND) to bypass the directory prompting panel.

The menu system is designed to make the power of STARWARP available to all programmers. You can transfer your ISPF skill set directly into the STARWARP environment with no additional training. With the menu system's entry assist panels, you can overtype fields and the panels will format and enter subcommands to perform STARWARP functions. The primary menu panel is logically divided into option groups which perform different types of functions as explained below.

- 0 through 3**     Parameters, Browse, Edit and Utilities are similar to corresponding ISPF options. Results are usually displayed with the EDITLOG facility; control normally remains in the menu system.
- 4 through 5**     Warp and Batch support the STARWARP data warping applications, analysis applications and STARWARP in batch functions.
- DE through X**   Demo, ISPF, Index, New and Exit provide various support services. Control normally returns back to the primary menu panel for each of these options except for Exit.
- 6 through 8**     All, Output and Log perform specialized STARWARP functions. Control normally transits to tutorial panels first but these can be suppressed (see SETPANEL).
- 9 through 11**    Activate, Search and Recover provide easy access to STARWARP subcommands and functions. Control is usually received in the log to display results. You can usually press END to return to the last panel displayed; then, you can retype any fields desired and press ENTER to reexecute a subcommand or you can continue backing up in the panel hierarchy by pressing END.
- 12**                "Subcommands" varies according to the active data set type (Direct, Sequential, Source PDS/PDSE, Load PDS/PDSE or VSAM). Only subcommands supported for the active data set type will be listed for this option. Option 12 is automatically selected from the primary panel or from any of the STARWARP functions if a subcommand is entered without operands.
- 13 through 15**   Examine, Profman and Global provide advanced STARWARP subcommand and function support. Control is usually received in the log where results are displayed. You can normally press END to return to the last panel displayed; then, you can retype any fields desired and press ENTER to reexecute a subcommand or you can continue backing up in the panel hierarchy by pressing END.

Individual items from the primary menu panel are explained in more detail in the following:

- OPTION** ==>     provides an place to enter commands. You may enter a STARWARP function, command, dynamic command, subcommand or a numbered option.
- Current data set:** provides a convenient way to change to a different current data set. You may tab to this field and enter any one of the following:
- |                      |  |
|----------------------|--|
| <b>data set name</b> | formatted as for an ISPF "Other data set" field. Note that uncataloged data sets are not supported here but you can use option <b>9</b> for these data sets. |
| <b>FILE(ddname)</b>  | to change to a preallocated DDNAME. Note that a NUMBER(n) parameter may also be entered to select the nth data set in a set of concatenated data sets.       |
| *                    | to change to the previous data set.  |
| <b>blank</b>         | to request a data set prompting panel.   |

<b>0 - Parameters</b>	provides a menu for the STARWARP SET panels.
<b>1 - Browse</b>	provides a method to specify a new current data set for STARWARP. If a member group name is specified, STARWARP just builds a member list of those members; otherwise, ISPF Browse is invoked on a single member.
<b>2 - Edit</b>	provides a method to specify a new current data set for STARWARP. If a member group name is specified, STARWARP just builds a member list of those members; otherwise, ISPF Edit is invoked on a single member.
<b>3 - Utilities</b>	provides a method to specify a new current data set for most subordinate utility functions. STARWARP performs functions corresponding to the equivalent ISPF 3.x options.
<b>4 - Warp</b>	supports all STARWARP data warping applications and analysis applications.
<b>5 - Batch</b>	creates background jobs; controls are available to build batch JCL from allocations in the DDNAME/LISTA table, build a STARWARP batch job from subcommands in the log and manage the DDNAME/LISTA table.
<b>DE - Demo</b>	demonstrates STARWARP facilities on data sets created for the demonstration.
<b>I# - ISPF</b>	stacks an ISPF dialog; for example, <b>I3.4</b> is equivalent to <b>ISPF 3.4</b>
<b>IN - Index</b>	enters the STARWARP tutorial index.
<b>N - New</b>	displays a summary of changes for the current release as well as older releases.
<b>X - Exit</b>	terminates STARWARP (note: this option displays the number of current active GO sessions).
<b>6 - All</b>	builds a member list containing all data set members; this is equivalent to the <b>ALL</b> command.
<b>7 - Output</b>	outputs the current dialog table; this is equivalent to the <b>OUTPUT</b> command.
<b>8 - Log</b>	switches from the menu panel to the log display; this is equivalent to the <b>LOG</b> command.
<b>9 - Activate</b>	controls GO sessions and allows changing the current data set.
<b>10 - Search</b>	provides dialogs to search and update several different types of data.
<b>11 - Recover</b>	provides several dialogs to recover and repair the current data set.
<b>12 - Subcommands</b>	provides a menu for subcommand entry assist panels based on the current data set organization.
<b>13 - Examine</b>	controls several specialized subcommands that display environmental information.
<b>14 - Profman</b>	manages saved copybook, MEMLIST, LISTC/LISTF and WORKPAD tables.
<b>15 - Global</b>	controls global commands for DDNAME/LISTA, LIST/LISTF, LISTV and WORKPAD. You can create or extend the tables and initiate global processing.

From STARWARP table panels, an **END** command will normally invoke the menu primary panel so that you may choose your next option.

## STARWARP User Guide

After data is entered on an entry assist panel, the panel logic formats a STARWARP subcommand or function and enters the command. You are then placed in one of the STARWARP functions so that you can review the results of the subcommand. For STARWARP subcommands, you will normally transit to the log where the formatted subcommand and its output are displayed.

When you have reviewed the results of the subcommand, you can normally press **END** to return to the last menu panel displayed since the menu system operates in hierarchical mode (normally, STARWARP operates in parallel mode). Note that hierarchical mode is only maintained until a parallel mode command is entered. For example, if you are examining the log, you may enter any of the following commands without interrupting hierarchical mode: **F, OUTPUT, RIGHT, LEFT, UP, DOWN, EDITLOG, EDITTBL, X**; however, if you enter a mode switch command or function command like **ML** or a subcommand such as **FIND**, you begin operating in parallel mode. This means that a following **END** command would transit to the menu primary panel instead of the last menu panel displayed.

In the menu panels, an **END** command backs up a single panel. If **END** is entered from the primary menu panel, the current GO session is dropped; however, you will be asked to confirm termination of each GO session if your **SETALL** value for "termination prompting" is **YES**. If **END** is entered for the last GO session, you will always be asked to confirm termination since you are leaving the STARWARP environment. If you want to terminate STARWARP with no prompts, enter an **X** command from the menu primary panel.

Note that jump commands (like **=3.4**) are supported and they normally behave as expected for a primary panel (jump commands request STARWARP options) unless "Internal Jump" in SETALL is set to **NO** (then, jump commands terminate STARWARP and request ISPF options). Note that as a convenience, STARWARP also supports internal jump commands if you enter a blank between the equal sign and the number (for example, **= 3.4**).

- If "Internal Jump" is YES
  - ◆ **=X** from any STARWARP panel will terminate STARWARP
  - ◆ **=3.4** will execute STARWARP option 3.4
  - ◆ **RETURN** will transit to the primary panel
  
- If "Internal Jump" is NO
  - ◆ **=X** from any STARWARP panel will terminate STARWARP and execute ISPF option X
  - ◆ **=3.4** will terminate STARWARP and execute ISPF option 3.4
  - ◆ **RETURN** will terminate STARWARP

---

---

## Guide to Data Aging

---

---

### Objective

In this section, we provide the new STARWARP user a tour of STARWARP data aging capabilities by example. If you read this section and follow the STARWARP online tutorials, you will have all the tools necessary to execute the STARWARP dialog and create STARBAT control statements to execute data aging jobs.

### Basics

STARWARP must always point to a data set; it is referred to as the "current data set". If a data set name is not specified when you start a STARWARP session, the last data set you used in a STARWARP session will be used.

In a STARWARP session, you may get help with the **M** (for Menu) line command. This will provide entry assist panels which provide panels with entry fields like the following example:

```
Find conditions ==> (WORD, PREFIX or SUFFIX)
```

If the entry field arrow is displayed in green (instead of the normal turquoise), that means that you can place the cursor in the entry field and press **HELP** or PF1 to get a pop-up help panel for that field. If the arrow is turquoise, pressing **HELP** or PF1 will get a more general help panel for the current panel or topic instead.

You may also enter an **O** (for Option) primary or line command. This will invoke entry assist panels which help you enter STARWARP commands by displaying command syntax. If you need additional information, press **HELP** or PF1 for a tutorial on the current subcommand.

All subcommands and their output are written to the log that can be directed to a data set or a printer using the **OUTPUT** subcommand.

## Scrolling

Table panels like the following example are scrollable as indicated by the "**Row 1 to 6 of 26**" marker. This means that there is more information below the visible part of the panel and you may scroll forward by pressing **DOWN** or PF8. After you scroll forward, the Row marker will change to indicate your position in the table; and you may press **UP** or PF7 to scroll backward.

```

----- Copybook for StarWarp ----- Row 1 to 6 of 26
COMMAND ==>                               SCROLL ==> CSR_

CMD NAME                                PICTURE    COLUMN    TYP    LENGTH
-   01 TESTBOOK
-   02 MYFILE-SKIP                       X(1) @ 1    CHR     1
-   02 MYFILE-IDENTIFIER                  X(14) @ 2   CHR    14
-   02 MYFILE-KEY
-   05 MYFILE-KEY-FIRST2                  X(2) @ 16   CHR     2
-   05 MYFILE-KEY-REST                    X(7) @ 18   CHR     7
    
```

**Figure 10. Sample Table**

In the following example scrollable panel, note the "**More: +**" marker. This means that there is more information below the visible part of the panel and you may scroll forward by pressing **DOWN** or PF8. After you scroll forward, the More signal will add a minus to indicate that you may press **UP** or PF7 to scroll backward. If the More signal is indicated by "**More: - +**", you may scroll in either direction.

```

----- Sample options -----
OPTION ==>

Date or data picture ==>                (descriptive picture, like CCYY-MM-DD)
Type of action      ==> ADD              (ADD or SUB to increment; DATE to insert)
                                                More: +
Date adjustment     ==>                (number and code like 90D or actual date)
Corrective action   ==>                (action to take, like FLAG)
    
```

**Figure 11. Sample Scrollable Panel**

The following example scrollable tutorial panel resulted from placing the cursor in the "**Date or data picture**" field and pressing **HELP** or PF1. It is similar to a scrollable panel and contains a "**More: +**" marker. However, because it is a tutorial panel, you must scroll forward by pressing **ENTER, RIGHT** or PF11 and you scroll backward by pressing **LEFT** or PF10. **UP** and **DOWN** are directives for the tutorial display program.

```
----- Sample options -----
OPTION ==>

Date or data picture ==>          (descriptive picture, like CCYY-MM-DD)
----- Date or Data Picture -----
| TUTORIAL ----- Note: required -----
| COMMAND ==>
|
| The following date and data pictures are available for numeric elements:
| S99...      for numeric data with up to fifteen 9 digits and a sign
| YYDDD      for Julian dates with two year digits
| CCYYDDD    for Julian dates with four year digits
|
| More:      +
```

Figure 12. Sample Scrollable Tutorial Panel

### Option 4 -- Warp

Option 4 of the STARWARP primary panel includes all of the STARWARP options needed to age dates and analyze your programming environment for potential year 2000 problems.

The menu system was designed to be similar to ISPF. The STARWARP Primary Options panel is normally displayed on entry to STARWARP.

If this is the first time you have used STARWARP, the current data set will default to your ISPF profile data set. To change the current data set, you may overwrite the data set name after:

**Current Data Set ==>**

or enter **CHANGE** followed by the new data set name in TSO syntax as a primary command.

Remember, the "**More: +**" in the upper right hand corner indicates that there is more text below; this panel is scrollable for additional options. Press **DOWN** or PF8 to scroll forward or press **UP** or PF7 to scroll backward.

```

----- StarWarp Version 6.1.0 Primary Options -----
OPTION  ==> 4A

Current data set ==> ISPF.PROFILE

Basic Applications:
0 Parameters - Specify StarTool parameters          USERID - WSER50
1 Browse     - Display source data or output listings PREFIX - WSER50
2 Edit      - Create or change source data          TIME   - 16:20
3 Utilities - Perform utility functions            DATE   - 2000/01/0
4 Warp      - StarWarp Year 2000 options            JUL DATE - 2000.001
                                                    TERMINAL - 3278
                                                    LOG PROC - ISPFPROC
Data Warping Applications:
4A WARPCOPY - Create a job from a copybook          MVSID  - P390
4B WARPSAVE - Create a job from a saved table       APPLID - ISR
4C PROFMAN  - Manipulate StarWarp saved tables     TRIAL  - 48 days
4D DATECALC - StarWarp date calculator
4E EUROCALC - StarWarp Euro conversions
4F DATEDIFF - StarWarp date difference calculator
4G HOLIDAYC - Create a StarWarp holiday calendar

Analysis Applications:
4H IMPACT   - Reconcile load members with source
4I COMPCHK  - Check for compiler usage
4J CSECTCHK - Check like CSECTS consistent uses
4K LANGTYPE - Mark languages used in a MEMLIST
4L HISTORY  - Display member CSECT history
4M MAP      - Display member CSECT structure
4N XREF     - Display inter-module references

Intermediate Applications:
5 Batchjcl  - Create a background StarTool job
    
```

Figure 13. Sample Primary Menu Panel

**Option 4A -- WARPCOPY: Create a Job from a Copybook**

Select Option **4A** from the STARWARP Primary Options menu and press **ENTER** to create a job from a copybook.

```

----- Copybook Options for StarWarp -----
COMMAND ==>

  Copybook source      ==> COPYBOOK (COPYBOOK, EXTENDED, or TABLE)
                                                                More:  +
Table library options:
  WARP table name     ==>          (3 to 6 character name)
  Source library      ==> PROFILE (Profile, Group, Sample or Other)
  Other DSNAME        ==>
Copybook data set name:
  Data set            ==> pdsdemo.cntl
  Member name         ==> CBDEMO   (name of member if not included in dsname)
  Data set type       ==> MVS      (MVS, Panvalet or Librarian)
Optional:
  Language type       ==> EITHER   (COBOL, PLI or EITHER)
  Starting name       ==>
  Starting column     ==> 1        (1 to 32760 if first variable is offset)
Extended copybook options:
  Extended copybook   ==> TEMP     (Temp, Create, Mod, or Use)
  Copybook save name ==>          (3 to 6 character name)
  Source library      ==> PROFILE (Profile, Group, Sample, or Other)
  Other DSNAME        ==>

Notes:
  For additional specific information on any of the above fields, place the
  cursor in the field and press HELP for a detailed pop-up tutorial.

  Enter the Data set name in standard TSO syntax.  For a partitioned data set
  you should enter the member name in parentheses.  For example, my.data(abc)

  If Data set type is set to Panvalet or Librarian, you must enter a member
  name in the Member if PAN/LIB field.

  When the Language type is set to EITHER, PEDIT will try to determine the
  language type dynamically based on the statements scanned.

  If the Starting name is blank, the first 01 level name found will be used;
  enter a name if you want to override this action.

  The Starting column may be changed to override the starting data position.

```

**Figure 14. Sample STARWARP Prompt Panel**

## STARWARP User Guide

STARWARP parses the copybook and creates a copybook table with one row per copybook element. Then, STARWARP displays the copybook table as shown below.

```
----- Copybook for StarWarp ----- Row 1 to 24 of 26
COMMAND ==>                               SCROLL ==> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  MSG      NAME                                PICTURE COLUMN TYP  LENGTH
----  ---      ---                                -
01  TESTBOOK                                1          80
02  MYFILE-SKIP                            X(1) 1     CHR   1
02  MYFILE-IDENTIFIER                     X(14) 2     CHR  14
02  MYFILE-KEY                             16        9
05  MYFILE-KEY-FIRST2                     X(2) 16     CHR   2
05  MYFILE-KEY-REST                       X(7) 18     CHR   7
s_  02  MYFILE-PLACE-OF-BUSINESS            X(19) 25     CHR  19
02  MYFILE-DATE-OF-LAST-CHANGE            X(8) 44     CHR   8
02  MYFILE-FIRST-BINARY                   4 BIN 52     BIN   4
02  FILLER                                X 56      CHR   1
02  MYFILE-MISSION-KEYS                   57        4
05  MYFILE-KEY-ONE                        X(1) 57     CHR   1
05  MYFILE-KEY-TWO                       X(1) 58     CHR   1
05  MYFILE-KEY-THREE-FOUR                 X(1) 59     CHR   1
05  MYFILE-KEY-THREE-FOUR                 #2 60      CHR   1
02  FILLER                                XX 61      CHR   2
```

**Figure 15. Sample Copybook Table**

You should enter an **S** line command next to a copybook element as shown above if you want to test for specific data values before warping associated date elements.

A **SELECT** or **S** line command generates an **IF=**, **OR=** or **AND=** data test to control whether the subsequent WARP command will be processed using the following entry panel.

```

----- Select: IF, OR or AND Options -----
OPTION  ==>

Copybook LEVEL and NAME          PICTURE  COLUMN  TYP  LENGTH
      02 MYFILE-PLACE-OF-BUSINESS      X(19)   25      CHR    19

Type of selection    ==> IF    (IF, OR or AND)
Operator for compare ==> EQ    (EQ, NE, GT, LT, GE or LE)
Compare data        ==> C'Las Vegas'

Notes:
For additional specific information on any of the above fields, place the
cursor in the field and press HELP for a detailed pop-up tutorial.

Type of Selection: This field specifies the type of StarBat keyword that is
to be generated to select records for warping.

Operator for Compare: This field specifies the type of comparison desired to
select records for warping.

Compare Data: This field specifies the data values to compare against record
values to select records for warping.

Because multiple strings are allowed, the following example would be valid:
C'This is one type',C"Another type",P"-1234",X'C1C2C3'

```

**Figure 16. Sample Select Panel**

Note that you can position the cursor on each of the entry fields of this panel and press **HELP** or **PF1** to get a pop-up panel which explains that particular option.

## STARWARP User Guide

Press **ENTER** to return to the copybook table. If you need to select additional items before processing a **WARP** command, repeat the selection step for each additional test.

```
----- Copybook for StarWarp ----- Row 1 to 24 of 27
COMMAND ===>                               SCROLL ===> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  MSG      NAME                                PICTURE COLUMN TYP  LENGTH
----
_____ 01 TESTBOOK                                1          80
_____ 02 MYFILE-SKIP                            X(1) 1     CHR   1
_____ 02 MYFILE-IDENTIFIER                      X(14) 2     CHR  14
_____ 02 MYFILE-KEY                              16         9
_____ 05 MYFILE-KEY-FIRST2                     X(2) 16     CHR   2
_____ 05 MYFILE-KEY-REST                       X(7) 18     CHR   7
_____ *SELECT* 02 MYFILE-PLACE-OF-BUSINESS             X(19) 25     CHR  19
w_____ 02 MYFILE-DATE-OF-LAST-CHANGE         X(8) 44     CHR   8
_____ 02 MYFILE-FIRST-BINARY                  4 BIN 52     BIN   4
_____ 02 FILLER                                X 56     CHR   1
_____ 02 MYFILE-MISSION-KEYS                   57         4
_____ 05 MYFILE-KEY-ONE                       X(1) 57     CHR   1
_____ 05 MYFILE-KEY-TWO                       X(1) 58     CHR   1
_____ 05 MYFILE-KEY-THREE-FOUR                X(1) 59     CHR   1
_____ 05 MYFILE-KEY-THREE-FOUR                #2 60     CHR   1
_____ 02 FILLER                                XX 61     CHR   2
```

**Figure 17. Sample WARP Line Command**

After all the selection items have been entered, issue a **WARP** or **W** line command next to the copybook element that contains the data field for aging as shown above.

This will cause the following WARP options selection panel to be displayed. In this example, we specify a date format of CCYYMMDD and we want to add four years to each date field. If the resulting date is not a business day, we want to move the date to the next business day. An exit routine named CALEXIT is to be called to determine if a date is a holiday.

```

----- WARP options -----
OPTION  ==>

Copybook LEVEL and NAME          PICTURE COLUMN TYP LENGTH
      02 MYFILE-DATE-OF-LAST-CHANGE      X(8) 44      CHR      8

Date or data picture ==> CCYYMMDD          (descriptive picture like CCYYMMDD)
Type of action       ==> ADD              (ADD or SUB to increment; DATE to insert)
Date adjustment      ==> 4Y              (number and code like 90D or actual date)
Corrective action    ==> NEXTBDAY        (action to take, like FLAG)
                                                    More:      +

Optional:
Date validation      ==>                  (validation check, like WKEND)
Ignore Flags         ==>                  (LIST of code characters)
Pivot Year           ==>                  (0 to 99, for windowing two digit years)
Fiscal Month number ==>                  (1 to 12) 1=January, 12=December
Holiday exit name    ==> CALEXIT          (routine name for holiday date checking)
Error DDNAME         ==>                  (DDNAME to write records with errors)
Output picture       ==>                  (descriptive picture: CCYYMMDD)
Output format        ==>                  (B, C or P for binary, character, packed)
Output location      ==>                  (1-32760 for a column number)
Expand field         ==> NO              (Yes or No to expand the field)
Repeat for elements  ==> NO              (Yes or No to apply to same name elements)
Depending on prompt  ==> NO              (Yes or No to prompt for OCCURS DEPENDING

```

Figure 18. Sample WARP Panel

Note that you can position the cursor on each of the entry fields of this panel and press **HELP** or PF1 to get a pop-up panel which explains that particular option. Press **ENTER** to return to the copybook table.

## STARWARP User Guide

Note that if you scroll to the bottom of the copybook table, an entry for each of the specified STARWARP control cards will be displayed as shown below. You can select different panel views to display values that you entered by pressing **RIGHT** or **LEFT** PF keys; this will be demonstrated later. After you have specified all condition tests and warp criteria, enter a **BUILD** primary command as shown below to transit to a data set prompt panel.

```
----- Copybook for StarWarp ----- Row 23 to 28 of 28
COMMAND ==> build                      SCROLL ==> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  MSG      NAME                                PICTURE COLUMN TYP  LENGTH
----
_____ 02 MYFILE-HIT-ENTRY                    #2 76      BIN   2
_____ 02 MYFILE-HIT-ENTRY                    #3 78      BIN   2
_____ 02 LAST-DATA                          X 80      CHR   1
_____ *** START OF CONTROL CARD ENTRIES ***
_____ *IF      02 MYFILE-PLACE-OF-BUSINESS          X(19) 25    CHR  19
_____ *WARP    02 MYFILE-DATE-OF-LAST-CHANGE    X(8) 44    CHR   8
***** Bottom of data *****
```

Figure 19. Sample Control Statement Entries

In this panel, you can specify input and output data set names as well as JOB and JOBLIB statements for the STARBAT job which will perform the date warping. Press **ENTER** after you have made any desired changes to transit to an ISPF edit session with the generated JCL and STARBAT control statements.

```

----- StarWarp Data Set Prompt -----
COMMAND ==>

Foreground execution ==> NO          (YES for Foreground, NO for batch)
Input data set       ==> LIB.PDSE(STARBATT)
Output data set      ==> LIB.CNTLZ
Model the input      ==> NO          (Yes or No to Model the input data set)
Copy type operation ==> COPYSOME    (COPYSOME/COPYALL/UPDATEALL/COPY/UPDATE)
Maximum date errors ==> 100         (1 to 9999999, for invalid date messages)
Record Desc Word     ==> 3          (RDW, default 3)

More: +
Specify JOB and JOBLIB statements for the batch job to be created:
==> //WSER07J JOB (X170,374),MYJOB,CLASS=A,
==> //  MSGCLASS=X,TIME=(0,22),NOTIFY=WSER07
==> //JOBLIB DD DSN=WSER07.LINK.LOAD,DISP=SHR
==> //

Optional StarBat parameters:
Maximum input records ==>          (1 to 99999999)
Maximum output records ==>         (1 to 99999999)
Select every Nth      ==>          (1 to 99999999)

Optional Default StarWarp parameters:
Type of action        ==>           (ADD or SUB to increment; DATE to insert
Date adjustment       ==>           (number and code like 90D or actual date
Corrective action     ==>           (action to take, like FLAG)
Date validation       ==>           (validation check, like WKEND)
IGNORE Flags         ==>           (LIST of code characters)
Pivot Year            ==> 70        (0 to 99, for windowing two digit years)
Base Year             ==>           (1600 to 9999 base for two digit years)
Fiscal Month number  ==>           (1 to 12) 1=January, 12=December
Holiday exit name    ==>           (routine name for holiday date checking)
Exit library dsn     ==>

In foreground mode, ISPLLIB LIBDEF will be used to allocate exit library.
In background mode, a commented STEPLIB will be created. The user can
otherwise include the library as a JOBLIB in the JCL cards above.

```

**Figure 20. Sample STARWARP Data Set Prompt Panel**

## STARWARP User Guide

In this edit session, you may make any desired changes to the STARBAT JCL and control statements. Enter **SUB** as a primary command to submit the job.

```
EDIT          WSER11.SPFTEMP1.CNTL                      Columns 00001 00072
Command ==>  SUB                                       Scroll ==> CSR
***** ***** Top of Data *****
000001 //WSER07J JOB (X170,374),MYJOB,CLASS=A,
000002 //  MSGCLASS=X,TIME=(0,22),NOTIFY=WSER07
000003 //JOBLIB  DD DSN=WSER07.LINK.LOAD,DISP=SHR
000004 //* USING SKELETON PDS$WBAT
000005 //PDJ07241 EXEC PGM=STARBAT,DYNAMNBR=10,REGION=5M
000006 //*STEPLIB DD DISP=SHR,DSN=SYS1.STARTOOL.LOAD
000007 //DD01 DD DISP=SHR,DSN=WSER07.LIB.PDSE(STARBATT)
000008 //DD010 DD DISP=SHR,DSN=WSER07.LIB.CNTLZ
000009 //SYSPRINT DD SYSOUT=*
000010 //SYSLIST  DD SYSOUT=*
000011 //SYSIN DD *
000012 DD01 COPYSOME RDW=3,MAXDATERR=100,
000013 IF=(25,EQ,C'Las Vegas'),
000014     WARP=(44,C,CCYYMMDD,
000015     HOLIDAY=CALEXIT,
000016     ACTION=NEXTBDAY,
000017     ADD=4Y),
000018     ABEND=1
000019 /*
***** ***** Bottom of Data *****
```

**Figure 21. Sample Generated STARBAT Control Statements**

After you submit the job, you may press **END** or PF3 to return to the STARWARP dialog; then, an **END** or PF3 will terminate the STARWARP dialog.

At this point, the job to warp this data set has been completed and submitted. There are many other things that you can do in this dialog; we will cover other variations in a later section.

**Option 4B -- WARPSAVE: Create a Job from a Saved Table**

Select Option **4B** from the STARWARP Primary Options menu and press **ENTER** to create a job from a saved copybook table.

```

----- WARP table Prompt -----
OPTION  ===>

Enter operands for WARP:
WARP table name      ===> WPTST1  (3 to 6 character name)
Reset current table  ===> YES    (Yes/No to reset and replace current warp)
Source library       ===> PROFILE (Profile, Group, Sample or Other)
Other DSNAME        ===>
    
```

**Figure 22. Sample WARP Table Prompt**

This will activate a STARWARP copybook table as shown below. Note that the table name (WPTST1) is displayed in the copybook table header for identification.

```

----- Copybook for StarWarp WPTST1----- Row 1 to 25 of 28
COMMAND ===>                                SCROLL ===> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain
Help

CMD  MSG      NAME                                PICTURE COLUMN TYP  LENGTH
----
01  TESTBOOK                                1          80
02  MYFILE-SKIP                             X(1) 1     CHR   1
02  MYFILE-IDENTIFIER                       X(14) 2     CHR  14
02  MYFILE-KEY                               16         9
05  MYFILE-KEY-FIRST2                       X(2) 16     CHR   2
05  MYFILE-KEY-REST                          X(7) 18     CHR   7
02  MYFILE-PLACE-OF-BUSINESS                 X(19) 25     CHR  19
02  MYFILE-DATE-OF-LAST-CHANGE               X(8) 44     CHR   8
02  MYFILE-FIRST-BINARY                      4 BIN 52     BIN   4
02  FILLER                                   X 56       CHR   1
02  MYFILE-MISSION-KEYS                      57         4
05  MYFILE-KEY-ONE                           X(1) 57     CHR   1
05  MYFILE-KEY-TWO                           X(1) 58     CHR   1
05  MYFILE-KEY-THREE-FOUR                    X(1) 59     CHR   1
05  MYFILE-KEY-THREE-FOUR                    #2 60      CHR   1
02  FILLER                                   XX 61      CHR   2
    
```

**Figure 23. Sample Resulting STARWARP Table**

Note that you can also retrieve saved STARWARP tables from other sources: GROUP, SAMPLE or OTHER. By default, the table is retrieved from the PROFILE data set. For the OTHER data set, you may specify any existing partitioned data set name (with no member name) and STARWARP will allocate it and retrieve the saved table.

**Option 4C -- PROFMAN: Manipulate STARWARP Saved Tables**

Select Option **4C** from the STARWARP Primary Options menu and press ENTER to invoke the **PROFMAN** dialog; **PROFMAN** can be used to activate, rename or delete a STARWARP copybook table from a table display like the following example. You should enter an **S** line command as shown below to active a STARWARP copybook table.

```

----- PROFMAN Table List ----- Row 1 to 3 of 3
COMMAND ==>                               SCROLL ==> CSR
Enter an ISPF command or a special control code:
-----
  CMD FUNC NAME      DATA/MSG  Last Modified      SIZE First entry in table
  LC   IMPACT          1998/06/05 09.49      2 WSER11.SERNET.ASSEMBLE
s SW   WPTST1 *SELECT* 1998/06/06 08.10      28 This is a test
      SW   WPTST4          1998/06/08 07.56      4 This is test #4
      WO   IMPACT          1998/06/05 09.53      231 D 'WSER11.SER.LINKLIB'
  
```

**Figure 24. Sample PROFMAN Table**

The **PROFMAN** table is built from members in your profile data set; the table is sorted by function type (**CB** for copybook, **LC** for LISTC, **ML** for MEMLIST, **SW** for STARWARP, and **WO** for WORKPAD) and table name.

The following line commands are supported to manage the saved tables:

- DEL** Deletes a saved STARWARP copybook table.
- REN** Renames a saved STARWARP copybook table. Enter the new name in the DATA/MSG column.
- S** Activates a saved STARWARP copybook table.
- X** Removes a saved STARWARP copybook table from the display.

### Option 4D -- DATECALC: Calculate Dates

Select Option **4D** from the STARWARP Primary Options menu and press **ENTER** to utilize the date calculator. A start date can be entered in any supported character date format and the result is displayed in the middle of the panel as shown below.

In this example, we add four years to July 4, 1998 and move the resulting date to the next business day. Holiday exit PDS#HOLI is to be invoked to determine if a week day is a holiday.

```

----- Date Calculator -----
OPTION  ===>

Input dates:
Input date picture  ==> CCYY/MM/DD (descriptive picture, like CCYY-MM-DD)
Date value         ==> 1998/07/04 (start date in the above format)
Output date picture ==> CCYY-MM-DD (descriptive picture, like CCYY-MM-DD)

Resulting converted date =          2002-07-05  Friday
                                                                More:      +

Optional Warp parameters:
Type of action      ==> ADD          (ADD to increment or SUB to decrement)
Date adjustment     ==> 4Y          (number and code like 90D, 2W, 4M or 10Y)
Corrective action   ==> NEXTBDAY    (action to take, like FLAG)
Holiday exit name   ==> PDS#HOLI    (routine name for holiday date checking)
Exit library dsn    ==>

Notes:
For additional specific information on any of the above fields, place the
cursor in the field and press HELP for a detailed pop-up tutorial.

This date calculator will convert and display the date in any of the date
formats supported by StarWarp.

The following date pictures are supported:
YYDDD, CCYYDDD, DDDYY, DDDCCYY, YMMDD, CCYYMMDD, DMMYY, DMMCCYY,
MDDYY, MMDDCCYY, HYYDDD, HYYMMDD, CYMMDD, LLLLLL, COBLLL, 99MMDD,
CC99MMDD, 99DDD, CC99DDD, DYNDATEA, DYNDATEB, DDMONYY, DDMONCCYY,
YY/MM/DD, CCYY/MM/DD, DD/MM/YY, DD/MM/CCYY, MM/DD/YY, MM/DD/CCYY,
YY-MM-DD, CCYY-MM-DD, DD-MM-YY, DD-MM-CCYY, MM-DD-YY, MM-DD-CCYY,
YY.MM.DD, CCYY.MM.DD, DD.MM.YY, DD.MM.CCYY, MM.DD.YY, MM.DD.CCYY.

Enter date values in the same format as the corresponding date picture.
For example, with a date picture of CCYY-MM-DD, a date of 1999-12-31
is the last day of 1999.

```

**Figure 25. Sample Date Calculator Panel**

The date calculator is often used to test how sample dates will be aged in a data set or to check the results of a date calculation in another program.

### ***Option 4E -- EUROCALC: Monetary Conversion Calculator***

Select Option **4E** from the STARWARP Primary Options menu and press **ENTER** to utilize the Euro calculator. This calculator may be used to convert currencies between European member states; or it may be used for other numeric calculations which require addition, subtraction, multiplication or division operations. The result is displayed in the middle of the panel as shown below.

In this example, we add four years to July 4, 1998 and move the resulting date to the next business day. Holiday exit PDS#HOLI is to be invoked to determine if a week day is a holiday.

```

----- Euro Calculator -----
OPTION ==>

Calculator values:
Input euro picture   ==> 99999V99      (descriptive picture, like 99999V99)
Monetary value      ==> 01000.00      (money amount in the above format)
Output euro picture ==> 99999V99      (descriptive picture, like 99999V99)
Type of action      ==> CONV          (ADD, SUB, MULT, DIV or CONV)
Data adjustment     ==> DEMFRF       (number or CONV name)

Resulting converted value =          3353.86

Optional parameter:
Exit library name   ==>

Notes:
This calculator may be used to convert currencies between European member
states. In addition, it may be used for other numeric calculations which
require addition, subtraction, multiplication or division operations.

For additional specific information on any of the above fields, place the
cursor in the field and press HELP for a detailed pop-up tutorial.

The following Euro pictures are supported:
99, 999, ... up to fifteen 9 digits with an optional V decimal position
indicator. For example, 999V99 is a valid picture but 999.99 is not.

The Monetary value must be entered in the same format as the input euro
picture; however, you may enter a decimal point in the position indicated
by the V character of the input picture. For example, 123.45 is a valid
entry for a picture of 999V99; however 123V45 is not a valid monetary value

The Type of action should be ADD, SUB, MULT, DIV or CONV for the item
specified in the Data adjustment field.

The Data adjustment indicates the quantity for the Type of action field
as described below:
ADD - adds the value provided.
SUB - subtracts the value provided.
MULT - multiplies by the provided value and rounds the result.
      A multiplication is usually performed to convert currency in euros
      to member currency. For example, euros to French Francs.
DIV - divides by the provided value and rounds the result. A division
      is usually performed to convert member currency to euro units.
CONV - looks up the conversion indicated and performs a division and/or
      a multiplication and rounds the result.

The Exit library name may be entered if the PDS#CONV exit routine needs to be
accessed with an ISPF LIBDEF command. If the exit routine resides in a
linklist, JOBLIB or STEPLIB data set, you do not need to use a LIBDEF.

```

**Figure 26. Sample Euro Conversion Calculator**

Note that you can also retrieve saved STARWARP tables from other sources: GROUP, SAMPLE or OTHER. By default, the table is retrieved from the PROFILE data set. For the OTHER data set, you may specify any existing partitioned data set name (with no member name) and STARWARP will allocate it and retrieve the saved table.

## STARWARP User Guide

For a conversion like Deutsche Marks to French Francs, you would enter CONV=DEMFRF as a token. The exit will first look up DEMEUR and then EURFRF to return: DIV=1.92573,MULT=6.45863. 100 Deutsche Marks would convert to 335.39 French Francs after rounding

This type of lookup requires six character names made up of two three character pieces which use the defined euro names shown below.

<b>BEF</b>	Belgian Franc (smallest currency unit is 1.00)
<b>LUF</b>	Luxembourg Franc (smallest currency unit is 1.00)
<b>DKK</b>	Danish Krone (choose not to join initially)
<b>DEM</b>	Deutsche Mark
<b>GRD</b>	Greek Drachma (not actually eligible)
<b>ESP</b>	Spanish Peseta (smallest currency unit is 1.00)
<b>FRF</b>	French Franc
<b>IEP</b>	Irish Punt
<b>ITL</b>	Italian Lira (smallest currency unit is 1.00)
<b>NLG</b>	Netherlands Guilder
<b>ATS</b>	Austrian Schilling
<b>PTE</b>	Portuguese Escudo (smallest currency unit is 0.10)
<b>FIM</b>	Finish Markka
<b>SEK</b>	Swedish Krona (choose not to join initially)
<b>GBP</b>	United Kingdom Pound Sterling (choose not to join initially)

### Option 4F -- DATEDIFF: Date Difference Calculator

Select Option **4F** from the STARWARP Primary Options menu and press **ENTER** to utilize the date difference calculator. A start date and end date can be entered in any supported character date formats and the result in days is displayed in the middle of the panel as shown below.

In this example, we wish to determine the number of days from June 1, 1998 to January 1, 2000.

```

----- Date Difference Calculator -----
OPTION  ===>

Input dates:
First date picture  ==> CCYY/MM/DD (descriptive picture, like CCYY-MM-DD)
First date value   ==> 1998/06/01 (first date in the above format)
Second date picture ==> CCYY.DDD  (descriptive picture, like CCYY-MM-DD)
Second date value  ==> 2000.001  (second date in the above format)

Resulting difference in Days =      579

Notes:
For additional specific information on any of the above fields, place the
cursor in the field and press HELP for a detailed pop-up tutorial.

This date calculator can display the difference between any two dates in
any of the date formats supported by StarWarp.

The following date pictures are supported:
YYDDD, CCYYDDD, DDDYY, DDDCCYY, YYMMDD, CCYYMMDD, DDMYY, DDMCCYY,
MMDDYY, MMDDCCYY, HYDDD, HYMMDD, CYMMDD, LLLLLL, COBLLL, 99MMDD,
CC99MMDD, 99DDD, CC99DDD, DYNDATEA, DYNDATEB, DDMONYY, DDMONCCYY,
YY/MM/DD, CCYY/MM/DD, DD/MM/YY, DD/MM/CCYY, MM/DD/YY, MM/DD/CCYY,
YY-MM-DD, CCYY-MM-DD, DD-MM-YY, DD-MM-CCYY, MM-DD-YY, MM-DD-CCYY,
YY.MM.DD, CCYY.MM.DD, DD.MM.YY, DD.MM.CCYY, MM.DD.YY, MM.DD.CCYY.

Enter date values in the same format as the corresponding date picture.
For example, with a date picture of CCYY-MM-DD, a date of 1999-12-31
is the last day of 1999.

```

**Figure 27. Sample Date Difference Calculator**

### Option 4G -- HOLIDAYC: Create a Holiday Calendar

Select Option **4G** from the STARWARP Primary Options menu and press **ENTER** to build a holiday calendar for your application or your site. This will prompt for a data set containing holiday dates in CCYYMMDD format and build a holiday table that can be utilized by STARWARP. The input data set is compatible with TransCentury calendar inputs.

```

----- Holiday exit build Prompt -----
COMMAND ===>

Input data set      ==> PDSE610.CNTL(HOLIDATA)
The input data set must be sequential or a PDS with member included
Output data set     ==> LINK.LOAD(PDS#HOLX)
The output data set must be a load library PDS with member included

Specify JOB and JOBLIB statements for the batch job to be created:

===> //WSER07J JOB (X170,374),MYJOB,CLASS=A,
===> //  MSGCLASS=X,TIME=(0,22),NOTIFY=WSER07
===> //JOBLIB   DD DSN=WSER07.LINK.LOAD,DISP=SHR
===> //
    
```

Figure 28. Sample HOLIDAYC Prompt

The following member is the input dates in CCYYMMDD format. Note that the first **TBL-ID** entry is not required by STARWARP.

```

EDIT          WSER07.PDSE610.CNTL(HOLIDATA)          Columns 00001 00072
Command ===>                                         Scroll ===> CSR
***** ***** Top of Data *****
000001 TBL-ID=01
000002 19990101
000003 19990115
000004 19990219
000005 19990528
000006 19990704
000007 19990903
000008 19991008
000009 19991111
000010 19991112
000011 19991122
000012 19991225
000013 20000101
    
```

Figure 29. Sample HOLIDAYC Inputs

This process will build JCL and source statements as shown below. You may **SUBMIT** this job to build the holiday table or place the member in a data set for use later.

```

EDIT          WSER07.SPFTEMP1.CNTL          Columns 00001 00072
Command ==> submit                        Scroll ==> CSR
***** ***** Top of Data *****
000001 //WSER07J JOB (X170,374),MYJOB,CLASS=A,
000002 //  MSGCLASS=X,TIME=(0,22),NOTIFY=WSER07
000003 //JOBLIB DD DSN=WSER07.LINK.LOAD,DISP=SHR
000004 /** USING SKELETONS PDS$HOLC, PDS$HOL2, PDS$HOL3
000005 //PDS#HOLI EXEC PGM=ASMA90,REGION=4000K,COND=(5,LT),
000006 // PARM=(DECK,NOOBJECT,NORLD,TERM,NOXREF)
000007 //SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(10,5))
000008 //SYSPRINT DD SYSOUT=*
000009 //SYSTEM DD SYSOUT=*
000010 //SYSPUNCH DD UNIT=SYSDA,SPACE=(TRK,(5,5)),
000011 // DISP=(MOD,PASS),DSN=&X,DCB=BLKSIZE=3120
000012 //SYSLIB DD DSN=SYS1.MACLIB,DISP=SHR,DCB=BLKSIZE=32000
000013 //SYSIN DD *
000014 PDS#HOLI TITLE 'STARTOOL HOLIDAY EXIT INTERFACE -- SAMPLE CALL EXIT'
000015 ***
000016 *** NOTE: THIS IS AN SAMPLE EXIT ROUTINE TO SUPPORT HOLIDAYS
000017 ***
000018 PDS#HOLI CSECT

```

**Figure 30. Sample Euro Conversion Calculator**

### Option 4H -- IMPACT: Load to Source Reconciliation

Load option **4H** or **IMPACT** identifies CSECTS within load members and matches these CSECT names to source library member names (and program names within members) to locate matching source and load members.

**IMPACT** performs the following steps:

1. **IMPACT** prompts for any number of source and load libraries and builds a customized data set table.
2. **IMPACT** prompts for member groups to analyze for each of the following member types: load library members, assembler members, COBOL members and PL/I members.
3. **IMPACT** analyzes and extracts the CSECT names and member names (from STARWARP HISTORY GENERATE subcommand outputs) and builds a list of members to find.
4. **IMPACT** searches the source data sets with a GLOBAL FIND command for each source language type (Assembler, COBOL and PL/I) for program names, matches CSECT names to these program names and displays intermediate search results.
5. At the conclusion of the above searches, **IMPACT** places you into a **WORKPAD** which has an entry for each member name, program name or CSECT, date if available, and the corresponding data set name. Note that comments will be added to **WORKPAD** to indicate potential missing source or missing load members.

The following panel is used to build a customized data set list which will contain all source and load data sets that are to be searched in the **IMPACT** matching process. From this data set prompt panel, you may enter up to 10 data set names or data set masks, press **ENTER** and continue entering data set names or data set masks.

```

----- IMPACT: Impact analysis--locate source for load -----
OPTION  ==>

Enter one or more data set names or data set masks.
Note that names must be fully qualified with no quotes.

Press ENTER to add entries to the LISTC table or
press END (PFK 3) when you have specified all source and load data sets.

==> wser07.test.sourc*
==> wser07.test.load*
==>
==>
==>
==>
==>
==>
==>
==>

The Data set mask field should be entered as a partial data set name and
each level may contain an asterisk before or after any characters in a level.
For example, the following specifications would be valid:
  SYS1.THE.DATASET.NAME
  HABL.LI*.*.*NTL
    
```

Figure 31. IMPACT Data Sets Prompt Panel

After pressing **ENTER** on the above prompting panel, you will see a transient panel like the following one which displays the table of source and load data sets as they are being built. Control will return to the data set prompt panel; however, so that you can enter additional data set names.

```

----- List files 0          - (Attributes) ----- Row 1 to 4 of 4
COMMAND ==>
- DSN=WSER07.LIB.CLIST,VOL=SER=SER002 MEM=LANGTYP -----
CMD  C V DATA/MSG -----DATA SET NAME ----- VOLUME DO RECFM LRECL BLKSI
     Y Y          WSER07.TEST.LOAD1_____ SHARE1 PO U          6144 19069
     Y Y          WSER07.TEST.LOAD2_____ SHARE1 PO U          6144 19069
     Y Y          WSER07.TEST.SOURCE1_____ SHARE1 PO FB           80  9040
     Y Y          WSER07.TEST.SOURCE2_____ SHARE1 PO FB           80  9040
***** Bottom of data *****

```

**Figure 32. IMPACT Data Sets Transient Panel**

You may continue entering as many data set names or masks as you want. To terminate this prompting sequence, respond with an **END**.

```

----- IMPACT: Impact analysis--locate source for load -----
OPTION  ==> end

Enter one or more data set names or data set masks.
Note that names must be fully qualified with no quotes.

Press ENTER to add entries to the LISTC table or
press END (PFK 3) when you have specified all source and load data sets.

==>
==>
==>
==>
==>
==>
==>
==>
==>
==>

The Data set mask field should be entered as a partial data set name and
each level may contain an asterisk before or after any characters in a level.
For example, the following specifications would be valid:
SYS1.THE.DATASET.NAME
HABL.LI*.*.*NTL

```

**Figure 33. IMPACT Data Sets Prompt Panel (Again)**

In this panel, you may enter the member or member groups which are to be analyzed. Normally, you will enter a member group for the load libraries and at least one member group for Assembler, COBOL and PL/I source members. In this case, all members for each member type are to be analyzed.

Note that this process can execute its analysis phase in the background. If you run IMPACT in the background be sure to execute CLIST IMPACTXR to copy the results back into your edit profile after the batch job completes.

```

----- IMPACT: Impact analysis--locate source for load -----
OPTION ==>

- DSN=WSER07.LIB.CLIST,VOL=SER=SER002 MEM=LANGTYP -----
  Member or member group for load libraries ==> :
  Member or member group for ASM libraries ==> :
  Member or member group for COB libraries ==> :
  Member or member group for PL/I libraries ==> :
  Process in foreground ==> YES (No for background)
  Note: If a language type search is not required, leave the prompt field blank.

After processing in the background, use CLIST IMPACTXR to retrieve results.

You need to enter a member name or member group name to specify which
members are to be processed. The following are example names:
  abcde - for the single member ABCDE
  : - for all members in the data set
  aa* - for all members beginning with AA (* denotes a generic)
  (a,b,c2) - for members A, B and C2 (parentheses enclose a member list)
  (acc*,inv*,xyz*) - List of generic names enclosed in parentheses

Field level help from the member group input field will provide additional
samples (place the cursor on the input field and press PF1).
  
```

Figure 34. IMPACT Member Groups Prompt

This intermediate results panel is produced after load members are analyzed. Press **ENTER** on each of the intermediate results panels when you are ready to proceed to the next IMPACT segment.

```

----- List files IMPACT - (Attributes) ----- Row 1 to 4 of 4
COMMAND ==>
- DSN=WSER07.LIB.CLIST,VOL=SER=SER002 MEM=: -----
CMD C V DATA/MSG ----- +----- Search Data Set Results -----+
- Y *HISTORY WSER07. |
- Y *HISTORY WSER07. | OPTION =>
- Y *BYPASS* WSER07. |
- Y *BYPASS* WSER07. |
***** |
| Number of data sets searched: 2
| Number of data sets with matches: 2
| Total number of members: 13
|
| Press ENTER to view the log
| Press END to display data set table
| Press RCHANGE (PF6/18) to remove data
| sets bypassed or with no matches
+-----+
  
```

Figure 35. IMPACT Load Analysis Panel

This optional intermediate results panel is produced if Assembler source members are analyzed. In this case, three Assembler routines were identified in one data set.

```

----- List files IMPACT - (Attributes) ----- Row 1 to 4 of 4
COMMAND ==>                                     SCROLL ==> CSR
- DSN=WSER07.LIB.CLIST,VOL=SER=SER002 MEM=: -----
CMD  C V DATA/MSG -----+----- Search Data Set Results -----+
      - Y *BYPASS* WSER07.
      - Y *BYPASS* WSER07.      OPTION =>
      - Y *NOMATCH WSER07.
      - Y *FIND   WSER07.      Number of data sets searched:      2
*****
      Number of data sets with matches:      1
      Total number of members:      3

      Press ENTER to view the log
      Press END to display data set table
      Press RCHANGE (PF6/18) to remove data
      sets bypassed or with no matches
-----+-----
  
```

Figure 36. IMPACT Assembler Analysis Panel

This optional intermediate results panel is produced if COBOL source members are analyzed. In this case, ten COBOL members were identified in two data sets.

```

----- List files IMPACT - (Attributes) ----- Row 1 to 4 of 4
COMMAND ==>                                     SCROLL ==> CSR
- DSN=WSER07.LIB.CLIST,VOL=SER=SER002 MEM=: -----
CMD  C V DATA/MSG -----+----- Search Data Set Results -----+
      - Y *BYPASS* WSER07.
      - Y *BYPASS* WSER07.      OPTION =>
      - Y *FIND*   WSER07.
      - Y *FIND*   WSER07.      Number of data sets searched:      2
*****
      Number of data sets with matches:      2
      Total number of members:      10

      Press ENTER to view the log
      Press END to display data set table
      Press RCHANGE (PF6/18) to remove data
      sets bypassed or with no matches
-----+-----
  
```

Figure 37. IMPACT COBOL Analysis Panel

Another similar optional intermediate results panel is produced if PL/I source members are analyzed. In this case, a single PL/I member was identified in one of the source libraries.

After pressing ENTER on each of the above intermediate results panels, you will be placed into a WORKPAD called IMPACT. This panel displays the final results with all source and load matches and you may enter line commands such as ED to edit any source member.

## STARWARP User Guide

The IMPACT WORKPAD is sorted in PGM NAME order. Also, several other sorts are defined so that you may view this information in other ways.

Following are some notes on interpreting the following WORKPAD.

1. Load member TBLHNDLR contains a CSECT called ALLOCMEM in data set WSER07.TEST.LOAD1.
2. Source member MEMALLOC contains a program called ALLOCMEM corresponding to this member.
3. Source member BREAKOUT contains a program called BREAKDSN with no corresponding load member.
4. Source member CNV2GREG is at an earlier level than load member CNV2GREG.
5. Source member CONVDATE does not have ISPF statistics; it may be at the level of the two load members.
6. Source member DATECONV is at a later level than load member DATECONV; however, source member DATECON2 was modified on the same date that this load member was created.
7. Load member WATDSN contains a CSECT called DSNTAB which has no corresponding source member.

```

----- StarTool WORKPAD Table IMPACT ----- Row 1 to 32 of 44
COMMAND ==>                                     SCROLL ==> CSR
Enter an ISPF command or a special control code:
-----
OPT  MSG/RC  TYP MEMBER  PGM NAME  -- DATE --  ----- DSNAME/COMMAND -----
ed  D TBLHNDLR ALLOCMEM 1994/04/06 'WSER07.TEST.LOAD1'
    D MEMALLOC ALLOCMEM 1994/04/06 'WSER07.TEST.SOURCE2'
    D BREAKOUT BREAKDSN 1996/01/07 'WSER07.TEST.SOURCE2'
    * *          BREAKDSN          No matching load
    D CNV2GREG CNV2GREG 1992/12/10 'WSER07.TEST.LOAD1'
    D CNV2GREG CNV2GREG 1990/05/24 'WSER07.TEST.SOURCE1'
    D DATECONV CONVDATE 1987/12/07 'WSER07.TEST.LOAD1'
    D GETDSCB1 CONVDATE 1987/12/07 'WSER07.TEST.LOAD1'
    D CONVDATE CONVDATE          'WSER07.TEST.SOURCE2'
    D DATECONV DATECONV 1991/12/22 'WSER07.TEST.LOAD1'
    D DATECON2 DATECONV 1991/12/22 'WSER07.TEST.SOURCE1'
    D DATECONV DATECONV 1996/08/14 'WSER07.TEST.SOURCE1'
    D WATDSN   DSNTAB   1991/07/01 'WSER07.TEST.LOAD1'
    * *          DSNTAB          No matching source
    D GETDSCB1 GETDSCB1 1992/07/27 'WSER07.TEST.LOAD1'
    D GETDSCB1 GETDSCB1 1992/07/27 'WSER07.TEST.SOURCE1'
    D DATECONV IKJUPDT 1994/06/25 'WSER07.TEST.LOAD1'
    * *          IKJUPDT          No matching source
    D LOAEXTR  LOAEXTR  1990/02/01 'WSER07.TEST.LOAD1'
    D LOAEXTR  LOAEXTR  1990/02/01 'WSER07.TEST.SOURCE1'
    D LOADPRNT LOADPRNT 1990/01/07 'WSER07.TEST.LOAD1'
    D LOADPRNT LOADPRNT 1990/01/07 'WSER07.TEST.SOURCE1'
    D VSAMMAPP O929B15 1996/07/07 'WSER07.TEST.LOAD2'
    D CBT1269  O929B15 1996/07/07 'WSER07.TEST.LOAD2'
    D VSAMMAPP O929B15 1996/07/07 'WSER07.TEST.SOURCE2'
    D GETDSCB1 PASSDSCB 1990/11/21 'WSER07.TEST.LOAD1'
    * *          PASSDSCB        No matching source
    D DATECONV PCL      1991/12/22 'WSER07.TEST.LOAD1'
    * *          PCL             No matching source
    D TBLHNDLR SCAN4HIT 1994/04/06 'WSER07.TEST.LOAD1'
    * *          SCAN4HIT        No matching source
    D DATECONV SERLCONV 1991/06/01 'WSER07.TEST.LOAD1'
    D SERLCONV SERLCONV 1991/06/01 'WSER07.TEST.LOAD1'
    D SERLCONV SERLCONV 1991/06/01 'WSER07.TEST.SOURCE1'
    D SSR01AR  SSR01AR  1996/06/09 'WSER07.TEST.LOAD1'
    D SSR01AR  SSR01AR  'WSER07.TEST.SOURCE1'

```

Figure 38. IMPACT WORKPAD Panel

The following edit session is the result of entering an **ED** line command in the WORKPAD above.

```

File  Edit  Confirm  Menu  Utilities  Compilers  Test  Help
-----
EDIT      WSER07.TEST.SOURCE2(BREAKOUT) - 01.00          Columns 00001 00072
Command ==>                                          Scroll ==> CSR
***** ***** Top of Data *****
000001 000010*****
000002 000100 ID DIVISION.
000003 000110*****
000004 000200 PROGRAM-ID.          BREAKDSN.
000005 000300 AUTHOR.            GARY DUFFIELD.
000006 000310 REMARKS.
000007 000320          THIS PROGRAM EDITS ANY DATASET NAME PASSED TO IT
000008 000330          AND SPLITS OUT THE PDS MEMBER NAME IF IT EXISTS.
000009 000340
000010 000350*****
000011 000400 ENVIRONMENT DIVISION.
000012 000410*****
000013 000500 CONFIGURATION SECTION.
000014 000600 SOURCE-COMPUTER.  IBM-370.
000015 000700 OBJECT-COMPUTER. IBM-370.
000016 000800
000017 000810*****
000018 000900 DATA DIVISION.
000019 000910*****
000020 001100 WORKING-STORAGE SECTION.

```

**Figure 39. IMPACT Result of ED Line Command**

### Option 4I -- COMPCHK: Load Library Summary

Option 4I or COMPCHK routine was designed to summarize the contents of a load library. It provides four different load library analysis reports which could be used to evaluate selected members or an entire load library.

COMPCHK operates with the STARWARP log. COMPCHK first clears the log and places the output of a STARWARP HISTORY subcommand for the specified members into the log. Then, COMPCHK filters out IBM compiler run-time routines, sorts the data and places the results of its analysis into the log after the end of the HISTORY output.

The load library analysis reports are as follows:

- Compiler summary** shows the compilers (with the translator identifier) and versions used in compiling the specified members. Note that a member can be counted for several different compilers or compiler levels if it is made up of multiple separately compiled CSECTS.
- AMASPZAP summary** is produced if any CSECTS analyzed have zaps applied.
- USERDATA summary** is produced if any CSECTS analyzed contain user IDR data (USERDATA information is normally used by IBM and other software vendors to document internal changes).
- Linkage edit summary** provides a summary of linkage edit dates. With this information, you can easily get an idea of the relative age of the members in a data set.

The first panel presented permits you to specify the load library and members COMPCHK is to analyze.

```

----- Compiler analysis -----
OPTION  ==>

- DSN=WSER07.LIB.FILE112,VOL=SER=SER002 MEM=(DSAT -----
  Enter Load DSNAME      ==> LINK.LOAD
  Member or member group ==> (DSAT,PDSE5*)
  Main members only      ==> YES          (Yes to perform SUBLIST ONLYMAIN or
                                          No to process all members in the
                                          member group including alias members)

You need to enter a member name or member group name to specify which
members are to be processed. The following are example names:
  abcde   - for the single member ABCDE
  :       - for all members in the data set
  aa*     - for all members beginning with AA (* denotes a generic)
  (a,b,c2) - for members A, B and C2 (parentheses enclose a member list)
  (acc*,inv*,xyz*) - List of generic names enclosed in parentheses

Field level help from the member group input field will give
additional samples (put cursor on the input field and press PF1)

```

Figure 40. COMPCHK Prompting Panel

Results are displayed in the log as shown below. Note: all of the analyzed output from the HISTORY subcommand is in the log just above start of these results.

```

----- ISPMODE Session# 1 Log# Row 1,011 to 1,037 of 1,062
COMMAND ==>
                                           SCROLL ==> CSR
- DSN=WSER07.LINK.LOAD,VOL=SER=SER002 MEM=(DSAT -----
>----->Analysis results

Compiler summary:
Compiler  Trans-id      Vnn Mnn  members
-----
ASMA90    569623400   V01 M01    205
ASMA90    569623400   V01 M02    18
ASMH      566896201   V02 M01    22
COBOL370  566895807   V11 M00    31
PL1       5734-PL1    V03 M00     7

AMASPZAP summary:
Zapname   members
-----
P440END   12
P510AFT   4
P510COK   1

USERDATA summary:
User data name                               members
-----
STARWARP, VERSION 5.2.0, 1997.084           1
UZ23355                                       14
UZ25385                                       3

Linkage edit summary:
Linkdate   members
-----
1994/03/17    22
1995/12/19   101
1997/04/03    2
    
```

Figure 41. COMPCHK Results

### Option 4J -- CSECTCHK: Load Library Component Check

Option **4J** or **CSECTCHK** identifies inconsistencies in CSECTS (differing compilers, link dates, USERDATA, ZAP or translator level) within multiple load members. CSECTCHK can be used to evaluate selected members of a load library or an entire load library.

**CSECTCHK** assumes that if a CSECT appears in a load member, it should be at the same level for all load members checked. Note: if the checked members represent multiple release levels of the same source member or if the same CSECT name is used in multiple load members, the CSECTCHK routine will report many lines of potential problems.

CSECTCHK operates with the STARWARP log. CSECTCHK first clears the log and places the output of a STARWARP HISTORY subcommand for the specified members into the log. Then, CSECTCHK filters out IBM compiler run-time routines, sorts the data and places the results of its analysis into the log after the end of the HISTORY output.

The internal consistency report shows several types of potential problems:

- Compilation date differs** in the following example, a CSECT named DVOL in members DVOL and DVOL3 were compiled or assembled on different dates.
- Zap date differs** in the following example, a CSECT named DVOL in members DVOL and DVOL3 have different AMASPZAP dates.
- Translator or level differs** in the following example, a CSECT named DVOLPCL was compiled or assembled with a different compiler or a different level of the same compiler.
- User-supplied data differs** in the following example, a CSECT named IXTOFMT5 in MEMBERS DUSE and XREFA has different user IDR information; IBM and other software vendors utilize IDR information to document internal module changes.

Internal consistency example:

CSECT	MEMBER1	MEMBER2	Problem
DVOL	DVOL	DVOL3	Compilation date differs
DVOL	DVOL	DVOL3	Zap date differs
DVOLPCL	DVOL	DVOL3	Translator or level differs
IXTOFMT5	DUSE	XREFA	User-supplied data differs

The first panel presented permits you to specify the load library and members CSECTCHK is to analyze.

```

----- CSECT consistency analysis -----
OPTION  ==>

- DSN=WSER07.LIB.FILE112,VOL=SER=SER002 MEM=(DSAT -----
  Enter Load DSNAME      ==> LINK.LOAD
  Member or member group ==> (DSAT,PDSE5*)
  Main members only      ==> YES          (Yes to perform SUBLIST ONLYMAIN or
                                          No to process all members in the
                                          member group including alias members)
  
```

Figure 42. CSECTCHK Prompting Panel

Results are displayed in the log as shown below. Note: all of the analyzed output from the HISTORY subcommand is in the log just above start of these results.

```

----- ISPMODE Session# 1 Log# Row 1,001 to 1,027 of 2,826
COMMAND ==>
- DSN=WSER07.LINK.LOAD,VOL=SER=SER002 MEM=(DSAT -----
>----->Analysis results
Internal consistency problems:
CSECT      MEMBER1    MEMBER2    Problem
-----
COMPARED   PDSE510    PDSE510B   Compilation date differs
COMPARED   PDSE510    PDSE510D   Compilation date differs
COMPARED   PDSE510    PDSE520    Compilation date differs
PDSMAIN    PDSE510D   PDSE520A   Zap date differs
PDSMAIN    PDSE510D   PDSE520A   User-supplied data differs
PDSMAIN    PDSE510D   PDSE520B   Compilation date differs
PDSMAIN    PDSE510D   PDSE520B   Zap date differs
PDSMAIN    PDSE510D   PDSE520B   User-supplied data differs
PDSMAIN    PDSE520    PDSE520A   Compilation date differs
PDSVERIF   PDSE520    PDSE520B   Compilation date differs
PDSVTOCR   PDSE510    PDSE510B   TRANSLATOR or LEVEL differs
PDSVTOCR   PDSE510    PDSE510B   Compilation date differs

```

Figure 43. CSECTCHK Results

### Option 4K -- LANGTYPE: Classify Source Modules

Option 4K or LANGTYPE may be used to categorize members in a source library. LANGTYPE can be used to evaluate selected members of a load library or an entire load library.

LANGTYPE operates with the STARWARP log. LANGTYPE first clears the log and places the output of a STARWARP LIST subcommand for the specified members and the specified number of lines into the log.

Then, LANGTYPE searches the log, categorizes these members based on their syntax, and produces a summary report showing the number of members associated with each language type and a detail report showing which members are coded in each language. Finally, LANGTYPE builds a member list and marks the language type of each member if the corresponding option is requested.

```

----- Language Type Analysis -----
OPTION ==>

- DSN=WSER07.LIB.CNTL,VOL=SER=SER001 MEM=$$$$CUST -----
  Enter source DSNAME    ==> pdse521.tsrc
  Member or member group ==> :
  MEMLIST already built ==> NO   (Yes or No; YES to ignore the above entries)
  Build MEMLIST and Flag ==> YES  (Yes or No; YES to mark MEMLIST entries)
  Search limit          ==> 15   (1 to 999; for maximum records to search)

You need to enter a member name or member group name to specify which
members are to be processed. The following are example names:
  abcde    - for the single member ABCDE
  :        - for all members in the data set
  aa*      - for all members beginning with AA (* denotes a generic)
  (a,b,c2) - for members A, B and C2 (parentheses enclose a member list)
  (acc*,inv*,xyz*) - List of generic names enclosed in parentheses

Field level help from the member group input field will give
additional samples (put cursor on the input field and press PF1)

```

Figure 44. LANGTYPE Prompt

If you already have a MEMLIST built, you should reply YES to "MEMLIST already built" and STARWARP will utilize the current data set and the current member list that is already built. Otherwise, the name of the data set should be entered in the "Enter source DSNAME" field and the member group should be entered in the "Member or member group" field to specify the members to be analyzed.

If you would like to build a MEMLIST and have the members categorized (or flagged) in the MEMLIST, you should mark the "Build MEMLIST and flag" field as YES.

The "Search limit" field determines the number of lines that are to be listed and analyzed for each member. In some cases, you can increase the accuracy of the LANGTYPE language determination if you increase this number.

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
EDIT          ----->Analysis results          Columns 00001 00072
Command ==>   Scroll ==> CSR
***** ***** Top of Data *****
000001 Member summary:
000002
000003 Language          Total
000004 -----
000005 Assembler          8
000006 CLIST              0
000007 COBOL             10
000008 EDITMAC           0
000009 JCL                4
000010 PL/I              0
000011 REXX              0
000012 PANEL             0
000013 MSGS              0
000014 SKEL              0
000015 TSOHELP          13
000016 COPYBOOK         12
000017 unknown          6
000018 -----
000019 Total members      53
000020
000021
000022 Member details:
000023
000024 Language          Members
000025 -----
000026 Assembler          FSHELP, REV, REVIEW, TESTS102, TESTS103, TESTS202,
000027                   TESTS203, WARPDATA
000028 COBOL              TESTS101, TESTS104, TESTS105, TESTS106, TESTS107,
000029                   TESTS108, TESTS109, TESTS110, TESTS111, TESTS201
000030 JCL                TAPEMAP, WARPJCL, WPJCLA, WPJCLVS
000031 TSOHELP            BLK3350, BLK3380, BLK3390, BLK9345, COMPARE$, DSAT,
000032                   DVOL, HEL, L, LIST, PRINTOFF, VTOC, ZAP
000033 COPYBOOK           CBDEMO, CBDEMO@, COPY00, COPY01, COPY02, COPY03, COPY04,
000034                   COPY05, COPY06, COPY07, WARPCOPY, WPCPYBK
000035 unknown            $$$DOCT, #2WARP1, DEMOVSA@, DEMOVSA, DEM2VSAM, TESTS204
***** ***** Bottom of Data *****

```

Figure 45. LANGTYPE Reports Sample

After pressing ENTER from the LANGTYPE prompt panel, you are placed in edit on a summary report showing the number of members associated with each language type and a detail report showing which members are coded in each language. Press END or PF3 to exit from edit and continue with LANGTYPE.

```

----- Source MEMLIST 1, Session# 1 ----- Row 1 to 26 of 53
COMMAND ==> remove *tsohelp                                SCROLL ==> CSR
Enter an ISPF command, a StarTool subcommand or a special control code:
- DSN=WSER07.PDSE521.TSRC,VOL=SER=SER001 MEM=: -----
CMD  NAME      DATA/MSG VER.MOD  CREATED      LAST  MODIFIED  SIZE  INIT  ID
$$$DOCT *UNKNOWN  01.06  1996/01/26  1997/11/07  7:56    37    23  WSER07
#2WARP1 *UNKNOWN
BLK3350 *TSOHELP
BLK3380 *TSOHELP
BLK3390 *TSOHELP
BLK9345 *TSOHELP
CBDEMO  *COPYBOO
CBDEMO@ *COPYBOO  01.25  1995/05/24  1997/05/04  8:13    56    30  WSER07
COMPARE$ *TSOHELP  01.00  1992/12/09  1992/12/09  5:28   137   137  SER07
COPY00  *COPYBOO
DEMOVSA@ *UNKNOWN  01.16  1994/06/29  1995/08/17  8:30    63    23  WSER07
DEMOVSAM *UNKNOWN
DEM2VSAM *UNKNOWN  01.03  1994/06/29  1994/07/01  8:02    17    23  WSER07
DSAT    *TSOHELP  01.27  1994/08/05  1994/08/05  9:34   243   243  WSER07
DVOL    *TSOHELP
FSHELP  *ASSEMBL  26.00  1986/08/06  1996/02/15 15:20   863   335  T$QGP75
HEL     *TSOHELP  25.00  1986/08/06  1995/07/27 10:40   860   335  T$QGP75
L       *TSOHELP  ALIAS
LIST    *TSOHELP
PRINTOFF *TSOHELP  01.00  1993/03/17  1993/03/17  8:52   125   125  SER07
REVIEW  *ASSEMBL  26.00  1984/08/24  1996/02/15 15:25  1354   76  T$QGP75
TAPEMAP *JCL      01.00  1994/08/05  1994/08/05  8:38    20    20  WSER07
TESTS101 *COBOL   01.00  1980/01/07  1980/05/24 12:00   304   304  FILE270
TESTS102 *ASSEMBL  01.00  1980/01/07  1986/08/14 12:00   438   438  FILE270

```

Figure 46. LANGTYPE MEMLIST

Finally, LANGTYPE builds a member list and marks the language type of each member if the corresponding option is requested. The language type is marked in the DATA/MSG field. Note that these marks are comments and they will be cleared for most line commands issued against these members.

One way to use the results of a LANGTYPE session is to X out members that are not of interest or use the REMOVE command to drop all members of a certain type as shown in the above example.

```

PDSPN05 ----- Source MEMLIST 1, Session# 1 ----- Row 1 to 26 of 40
COMMAND ==>
Enter an ISPF command, a StarTool subcommand or a special control code:
- DSN=WSER07.PDSE521.TSRC,VOL=SER=SER001 MEM=: -----
CMD  NAME      DATA/MSG  VER.MOD  CREATED      LAST  MODIFIED    SIZE  INIT  ID
    $$$DOCT  *UNKNOWN  01.06   1996/01/26   1997/11/07  7:56     37    23  WSER07
    #2WARP1  *UNKNOWN
    CBDEMO   *COPYBOO
    CBDEMO@  *COPYBOO  01.25   1995/05/24   1997/05/04  8:13     56    30  WSER07
    COPY00   *COPYBOO
    DEMOVSA@ *UNKNOWN  01.16   1994/06/29   1995/08/17  8:30     63    23  WSER07
    DEMOVSAM *UNKNOWN
    DEM2VSAM *UNKNOWN  01.03   1994/06/29   1994/07/01  8:02     17    23  WSER07
    FSHELP   *ASSEMBL  26.00   1986/08/06   1996/02/15  15:20    863   335  T$QGP75
    REVIEW   *ASSEMBL  26.00   1984/08/24   1996/02/15  15:25   1354   76  T$QGP75
    TAPEMAP  *JCL      01.00   1994/08/05   1994/08/05  8:38     20    20  WSER07
    TESTS101 *COBOL    01.00   1980/01/07   1980/05/24  12:00    304   304  FILE270
    TESTS102 *ASSEMBL  01.00   1980/01/07   1986/08/14  12:00    438   438  FILE270

```

**Figure 47. LANGTYPE MEMLIST After REMOVE**

The above MEMLIST shows the members that remain after a REMOVE command.



## Advanced Topics

---

### *Warp Commands*

The STARWARP function supports primary commands and line commands. Primary commands are entered on the top line of the table display and they usually affect the entire table

The following primary commands are currently supported in the warp table:

<b>BUILD</b>	initiates the skeleton process to build STARBAT control statements
<b>F</b>	finds an element in the warp table by partial name and positions the table to that element
<b>HELP</b>	invokes the STARWARP tutorial for the warp table.
<b>OUTPUT</b>	outputs the warp table using the current display format
<b>REFRESH</b>	drops all warp table lines except control statements and rebuilds the table after prompting
<b>SAVE</b>	saves the warp table in the profile data set, group data set or OTHER data set
<b>SHRINK</b>	reduces the size of the warp table to only control statements just before a <b>BUILD</b>

Line commands are entered on individual lines of the warp table and they generally only affect that table line. The doubled letter line commands are block line commands and they require a matching block line command.

The following line commands are currently supported:

<b>=</b>	repeat the previous line
<b>a</b>	after command, copy or move after this marker
<b>b</b>	before command, copy or move before this marker
<b>c</b>	copy, copy this line (needs an <b>a</b> or <b>b</b> target)
<b>e</b>	edit, for warp or select (IF, AND or OR) item
<b>exp</b>	expand entry
<b>d</b>	display entry
<b>del</b>	delete entry from display
<b>k</b>	kill (remove line command) and clear all following line commands
<b>m</b>	move, move this line (needs an <b>a</b> or <b>b</b> target)
<b>o</b>	options, display supported line commands
<b>r</b>	repro, copy this line to the next line
<b>s</b>	select, prompt for IF, AND or OR options
<b>w</b>	warp, prompt for warp options
<b>x</b>	drop, remove the line from the display
<b>==</b>	block =
<b>=nnn</b>	multiple =
<b>cc</b>	block c
<b>dd</b>	block d
<b>mm</b>	block m
<b>rr</b>	block r
<b>ss</b>	block s
<b>xx</b>	block x
<b>xnnn</b>	multiple x

## Expanding a Date Field

Date items can be expanded with a **EXP** line command or by specifying **YES** to the Expand prompt shown below. Either of these will format a STARBAT EXPAND parameter when the control statements are built.

```

----- WARP options -----
OPTION  ==>

Copybook LEVEL and NAME                PICTURE COLUMN TYP LENGTH
      02 MYFILE-HIT-ENTRY                2 BIN 74      BIN      2

Date or data picture ==> YYDDD           (descriptive picture: CCYMMDD)
Type of action       ==> ADD             (ADD or SUB to increment; DATE to insert)
Date adjustment      ==> 4Y             (number and code like 90D or actual date)
Corrective action    ==>                (action to take, like FLAG)
                                                    More:      +

Optional:
Date validation      ==>                (validation check, like WKEND)
Ignore Flags         ==>                (LIST of code characters)
Pivot Year           ==>                (0 to 99, for windowing two digit years)
Fiscal Month number ==>                (1 to 12) 1=January, 12=December
Holiday exit name    ==>                (routine name for holiday date checking)
Error DDNAME         ==>                (DDNAME to write records with errors)
Output picture       ==> CCYMMDD        (descriptive picture: CCYMMDD)
Output format        ==> C              (B, C or P for binary, character, packed)
Output location      ==>                (1-32760 for a column number)
Expand field         ==> YES           (Yes or No to expand the field)
Repeat for elements ==> YES             (Yes or No to apply to same name elements)
Depending on prompt ==> NO              (Yes or No to prompt for OCCURS DEPENDING)

```

Figure 48. Sample WARP Expand

STARWARP shows the information it has on the field for expansion and you should overtype information that is to be changed. In this case, a YYDDD picture date in binary format is to be changed to a CCYYMMDD picture in character format. Note that C (or character) expansion is recommended for use with STARWARP date expansions because the field is left intact at the original location with following spaces; when the WARP statement updates the date, it will replace data at that location and any needed positions after that point.

```

----- EXPAND field prompt -----
OPTION  ==>

Copybook LEVEL and NAME          PICTURE COLUMN TYP LENGTH
      02 MYFILE-HIT-ENTRY          2 BIN 74      BIN      2

Type of field          ==> C          (B, C or P for binary, character, packed)
Old length             ==> 2          (old original length of the field)
New length             ==> 8          (new length of the field)

Notes:
The EXPAND option will be performed before WARP processing.
The Type field should be set to C so that during expansion the
field will have blank pad characters added to the right. This
way the input date will not be shifted. This can be used to
expand the date field area in order to receive an expanded date
created by the OUTPIC specification. Type P or B will process
the field as a decimal number with leading zeros added during
expansion, which may create an unsupported input date format.

```

Figure 49. Sample Expand Field Prompt

### Warping Multiple Date Fields

If a table element is warped, the "Repeat for elements" line in the table is set to **YES** by default as shown in the example below. If this is left at the YES value, all elements of the table will be warped in the same way.

```

----- WARP options -----
OPTION  ==>

Copybook LEVEL and NAME          PICTURE COLUMN TYP LENGTH
      02 MYFILE-HIT-ENTRY          2 BIN 74      BIN      2

Date or data picture ==> YYDDD      (descriptive picture: CCYMMDD)
Type of action       ==> ADD        (ADD or SUB to increment; DATE to insert)
Date adjustment      ==> 4Y         (number and code like 90D or actual date)
Corrective action    ==>            (action to take, like FLAG)
                                                    More:      +

Optional:
Date validation      ==>            (validation check, like WKEND)
Ignore Flags         ==>            (LIST of code characters)
Pivot Year           ==>            (0 to 99, for windowing two digit years)
Fiscal Month number ==>            (1 to 12) 1=January, 12=December
Holiday exit name    ==>            (routine name for holiday date checking)
Error DDNAME         ==>            (DDNAME to write records with errors)
Output picture       ==> CCYMMDD    (descriptive picture: CCYMMDD)
Output format        ==> C          (B, C or P for binary, character, packed)
Output location      ==>            (1-32760 for a column number)
Expand field         ==> YES        (Yes or No to expand the field)
Repeat for elements  ==> YES      (Yes or No to apply to same name elements)
Depending on prompt  ==> NO         (Yes or No to prompt for OCCURS DEPENDING)

```

Figure 50. Sample WARP Repeat

At build time, we get the following results after combining an expand with a repeat for a table with three elements.

```

EDIT          WSER07.SPFTEMP1.CNTL          Columns 00001 00072
Command ==>          Scroll ==> CSR
***** ***** Top of Data *****
000001 DD01 COPYSOME RDW=3,MAXDATERR=100,
000002   EXPAND=( 74,C,2,8),
000003     WARP=( 74,BIN,YYDDD,
000004     OUTPIC=CCYYMMDD,OUTFORM=C,
000005     ACTION=FLAG,
000006     ADD=4Y),
000007   EXPAND=( 76,C,2,8),
000008     WARP=( 76,BIN,YYDDD,
000009     OUTPIC=CCYYMMDD,OUTFORM=C,
000010     ACTION=FLAG,
000011     ADD=4Y),
000012   EXPAND=( 78,C,2,8),
000013     WARP=( 78,BIN,YYDDD,
000014     OUTPIC=CCYYMMDD,OUTFORM=C,
000015     ACTION=FLAG,
000016     ADD=4Y),
000017   ABEND=1
***** ***** Bottom of Data *****

```

**Figure 51. Sample STARBAT Control Statements for Repeat**

### Warping Occur Depending On Fields

If a COBOL OCCURS DEPENDING ON table element is warped, the " Depending on prompt " line in the table is set to **YES** by default as shown in the example below. If this is left at the YES value, you will be prompted for additional details in another panel so that you can warp all elements of the table in the same way.

```

----- WARP options -----
OPTION  ==>

Copybook LEVEL and NAME          PICTURE COLUMN TYP LENGTH
      3  ARRAY                    X(8) 27      CHR      8

Date or data picture ==> CCYYMMDD      (descriptive picture: CCYYMMDD)
Type of action       ==> ADD          (ADD or SUB to increment; DATE to insert)
Date adjustment      ==> 20m         (number and code like 90D or actual date)
Corrective action    ==> FLAG        (action to take, like FLAG)
                                                    More:      +

Optional:
Date validation      ==>              (validation check, like WKEND)
Ignore Flags         ==>              (LIST of code characters)
Pivot Year           ==>              (0 to 99, for windowing two digit years)
Fiscal Month number ==>              (1 to 12) 1=January, 12=December
Holiday exit name   ==>              (routine name for holiday date checking)
Error DDNAME        ==>              (DDNAME to write records with errors)
Output picture       ==>              (descriptive picture: CCYYMMDD)
Output format        ==>              (B, C or P for binary, character, packed)
Output location      ==>              (1-32760 for a column number)
Expand field         ==> NO          (Yes or No to expand the field)
Repeat for elements ==> NO          (Yes or No to apply to same name elements)
Depending on prompt ==> YES       (Yes or No to prompt for OCCURS DEPENDING)

```

Figure 52. Sample WARP Depending On

In this OCCURS DEPENDING ON prompt panel, STARWARP will show the information it has on the table and the DEPENDING ON variable. You may overtype any of the input fields if desired. In this case, we specified that we want to warp at most two items but only if they are not set to all 9's.

```

----- Occurs Depending on a Variable -----
OPTION  ==>

Copybook LEVEL and NAME                PICTURE COLUMN TYP LENGTH
      3  ARRAY                          X(8) 27      CHR      8
Copybook LEVEL and NAME of DEPENDING ON variable
      3  XXB                             99 1        DIS      2

Column for variable ==> 1      (location of DEPENDING ON variable)
Maximum value      ==> 2      (maximum number)
Width in characters ==> 8      (number of characters between elements)
More: +

Optional AND test for a variable within the same repeated group.
Column or name     ==> array
Operator for compare ==> EQ      (EQ, NE, GT, LT, GE or LE)
Compare data       ==> c'99999999'

Notes:
For additional specific information on any of the above fields, place the
cursor in the field and press HELP for a detailed pop-up tutorial.

Copybook LEVEL and NAME is the data element name from the copybook table.

Copybook LEVEL and NAME of DEPENDING ON variable is the name of the variable
which controls the number of OCCURS elements in the above variable. This
name will be initialized if you have selected the DEPENDING ON variable
before this OCCURS DEPENDING ON element.

Column for variable is the column number where the DEPENDING ON variable
starts. This field will be initialized if you selected the DEPENDING ON
variable before this OCCURS DEPENDING ON element.

Maximum value is the maximum number of elements in the OCCURS DEPENDING ON
table that are to be checked

Width in characters is the number of characters between elements of the
OCCURS DEPENDING ON table. If the elements do not have any data between
them, you would just use the width of each element; this value is added
to the column number of an element to locate the next element.

Column for variable is the column number or name of another field to test
before warping a date. If this field is non-blank, an AND parameter will
be generated for testing a data value before warping its associated date
field.

Operator for Compare: This field specifies the type of comparison desired to
select records for warping.

Compare Data: This field specifies the data values to compare against record
values to select records for warping.

```

Figure 53. Sample Depending On Prompt

At build time, we get the following results.

```
EDIT          WSER07.SPFTEMP1.CNTL          Columns 00001 00072
Command ==>          Scroll ==> CSR
***** ***** Top of Data *****
000001 DD01  COPYSOME RDW=3,MAXDATERR=100,
000002   IF=(1,GE,C'01'),
000003   AND=(27,NE,c'99999999'),
000004       WARP=(27,C,CCYYMMDD,
000005       ACTION=FLAG,
000006       ADD=20M),
000007   IF=(1,GE,C'02'),
000008   AND=(35,NE,c'99999999'),
000009       WARP=(35,C,CCYYMMDD,
000010       ACTION=FLAG,
000011       ADD=20M),
000012       ABEND=1
***** ***** Bottom of Data *****
```

Figure 54. Sample STARBAT Control Statements for Depending On

### Saving a STARWARP Table

STARWARP tables can be saved in your profile. You could use a saved table to correct previous date or data warping specifications and rebuild a STARBAT job or use it as the starting point for another STARWARP session.

Once you are satisfied with all the selection and WARP criteria, enter a **SAVE** primary command as shown below.

```

----- Copybook for StarWarp ----- Row 1 to 25 of 28
COMMAND ==> save                               SCROLL ==>
CSR_

      Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
      Line commands: S to Select records, W to Warp dates; Press PF1 to obtain
Help

CMD  MSG      NAME                                PICTURE COLUMN TYP  LENGTH
----
_____ 01 TESTBOOK                                1          80
_____ 02 MYFILE-SKIP                                X(1) 1     CHR   1
_____ 02 MYFILE-IDENTIFIER                          X(14) 2     CHR  14
_____ 02 MYFILE-KEY                                  16          9
_____ 05 MYFILE-KEY-FIRST2                          X(2) 16     CHR   2
_____ 05 MYFILE-KEY-REST                            X(7) 18     CHR   7
_____ *SELECT* 02 MYFILE-PLACE-OF-BUSINESS            X(19) 25     CHR  19
_____ *WARP*  02 MYFILE-DATE-OF-LAST-CHANGE          X(8) 44     CHR   8
_____ 02 MYFILE-FIRST-BINARY                        4 BIN 52     BIN   4
_____ 02 FILLER                                      X 56     CHR   1
_____ 02 MYFILE-MISSION-KEYS                        57          4
_____ 05 MYFILE-KEY-ONE                             X(1) 57     CHR   1
_____ 05 MYFILE-KEY-TWO                             X(1) 58     CHR   1
_____ 05 MYFILE-KEY-THREE-FOUR                     X(1) 59     CHR   1
_____ 05 MYFILE-KEY-THREE-FOUR                     #2 60     CHR   1
_____ 02 FILLER                                      XX 61     CHR   2

```

Figure 55. Sample SAVE Primary Command

In the STARWARP Save panel, you should enter a saved table name and optionally, a description of the table.

```
----- StarWarp Save -----  
-  
COMMAND ==>  
  
Specify the name for a saved version of this StarWarp copybook table  
StarWarp saved name    ==> wptst1  (3 to 6 character name)  
Replace existing entry ==> NO      (Yes or NO)  
Copybook description   ==> This is a test  
Save to library        ==> PROFILE (PROFILE, GROUP, OR OTHER)  
Other partitioned data set:  
DATA SET NAME    ==>  
  
Notes:  After this is saved in the profile, you can rename, delete or  
         activate these saved WARP tables with the PROFMAN dialog.  
         StarWarp copybook tables use entry type SW in the PROFMAN dialog.  
  
Profile DDname      :ISPPROF  
Group   DDname      :ANYDDNAM
```

Figure 56. Sample STARWARP Save Prompt

### WARP Table Views

The following screen is the normal view for the WARP table. You can obtain the other views in order by pressing RIGHT or you can press LEFT to return to a previous view.

```

----- Copybook for StarWarp WPTST1----- Row 23 to 28 of 28
COMMAND ===>                                SCROLL ===> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  MSG      NAME                                PICTURE COLUMN TYP  LENGTH
----
_____ 02 MYFILE-HIT-ENTRY                #2 76      BIN   2
_____ 02 MYFILE-HIT-ENTRY                #3 78      BIN   2
_____ 02 LAST-DATA                        X 80      CHR   1
_____ *** START OF CONTROL CARD ENTRIES ***
_____ *IF      02 MYFILE-PLACE-OF-BUSINESS        X(19) 25      CHR  19
_____ *WARP    02 MYFILE-DATE-OF-LAST-CHANGE    X(8) 44      CHR   8
***** Bottom of data *****

```

Figure 57. Sample WARP Table Showing Element Names

```

----- Copybook for StarWarp ParmS WPTST1--- Row 23 to 28 of 28
COMMAND ===>                                SCROLL ===> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  MSG      PARM OP DATA                                FORMAT      ACTION
VALID
_____
_____ *****
_____ *****
_____ *****
_____ *--*
_____ *IF      IF  EQ C'Las Vegas'
_____ *WARP    WARP AD 4Y                                CCYYMMDD    NEXTBDAY
***** Bottom of data *****

```

Figure 58. Sample WARP Table Showing Control Information #1

```

----- Copybook for StarWarp Optional fields WPTS Row 23 to 28 of 28
COMMAND ==> SCROLL ==> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  MSG          PARM OUT LOC FROM OUT PICTURE          HOLIDAY PIVOT FY IGNORE  CB#
----  ---          *---*
____  _____  *---*
____  _____  *---*
____  _____  *---*
____  _____  *---*
____  *IF          IF _____
____  *WARP        WARP          CALEXIT          1
***** Bottom of data *****

```

Figure 59. Sample WARP Table Showing Control Information #2

```

----- Copybook for StarWarp WPTST1----- Row 23 to 28 of 28
COMMAND ==> SCROLL ==> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD  NAME          PICTURE COLUMN TYP LENGTH  VALID
----  ---          *---*
____  _____  #2 76  BIN    2
____  _____  #3 78  BIN    2
____  _____  X 80  CHR    1
____  *** START OF CONTROL CARD ENTRIES ***
____  *---*
____  02 MYFILE-PLACE-OF-BUSINESS          X(19) 25  CHR    19
____  *IF          IF EQ C'Las Vegas'
____  02 MYFILE-DATE-OF-LAST-CHANGE          X(8) 44  CHR    8
____  *WARP        WARP AD 4Y          CCYYMMDD  NEXTBDAY
***** Bottom of data *****

```

Figure 60. Sample WARP Table with Double Lines

```

----- Copybook for StarWarp WPTST1----- Row 23 to 28 of 28
COMMAND ==>
                                SCROLL ==> CSR_

Primary commands: S to Save table, B to Build JCL; HELP to obtain Help
Line commands: S to Select records, W to Warp dates; Press PF1 to obtain Help

CMD   NAME                                PICTURE COLUMN TYP LENGTH
MSG   PARM OP DATA                        Format      ACTION      VALID
      OUT LOC FROM OUT PICTURE            HOLIDAY PIVOT FY IGNORE  CB#
-----
02 MYFILE-HIT-ENTRY                      #2 76      BIN        2
****
1
02 MYFILE-HIT-ENTRY                      #3 78      BIN        2
****
1
02 LAST-DATA                              X 80      CHR        1
****
1
*** START OF CONTROL CARD ENTRIES ***
*--*
1
02 MYFILE-PLACE-OF-BUSINESS              X(19) 25   CHR        19
*IF      IF      EQ C'Las Vegas'
1
02 MYFILE-DATE-OF-LAST-CHANGE            X(8) 44   CHR        8
*WARP    WARP AD 4Y
                                CCYYMMDD  NEXTBDAY
                                CALEXIT
1
***** Bottom of data *****

```

Figure 61. Sample WARP Table with Triple Lines

## Modeling the Input Data Set

If you respond **YES** to "Model the input" as shown in the example below, STARWARP will invoke itself using a MODEL command to obtain the attributes of the input data set.

```

----- StarWarp Data Set Prompt -----
COMMAND ==>

  Foreground execution ==> YES          (YES for Foreground, NO for batch)
  Input data set       ==> LIB.cntl
  Output data set     ==> LIB.CNTLNEW
  Model the input      ==> yes        (Yes or No to Model the input data set)
  Copy type operation ==> COPY SOME    (COPY SOME/COPY ALL/UPDATE ALL/COPY/UPDATE)
  Maximum date errors ==> 100         (1 to 9999999, for invalid date messages)
  Record Desc Word    ==> 3           (RDW, default 3)
  
```

Figure 62. Sample Model the Input

When you reach the MODEL input screen, you must change the data set name (it will be the name of the data set modelled) and you may change any other parameter you want before pressing ENTER to see the data set allocation messages.

```

----- MODEL entry panel -----
OPTION ==>

  New DSNAME   ==> 'WSER07.LIB.CNTLnew'
    on Volume ==> SER001      (optional)
    or Unit    ==>           (optional)
  Space units ==> TRK        (BLK, TRK or CYL)
  Round space ==> NO         (YES or NO to round up to nearest CYL)
  Primary space==> 2400      CURRENT Allocated space: 2400   TRKS
  Secondary    ==> 200       Unused space: 1706   TRKS
  Directory blk==> 88       Used directory blk: 55
  RECFM       ==> FB        DSORG: PO
  LRECL       ==> 80        Allocation type: TRK
  BLKSIZE     ==> 11440     Secondary quantity: 200
  Expiration date=>         (Julian)      Number of extents: 8
  Retention time =>         (number of days, this overrides EXPDT)
  Data class   ==>
  MGMT class   ==>
  Storage class ==>
  DSNTYPE      ==>         (PDS/LIBRARY)

  | Press RCHANGE to save in PROFILE
  | Press END to exit with no changes
  | Press ENTER with non-blank DSNAME
  | to create a new data set
  
```

Figure 63. Sample MODEL Prompt

STARWARP will show the results and you may press **END** to return to the previous build process. If the MODEL command fails for any reason, you may enter **MODEL** to restart the modelling process.

```

----- ISPMODE Session# 1 Log# 1 ----- Row 16 to 19 of 19
COMMAND ===>                                SCROLL ===> CSR
Enter an ISPF command, a StarTool subcommand or a special control code:
- DSN=WSER07.LIB.CNTL,VOL=SER=SER001 MEM= -----
>----->Create 'WSER07.LIB.CNTLNEW' TRK VOLUME(SER001) DATACLAS(*) MGMTCLAS(*)
-----STORCLAS(*) DSNTYPE(PDS) DIR(88) SPACE(2400,200) RECFM(FB) LRECL(80) BL
-----KSIZE(11440)
PDS174I 'WSER07.LIB.CNTLNEW' has been created on volume SER001
***** Bottom of data *****

```

**Figure 64. Sample MODEL Results**

## The STARBAT Program

The JCL and control cards for warping data can be created manually instead of using online panels. This section provides a brief overview of STARBAT and some of the frequently used STARBAT control cards for data aging. For full details on STARBAT, see the **STARBAT for STARWARP Guide**.

STARBAT is a MVS batch program designed to perform data manipulation tasks. You can use STARBAT to copy selected records or portions of records from one data set type to another, print data in different formats, expand data fields in records and process data set members selectively.

STARBAT uses standard MVS JCL. Functions and their associated parameters describing the operations to be performed are passed to STARBAT via control statements defined in the SYSIN DD statement. STARBAT supports sequential, direct, VSAM and partitioned data sets.

The following sample JCL may be used to execute STARBAT.

```
//STARBAT1 JOB (JOB CARD PARAMETERS)
//STEP1 EXEC PGM=STARBAT,REGION=2048K
//DD01 DD DSN=WSER11.INPUT.DATA,DISP=SHR
//DD01O DD DSN=WSER11.OUPUT.DATA,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSLIST DD SYSOUT=*
//SYSTOTAL DD SYSOUT=*
//SYSIN DD *
DD01 COPYREC IF=(1,0,C'ABC')
```

Figure 65. Sample STARBAT JCL

## Optional DD Statements

**SYSPRINT** lists all control statements and all error, status and completion messages; **SYSLIST** lists all output from DUMP, LIST, and PRINT functions and parameters and **SYSTOTAL** lists all comment control cards.

The SYSPRINT and SYSLIST DD statements will be dynamically allocated if they are not present while the SYSTOTAL DD statement will be ignored if not present. These outputs may be directed to a data set with RECFM=FBA, LRECL=133, and BLKSIZE = multiple of 133.

## Required DD Statements

**DDxx** describes the input data sets to STARBAT. The xx value can be any number from 00 through 99 that matches a control statement name like DDxx in the SYSIN input. For COPY and DROP functions, the xx also has to match a DDxxO DD statement.

**DDxxO** is required for the COPY or DROP functions. The xx value must match the xx value for the input data set defined in the corresponding DDxx statement. For example: `//DD01O` is the output data set corresponding to the input data set with DD statement `//DD01`. The DCB information will be copied from the input data set to the output data set if necessary. The output data sets may reside on any output media supported by MVS.

**SYSIN** provides control statements for STARBAT. Control statements must be in standard 80 character format and all 80 columns may be used. The SYSIN JCL statement may reference any sequential input supported by MVS.

## Control Statements

Control statements are made up of a data set identifier, a function, one or more parameters separated by commas and comments.

In the following example, **DD01** is the data set identifier, **COPYREC** is the function, **IF** and **WARP** are the parameters, and **TEST CRITERIA** and **AGING** are comments.

```
DD01 COPYREC IF=( 25 ,EQ ,C'Las Vegas' ) ,          TEST CRITERIA
          WARP=( 44 ,C ,CCYYMMDD ,ADD=4Y ,ACTION=ROLL )  AGING
```

## Control Statement Syntax

**Data Set Identifier** Must begin in column 1. Has format **DDxx** where **xx** is a number from 00 through 99 that corresponds to a matching `//DDxx DD` statement. For COPY and DROP functions, the **xx** also has to match an output `//DDxxO DD` statement.

**Function Identifier** Entered after the data set identifier and one or more spaces.

**Parameters** Entered as a keyword followed by an equal sign and one element as a single entity or as multiple elements enclosed in parenthesis. These elements usually contain a combination of location, length/operator, and data fields.

For example: **IF=( 25 ,EQ ,C'Las Vegas' )**

**25** is the location

**EQ** is the operator

**C'Las Vegas'** is the data

## Frequently Used Functions

- COPYREC** The COPYREC function is used to copy specific records to an output data set.
- COPY SOME** The COPY SOME function is used to copy specific records and apply all actions.
- COPY ALL** The COPY ALL function is used to copy all records to an output data set.
- COPY MBR** The COPY MBR functions is used to copy members selectively.
- UPDATEREC** The UPDATEREC function is used to update records in place. Use UPDATEREC to make permanent changes to an existing data set instead of a copy of the data set. You may preview all changes by using the print functions (PRINTHEX, PRINTCHR or PRINT) in a first pass. If all changes are correct, you may then change the function name to UPDATEREC and resubmit the job to make the actual update to the file.
- PRINTHEX** The PRINTHEX function prints data records in character and vertical hexadecimal format with additional information such as record number, RBA address and record length.
- PRINTCHR** The PRINTCHR function prints data records in a simple alphanumeric format.
- PRINT** The PRINT function prints data records in alphanumeric format with additional record information such as record number, RBA address and record length.

## Frequently Used Parameters

- IF** IF=( location , length / operator , data , loc2 , len2 / oper2 , data2 , ... )  
IF selects records to be processed by the function being executed.
- MAXDATERR** MAXDATERR=n  
MAXDATERR controls the number of invalid dates to permit before terminating.
- MOVE** MOVE=( location , length , from-loc )  
MOVE is used to build a record in parts when a field is expanded (for example, if a field is being changed from MMDDYY to MMDDCCYY format).
- WARP** WARP=( location , type , picture , DATE=dd , ADD=nni , SUB=nni , ACTION=a , VALID=vv , HOLIDAY=name , OUTLOC=loc , OUTFORM=type , OUTPIC=picture )  
WARP is used to modify numeric data values and dates.

## Examples

Add 5 years to the date and move the date to the next month if the derived date is invalid.

```
DD01 COPYREC WARP=( 10 , C , MM/DD/CCYY , ADD=5Y , ACTION=ROLL )
```

Subtract 2 months from the date and convert the output date format

```
DD01 COPYREC WARP=( 10 , C , MMDDYY , SUB=2M , OUTPIC=MMDDCCYY )
```

If column 10 equals to a '1', add 5 years to the date, move the date to the end of the month, and copy the entire file

```
DD01 COPYALL IF=( 10 , EQ , C ' 1 ' ) , WARP=( 20 , C , MMDDYY , ADD=5Y , ACTION=MEND )
```

Update all records with a '1' in column 5 and an 'A' in column 10 and subtract 5 months from the date

```
DD01 UPDATEREC IF=( 5 , EQ , C ' 1 ' ) , AND=( 10 , EQ , C ' A ' ) , WARP=( 20 , C , MMDDYY , SUB=5M )
```

Update all records with a '1' in column 5 or an 'A' in column 10 and subtract 5 months from the date

```
DD01 UPDATEREC IF=( 5 , EQ , C ' 1 ' , 10 , EQ , C ' A ' ) , WARP=( 20 , C , MMDDYY , SUB=5M )
```

Select all records with a '1' in column 5 and add 5 months or records with an 'A' in column 10 and add 1 year.

If the first condition is met, the second condition will not be tested; use **COPYALL** to test both conditions.

```
DD01 COPYREC IF=( 5 , EQ , C ' 1 ' ) , WARP=( 20 , C , MMDDYY , ADD=5M ) ,  
OR=( 10 , EQ , C ' A ' ) , WARP=( 30 , C , HYYMMDD , ADD=1Y )
```

Copy the file, modify the date and output it to 2 different locations and change the format

```
DD01 COPYREC WARP=( 20 , C , MMDDYY , ADD=10Y , OUTLOC=40 , OUTPIC=MMDDCCYY ) ,  
WARP=( 20 , C , MMDDYY , ADD=20Y , OUTLOC=50 , OUTPIC=CCYYMMDD )
```

Expand a date field from 6 to 8 characters and change the output date format

```
DD01 COPYREC EXPAND=( 12 , C , 6 , 8 ) , WARP=( 12 , C , YYMMDD , OUTPIC=CCYYMMDD )
```

## Change Management and STARWARP

---

STARWARP has many facilities which can help manage members and data sets before, during and after change management is implemented at an installation. This section will discuss several ways in which STARWARP can work with Change Man, a change management package from SERENA.

One facility which can assist in all of these phases of Change Man implementation is "saved data set lists". Data set lists can be built with any combination of data sets and saved, recalled or merged by name.

The HISTORY subcommand is useful in searching for COBOL modules compiled with specific options (for example: **COBOL(RESIDENT,NORENT,DATA31)**) It can also search for load members by generic compiler or assembler names (for example: **TRANS(COBOL)**, **TRANS(COBOL370)**, **TRANS(ASM)**, or **TRANS(ASMH)**)

Before Change Management is implemented, STARWARP can help inventory development, test and production data sets. STARWARP contains many power tools to investigate, search and report on many different types of data.

During Change Man implementation, STARWARP can help convert components to Change Man control. STARWARP contains several mass facilities to process multiple members or multiple data sets with a single command. These facilities can identify and correct components which do not yet comply with the new standards required by the change management system.

After Change Man has been implemented, STARWARP can help analyze and convert new versions of existing applications under Change Man control. In addition, STARWARP can validate that members and data sets are being used correctly and effectively under Change Man.

Several facilities in STARWARP are very useful in a change management environment:

- |                 |   |
|-----------------|---|
| <b>MEMLIST</b>  | displays the package date based on the SSI information added by Change Man for a stage or recompile from baseline   |
| <b>MAPLCT</b>   | produces Linkage Control Table data (linkage edit control statements) for <b>CMNSSIDN</b> .   |
| <b>LCTJCL</b>   | separates <b>MAPLCT</b> data into individual LCT members  |
| <b>GENERATE</b> | produces data which may be used to preload the Change Man history file. This will help convert new applications to Change Man control and new versions of existing internal or vendor supplied systems to Change Man control.         |
| <b>CMNLOAD</b>  | produces <b>CMNBAHST</b> inputs from the <b>HISTORY GENERATE</b> outputs. This rexx exec is designed to be tailored at each installation to identify compiler and linkage options of interest before Change Man history is preloaded. |

## MEMLIST Option for Change Man

STARWARP displays the package date based on the SSI information added by Change Man for a stage or recompile from baseline if the SETPANEL option for the Change Man panel is enabled.

```

----- Set Panel Defaults -----
OPTION ==>
  To save these variables in your ISPF profile as defaults for future StarTool
  sessions, enter SAVE as a primary command or press PF6 or PF18
                                          More:      +
. . . . .

MEMLIST Displays for load modules in order of RIGHT rotation:
Show Attributes view  ==> YES  (Yes/No) -- Load module attributes
Show TTR, Size view  ==> YES  (Yes/No) -- TTR location and Size in bytes
Show Double line view ==> YES  (Yes/No) -- Combined information
Show Change Man view ==> YES  (Yes/No) -- Change Man date
  
```

**Figure 66. Sample STARWARP SETPANEL Option for Change Man**

After turning on the Change Man MEMLIST panel, it will be made available for displaying load members.

```

FUNCTIONS  CONTROL A-M  LINE CMDS A-M  DSN CMDS  MEM CMDS A-M  FEATURES
          CONTROL N-Z  LINE CMDS N-Z          MEM CMDS N-Z  DEFAULTS
-----
----- Load MEMLIST (Change Man), Session# 1 --- Row 1 to 3 of 3
COMMAND ==>
- DSN=WSER07.LINK.LOAD,VOL=SER=SER001 MEM=PDSE -----
CMD  NAME      DATA/MSG ALIASOF  LEN/LKED  --  ATTRIBUTES  -  CMN date time
CMNAPSPL
CMNOFM      1994/04/14 REUS
PDSE        *ALIAS  1996/05/03 REFR RENT
  
```

**Figure 67. Sample STARWARP Change Man Panel**

## MAP LCT -- Produce Linkage Control Table Statements

The MAP subcommand has an LCT option to produce Linkage Control Table inputs for the CMNSSIDN program.

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
EDIT          -----> map cmnapspl                      Columns 00001 00072
Command ==>                                     Scroll ==> CSR
***** ***** Top of Data *****
000001
000002 ** MAP          CMNAPSPL
000003 CMNAPSPL 00000000 00005CF0  RMODE ANY AMODE ANY
000004 PDS441W IGZETUN (Weak)
000005 PDS441W IGZEOPT (Weak)
000006 IGZEBST 00005CF0 00000450  RMODE ANY AMODE 31
000007                                     IGZEB2 00005FAE
000008
000009 PDS103I Entry point at 00000000 -- CMNAPSPL
000010 PDS104I Module length 00006140 -- 25K

```

Figure 68. Standard MAP Output

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
EDIT          -----> map CMNAPSPL lct                  Columns 00001 00072
Command ==>                                     Scroll ==> CSR
***** ***** Top of Data *****
000001
000002 ** MAP          CMNAPSPL
000003 **LKED      EXEC  PGM=IEWL,
000004 **          PARM= 'NCAL,MAP,LIST,LET,RENT,REUS '
000005 INCLUDE     SYSLIB(CMNAPSPL)
000006 ORDER       CMNAPSPL,IGZEBST
000007 SETSSI        40BE1799
000008 MODE         RMODE(ANY),AMODE(31)
000009 ENTRY        CMNAPSPL
000010 NAME          CMNAPSPL(R)

```

Figure 69. MAP LCT Output for One Member

## STARWARP User Guide

The LCTJCL batch JCL member can separate MAP LCT statements into sets of linkage edit inputs by name.

```
----- Source MEMLIST 1, Session# 1 ----- Row 1 to 19 of 19
COMMAND ===>                               SCROLL ===> CSR
- DSN=P390.LIB.JCL,VOL=SER=SER001 MEM=: -----
CMD  NAME      DATA/MSG VER.MOD  CREATED      LAST  MODIFIED  SIZE  INIT  ID
     DSAT
     DSATL
     DSATMAIN
     DSATREPL
     DSATZ
     DSAT2
     DSAT2000
     DSAT2001
     DSAT381
```

**Figure 70. MEMLIST Resulting from LCTJCL Execution**

```
Menu  Utilities  Compilers  Help
-----
BROWSE      P390.LIB.JCL(DSAT)                Line 00000000 Col 001 080
Command ===>                               Scroll ===> CSR
***** Top of Data *****
**LKED     EXEC  PGM=IEWL,
**          PARM= ' NCAL,MAP,LIST,LET,RENT,REUS,REFR '
INCLUDE    SYSLIB(DSAT)
ORDER      DSAT,DSATPDS,DSATPCL
SETCODE    AC(1)
ENTRY      DSAT
ALIAS      DSATA,DSATAA
NAME       DSAT(R)
```

**Figure 71. Sample MAPLCT Member**



```

----- ISPMODE Session# 1 Log# 1 ----- Row 73 to 91 of 133
COMMAND ==>
- DSN=WSER07.HISTOUT,VOL=SER=SER001 ----- SCROLL ==> CSR
*
CMP=CMNAPSPL,LIB=L0D
COP=
LOP=RENT,REUS,RMODE(ANY),AMODE(31)
*
CMP=CMNAPSPL,LIB=SRC
COP=NOADV,APOST,DATA(24),NOD,DYN,LIB,NUM,OPT,RENT,NOSEQ,TERM,TRUNC(OPT),
LOP=RENT,REUS,RMODE(ANY),AMODE(31)
PRC=CMNCOB2,LNG=COBOL2
*
CMP=CMNZMGR,LIB=SRC
COP=FLOW,OPT,STATE
LOP=REUS
    
```

Figure 74. Output from CMNLOAD for Input to CMNBAHST

```

CMNUSR01 ----- STAGE: USER OPTIONS -----
COMMAND ==>

NAME: LIOCCP          TYPE: SRC          LANGUAGE: COBOL

COMPILE ONLY          ==>          ENTRY TO DLITCBL    ==>
CICS PRE-COMPILE      ==> Y          DROP INCLUDE STMTS ==>
EASYTRIEVE NO NAME    ==>          USER OPTION 06     ==>
USER OPTION 07         ==>          USER OPTION 08     ==>
USER OPTION 09         ==>          USER OPTION 10     ==>
USER OPTION 11         ==>          USER OPTION 12     ==>
USER OPTION 13         ==>          USER OPTION 14     ==>
USER OPTION 15         ==>          USER OPTION 16     ==>
USER OPTION 17         ==>          USER OPTION 18     ==>
USER OPTION 19         ==>          USER OPTION 20     ==>

Use Y or N to select or deselect additional processing options;
Press ENTER to continue; Enter END command to exit.
    
```

Figure 75. Stage User Options

```

CMNSTG04 ----- STAGE: COMPILE AND LINK EDIT -----
COMMAND ==>

PACKAGE ID: CKLY000712      STATUS: DEV      INSTALL DATE: 960611

STAGED NAME:      LIOCCP
LIBRARY TYPE:     SRC - Source Code
LANGUAGE:         COBOL
DATASET NAME:     WSER99.CMN.S410DP.CKLY.#000712.SRC

COMPILE PROCEDURE ==> CMNCOBOL (Blank for list; ? for designated procedure)
COMPILE PARMS     ==> APOST
LINK EDIT PARMS   ==> NCAL
DB2 PRE-COMPILE   ==> NO      (Y/N)
OTHER OPTIONS     ==> YES     (Y/N to display other options)
    
```

Figure 76. Stage Compile and Link Attributes from CMNBAHST

## Batch Processing

---

It surprises most users that batch operation of STARWARP is just as flexible and easy to use as STARWARP in line mode from a terminal. The complicated maneuvers that you learn to take for granted are just as easily accomplished in a batch job. The big difference is that all the steps must be anticipated in advance. In batch, you cannot usually evaluate the results of the first step before beginning the execution of the following steps. The results of all STARWARP steps are displayed in batch, however, because STARWARP always uses the PUTLINE-GETLINE TSO interface in its operation. With a little practice, you can design incisive and efficient batch jobs to perform many tasks.

It may be desired to design a batch job under circumstances when you would rather not wait at a terminal for some long execution. Examples might be: Adding ISPF statistics for a lot of libraries. Cloning your system libraries or other libraries through repeated execution of COPY and CHANGE subcommands. CLISTS that execute STARWARP subcommands may be long in running. It may be easier to execute them in a batch job, rather than fussing with the same process at a terminal.

Several facilities are useful in this environment.

1. The **CONDEND** (conditional END) subcommand may be placed just after a critical STARWARP subcommand (see example 2 below) to terminate STARWARP if a warning or error message is generated by that subcommand.
2. To capture entered subcommands and test their syntax before creating a batch JOB, enter **CONTROL TESTSYNTAX**. Note that ISPMODE facilities will still function as well as the CHANGE, CONTROL, QUIT and END subcommands.
3. To suppress informational messages, enter **CONTROL NOINFO**
4. To suppress subcommand echoing in the session log, enter **CONTROL NOCOMMAND**
5. To suppress all messages (except error messages), enter **CONTROL NOPUTLINE**
6. To capture the session output in a RECFM=FB,LRECL=80 data set, enter a subcommand similar to **CONTROL DSN(data.set(memnam))**. You may also add a disposition parameter: OLD, MOD, SHR or NEW.

It is often effective to generate input for STARWARP through some process that formats calls to a CLIST as shown in the following:

```
STARWARP 'SYS1.PROCLIB'
%CLISTP 'DATA.SET.TO.MANIPULATE'
%CLISTP 'ANOTHER.DATA.SET'
...
```

Note that CLISTP would contain a STARWARP subcommand to CHANGE to the data set and additional subcommands to manipulate the data set as needed. Any data encountered under CLIST mode (from an implied CLIST call or the EXEC subcommand) is assumed to be STARWARP subcommands or CLIST commands. If they conflict because their names are the same, you may use an abbreviated or alternate name to reference the STARWARP subcommand. For example to reference IF and CONTROL subcommands, you could use the subcommand names, IFX and CONTRO respectively.

One sample of JCL will suffice for non-ISPMODE applications of STARWARP in batch mode. The only variations are in the command input under the ddname SYSTSIN, in the program libraries included in a STEPLIB and in the CLIST libraries specified for SYSPROC. There is extra power in batch mode: ordinary TSO commands may be executed before or after STARWARP executions. Also, STARWARP can call ordinary TSO commands while executing, by using the STARWARP TSO subcommand as an interface within the batch job.

An example with JCL to execute STARWARP in the background without ISPF facilities follows; this example adds ISPF statistics to all members in several large libraries. Note that this sample is in the CNTL library distributed with STARWARP as member PDSBATCH.

```
//TSOBATCH JOB (JOB CARD PARAMETERS)
//TSOBATX EXEC PGM=IKJEFT01,REGION=3600K,DYNAMNBR=50
//STEPLIB DD DISP=SHR,DSN=load.library (if applicable)
// DD DISP=SHR,DSN=load.library2 (if applicable)
//SYSPROC DD DISP=SHR,DSN=clist.library (if applicable)
// DD DISP=SHR,DSN=clist.library2 (if applicable)
//SYSTSPT DD SYSOUT=* <== OUTPUT OF COMMANDS
//SYSTSIN DD * <== TSO COMMAND INPUT
STARWARP 'SYS1.HELP'
ATTRIB : ADDSTATS
C 'SYS1.MACLIB'
ATTRIB :
ADDSTATS ID(ESA43)
END
```

### ISPMODE in Batch

STARWARP supports the batch use of ISPMODE facilities if STARWARP panel and message data sets are allocated in an ISPF environment. This facility is useful for many different applications such as saving LISTC/LISTF, MEMLIST or WORKPAD tables for interactive use or processing multiple data sets in a table with global commands.

To support ISPMODE facilities in batch, several changes were made:

- Prompt panels simulate an **END** response.
- Syntax errors terminate the batch session.
- Checkpoints are ignored in the log.
- The **OUTPUT** command produces PUTLINE output by default.
- Line commands are not supported except with the **APPLY** command.

Several ISPMODE facilities are especially useful in a batch environment:

1. Global processing can process all data sets in a LISTA/DDNAME, LISTC/LISTF or WORKPAD table.
2. Global processing is very effective in searching, updating or compressing multiple data sets. Note however that in LISTC/LISTF tables, the **FIND**, **REPLACE** and **COMPRESS** commands skip over non-partitioned data sets.
3. **FIXPDS** with its percentage operators to modify data sets (**DIRFREE**, **ADDFREE** and **RELFREE**) is very effective at changing multiple data sets using the **GLOBAL** command.
4. Saved MEMLIST and LISTC/LISTF tables allow a batch application to save member or data set names of interest for other types of processing.
5. Saved MEMLIST and LISTC/LISTF tables could also be used to contain a list of member or data set names that a batch application is to process.
6. The **EXCLUDE** command can be used to trim dialog tables.
7. The **APPLY** command can be used to perform the same line command for all table entries.
8. The **SORT** command can also be used in conjunction with a **F** or **LOCATE** command to position to data of interest in a table followed by a **X ABOVE** command to trim dialog tables.
9. The **TRAP** command can be used to capture PUTLINE output from TSO commands in the log.
10. The **MASK** command can be used to build LISTC/LISTF tables.
11. The **FILTER** command can be used to set up data set filtering defaults for the **MASK** command.

In a batch application, you can filter data sets by type with global processing by using a variation of the GLOBAL name. This means that you do not need to trim a LISTC/LISTF table before executing GLOBAL commands.

The following GLOBAL data set type filter names may be used:

**GLOBAL** process all data sets (except MIGRAT or ARCHIV data sets)  
**GLOBALM** process PDSE or PDS data sets  
**GLOBALE** process PDSE data sets  
**GLOBALO** process PDS data sets  
**GLOBALS** process source PDS or PDSE data sets only (RECFM is not U)  
**GLOBALL** process load PDS or PDSE data sets only (RECFM is U)  
**GLOBALQ** process sequential or direct data sets only  
**GLOBALN** process non-VSAM data sets only  
**GLOBALV** process VSAM clusters only  
**GLOBALT** process all data sets marked with \*TAG\* in the DATA/MSG field

A sample of JCL to execute STARWARP in the background with ISPF facilities follows: Note that this sample is in the CNTL library distributed with STARWARP as member PDSISPF.

```
//TSOBATCH JOB (JOB CARD PARAMETERS)
//TSOBATX EXEC PGM=IKJEFT01,REGION=3600K,DYNAMNBR=50
//STEPLIB DD DISP=SHR,DSN=load.library (if applicable)
// DD DISP=SHR,DSN=load.library2 (if applicable)
//SYSPROC DD DISP=SHR,DSN=clist.library (if applicable)
// DD DISP=SHR,DSN=clist.library2 (if applicable)
//SYSTSPT DD SYSOUT=* <== OUTPUT OF COMMANDS
//ISPPROF DD DISP=SHR,DSN=profile.library (or a temporary):
//ISPPROF DD DISP=NEW,DSN=&&ISPF,UNIT=SYSDA,SPACE=(TRK,(5,5,5)),
// DCB=(LRECL=80,BLKSIZE=9120,RECFM=FB)
//ISPPLIB DD DISP=SHR,DSN=panel.library (and concatenations)
//ISPMLIB DD DISP=SHR,DSN=message.library (and concatenations)
//ISPSLIB DD DISP=SHR,DSN=skeleton.library (and concatenations)
//ISPTLIB DD DISP=SHR,DSN=table.library (and concatenations)
//SYSTSIN DD * <== TSO COMMAND INPUT
PROF MSGID PREFIX(userid)
ISPSTART PGM(STARWARP) PARM(STARWARP SDS.CNTL) NEWAPPL(ISR)
FILTER NOMIGRAT
MASK SER08.*.DATA
OUTPUT
COMPRESS
END
```

## F Command

Every function has a F (find) command to locate character strings within the function tables.

The find string is positional (it must be entered just after the **F** keyword) and it is normally entered as a simple string delimited by blanks; however, it may be delimited with quotes. The following examples show several valid commands:

```
F 'This is the first string' first
F "this is the second string" last
F mydata dsname
```

Normally, the ANY keyword is defaulted with a find command; this means that all data columns will be searched. If you desire to search only a specific column of data, you may enter the name of that data column. Another way to direct the find search is with keywords as NEXT, FIRST, LAST, PREFIX, SUFFIX and WORD. These keywords have the normal ISPF-implied meanings. If you are dealing with a lot of information, the find facility can help zero in on any desired information.

## EXCLUDE Facility

**EXCLUDE** facilities are available for all function tables. From a log or MEMLIST table, the **EXCLUDE** subcommand may be used to trim members from the MEMLIST table. The EXCLUDE subcommand supports the same attribute filtering operands that are available on a MEMLIST subcommand.

From other tables, the **EXCLUDE** command may be used to trim all table elements that match a search string (or do not match a search string with the **NOT** keyword). The EXCLUDE command supports the same operands as available on the F command with the exception of the directional keywords (NEXT, FIRST and LAST) because EXCLUDE always processes an entire table.

## SORT Command

The **SORT** command may be used in functions with sorted tables: CAX, CSECTS, LISTC/LISTF, LISTV, MEMLIST and NUCMAP. SORT with no operands sorts the table in default order, while SORT with a field name sorts the table in that order.

## L Command

The **L** (or LOCATE) command may also be used in functions with sorted tables. The L command locates data in the current sorted column. A partial string may be supplied to obtain a generic search.

## X Command

Another standardized feature is the **X** command. When X is used with an operand, it can be used to trim an entire table. The supported operands are ALL, ABOVE and BELOW where the current position in the table is the trim reference point.

## OUTPUT Command

The **OUTPUT** command is available for all functions to output the current log or table to SYSOUT or a data set. In batch mode if no operand is entered on the OUTPUT command, the table is output with PUTLINE messages. You may also enter an operand such as **=x** or **F(ddname)** after OUTPUT. If **=x** is entered, output is directed to **SYSOUT=X** and if **F(ddname)** is entered, output is directed to the named preallocated data set.

## Quick Start to Using STARWARP

---

STARWARP has many subcommands and you can often use alternative methods to perform different tasks. In this chapter, alternatives and options are suggested where appropriate; an attempt is also made to show how subcommands are related to each other.

References using the menu system may be *chained*; that is, multiple levels may be separated by periods. For example, M.10.2 is similar to entering M (for MENU), followed by option 10 (for search) and finally option 2 (note that one difference between these two methods is the effect of the **END** key since chained options are not used to back up in a panel hierarchy).

In the following table, each different task is described briefly and a page number is given.

<b>Task</b>	<b>Reference</b>
Allocating a Data Set	page 82
Building a Member List	page 82
Checkpointing the Log	page 83
Comparing Members	page 83
Editing Multiple Members	page 83
Excluding Multiple Members	page 84
Formating Member Data	page 84
Listing Module History	page 84
Locating Data in Dialog Tables	page 85
Managing Data Set Lists	page 85
Managing Member Lists	page 85
Modifying Member Statistics	page 86
Obtaining Assistance	page 86
Processing Multiple Data Sets	page 88
Saving a LISTC table	page 88
Saving a MEMLIST table	page 88
Searching a Data Set	page 88
Searching Multiple Data Sets	page 89
Selecting Members	page 89
Specifying a Member Group	page 90
Triming Dialog Tables	page 90
Updating a Data Set	page 91
Updating Multiple Data Sets	page 91
Validating Data	page 92

## Allocating a Data Set

The **MODEL** command acts as a front end to the IDCAMS subcommand for VSAM data sets or the **CREATE** subcommand otherwise. In both cases, allocation parameters are presented in a prefilled entry assist panel. Use either of the following:

1. **M.3.2** option A
2. **MODEL**

## Building a Member List

The **ML** or **MEMLIST** function is used to build member list tables. **MEMLIST** is always additive; that is, each invocation will add to the current **MEMLIST** unless the **RESET** keyword is specified.

Partial member lists are often built using member group specifications as follows:

- Range** Example: **MEMLIST start:end**
- AB:C** would select all members beginning with AB through all members beginning with C.
  - C:** would select all members beginning with C through the end of the members.
  - :F** would select all members up through members beginning with F.
  - :** would select all members.
- Pattern** Example: **MEMLIST part1/part2**
- AB/C** would select all members whose names contain AB anywhere and C anywhere.
  - ABC/** would select all members containing ABC anywhere.
  - /ABC** would select all members containing ABC anywhere.
- Note that placeholders (% or ? symbols) which match any single character may be used anywhere in a pattern specification.
- Combination** Example: **MEMLIST start\*end**
- AB\*C** would select all members whose names start with AB and contain C anywhere else.
  - ABC\*** would select all members whose names start with ABC.
  - \*ABC** would select all members containing ABC anywhere.
- Note that placeholders (% or ? symbols) which match any single character may be used anywhere in a combination specification.
- Placeholder** Example: **MEMLIST start%end**
- A%?C** would select all members with four character names beginning with A and ending with E.
  - A%** would select all members with two character names beginning with A.
  - ?F** would select all members with two character names ending with F.
- Note that % and ? are interchangeable; these placeholder symbols may be used anywhere in a pattern or combination name.

After the member name position, you may enter operands to filter members by attributes. For example, you may enter "LAST(nn)" where nn is a number of days to filter member age. For source members with ISPF statistics, you may filter on userid with "ID(partial-userid)". For load members, you may filter using linkage attributes such as "RENT", "REUS", "AUTH", "DC", "RMODE24", ... For a complete list of operands, enter **M.ML** or **O.ML**

To refer to all members from the current member list with a subcommand, use = as the member group specification. This has the effect of making the current member group equal to the members in the member list before performing the operation.

**MEMLIST** tables can be saved and recalled. Use the **SAVE** and **MERGE** commands to manage saved member list tables. **MEMLIST** table names may contain one to six alphanumeric characters.

Several MEMLIST short-cut commands are available for source or load libraries: **TODAY**, **WEEK** and **MONTH**; these commands add members which were linked or updated recently (according to ISPF statistics). Another useful command is **MINE** which adds source members saved with your userid in a member's ISPF statistics. If you want a complete list of members in a MEMLIST table, enter **ALL** command or a **6** command.

You may trim member lists with the **X** line command, the **X** primary command or the **EXCLUDE** subcommand. Note that the EXCLUDE subcommand can drop MEMLIST members based on attributes.

The **IF** subcommand has some additional capabilities for selecting members by attributes as compared to the MEMLIST subcommand. For example to select members updated by USERMOD UZ23654, you could enter:

```
IF : USERMOD(UZ23654) THEN(MEMLIST)
```

You may also use the **FIND** subcommand to select members by contents. For example to select members containing the string "ABCX" you could enter:

```
FIND : 'ABCX' THEN(MEMLIST)
```

Note that THEN(NEWML) could have been used in either of the above two examples to "RESET" any current member list.

## Checkpointing the Log

When a single STARWARP subcommand produces more than a fixed number of output lines (by default, 250 lines), STARWARP prompts you with a [checkpoint panel](#). You are expected to decide whether to continue or terminate the processing subcommand.

- To [continue](#) the subcommand, enter **CONTINUE**
- To [terminate](#) the subcommand, press **END**
- To [find data](#) in the log, enter a **F** command; to find the next string, press **RFIND**
- To [change](#) the checkpoint interval, enter **SETLOG**

## Comparing Members

The **COMPARE** subcommand is useful for comparing members. The COMPARE subcommand will actually utilize SuperC, COMPAREX or the Yale Compare program to perform the data comparison as specified during STARWARP installation. Note: the COMPAREX program is very effective in comparing load members since it compares them logically CSECT by CSECT and it marks differences with underscore characters.

Use a **COMP** line command in a member list to compare two members in the same data set (enter the second member name in the "DATA/MSG" field). If the members are in different data sets, enter **O.CO** to obtain a syntax assistance panel where you can specify the comparison data set name (and member).

Note that the syntax assistance panel for COMPARE remembers operands across sessions. Also, if the second data set name is entered without a member name for a PDS, the member name will default to the first member name (this is useful for comparing a member with a backup version in a different data set).

## Editing Multiple Members

The **EDIT** subcommand can process a group of members; when you enter an EDIT subcommand with such a group, each member will be presented to you in edit (in group order) with no intervening panels.

To modify several members with no displays to the terminal, you may request an appropriate initial edit macro. As an example for edit macro DOALL, you could enter **EDIT ABC\* MACRO(DOALL)**

### Excluding Multiple Members

The **X** command and the **EXCLUDE** subcommand should be used to trim MEMLIST tables. The **X** primary command supports an operand (**ALL**, **ABOVE** or **BELOW**) to trim the member table based on the top member shown in the table. It is often effective to sort the MEMLIST table on some criterion (for example, **SORT ID** to sort by userid) and position to a desired member by scrolling, with the **LOCATE** command or the **F** command and then enter **X ABOVE** or **X BELOW** to trim the table. In addition, the **X** line command or the **XX** block line command are also effective at eliminating undesired members after a **SORT**.

The **EXCLUDE** subcommand can filter members using the same attributes as the MEMLIST subcommand (enter **O.EXCLUDE** or **M.EXCLUDE** to see possible attributes). To exclude members, usually the = member group will be specified to examine the members in the current MEMLIST table; however, any member group specification is supported. If a member not in the member list is excluded, its status is not changed.

Note that members may be excluded by attributes (with the **IF** subcommand) or by contents (with the **FIND** subcommand) by specifying a "THEN(EXCLUDE)" operand on the **IF** or **FIND** subcommand.

### Formatting Member Data

The **FIND**, **LIST** and **REPLACE** subcommands format member data identically. **FIND** and **REPLACE** display segments of members; that is, only those lines containing a search string.

Seven formats are supported: **NUM**, **NONUM**, **SNUM**, **LDUMP**, **LBLOCK**, **DUMP** and **BLOCK**. **NUM** is the default for source data; **LDUMP** is the default for load members and **LBLOCK** is the default for VSAM data. When any of these formats is specified for a **FIND**, **LIST** or **REPLACE** subcommand, that format is then used for these subcommands until a different format is chosen.

For load members or VSAM data sets, the **NUM**, **SNUM** and **NONUM** formats do not apply (they are equivalent to the default format). **LDUMP** or **LBLOCK** formats should be used to list or search CSECT data in load members; with these formats, **OFFSET(hexoffset)** and **MODULE(csectname)** are supported. For VSAM index or data components, **DUMP** and **BLOCK** format physical data records.

### Listing Module History

For information on module history, IDR data and COBOL compile options, use the **HISTORY** subcommand. If you need translator (compiler or assembler) information by CSECT, enter **CONTROL TRANSLATOR** to ensure that this information will be provided by default for the remainder of the current session.

1. To obtain all history data for a load member, enter **HISTORY memname**. Note that if **CONTROL NOTRANSLATOR** is in effect, STARWARP will still provide COBOL compile options data.
2. To obtain translator (and COBOL compile option) data only, enter **HISTORY memname translator**
3. To obtain USERDATA (or SYSMOD) information only, enter **HISTORY memname userdata**
4. To obtain ZAP information only, enter **HISTORY memname zap**

You can also search a data set for **TRANSLATOR**, **USERDATA**, **ZAP** or **LKED** (linkage-editor) information. For example, to search a data set for SYSMOD UZ23546, you could enter **HISTORY : userdata(uz23546)**.

Note that you can select members containing specific IDR data by adding a MEMLIST, NEWML or SUBLIST operand to the **HISTORY** subcommand.

## Locating Data in Dialog Tables

Use the **LOCATE** (or **L**) command to position to data of interest in a sorted STARWARP table (this is not supported in the log, LISTA/DDNAME or WORKPAD). LOCATE searches for data in the field the table is sorted on. For example, in a source MEMLIST, the LOCATE command normally positions to a member name. After the command, **SORT ID**, LOCATE would search the userid data field.

## Managing Data Set Lists

STARWARP allows you to create an active data set list similar to ISPF's DSLIST. Unlike DSLIST, however, STARWARP allows you to save these lists across sessions, maintain multiple data set lists and add to or exclude data set names from a list. This facility enables you to maintain customized data set lists. To build an initial data set list, enter the **LISTC** primary command.

This command will display a prompting panel, which allows you to specify a catalog look-up for various data set qualifiers in a single invocation. Try this using your own TSO userid as the high level qualifier to test this facility. After pressing **ENTER** the LISTC processor will display all matching data sets in a scrollable table. Many commands are available to operate on these data sets; enter an **O** line command next to a data set name to get a list of all line commands.

To save this data set list, you could enter **SAVE INTEL** as a primary command. This table would then be saved as member **##INTEL** in your ISPPROF (profile) data set and you can retrieve it in a later session by entering **LISTC INTEL** as a primary command.

Additional invocations of LISTC from an active LISTC table will merge new entries in sorted order in the current list and the **X** line command can be used to drop data sets from the list. After obtaining the data sets desired in the LISTC table, you should again save the list for use in future sessions.

If you want to search disk volumes for data set names, use the **LISTF** function instead. The input required is similar to the LISTC function except that a volume name or a volume name mask must also be specified.

The LISTC/LISTF functions use a combined table; thus, you could have data sets from a volume source as well as a catalog source. The following line commands are commonly used in LISTC/LISTF applications:

<b>=</b>	Repeats the last line command on a different data set
<b>C</b>	Changes to a data set
<b>DEL</b>	Deletes a data set
<b>IDC</b>	Displays the result of an IDCAMS LISTC ENTRY ALL for a data set
<b>INFO</b>	Displays VSAM or non-VSAM data set statistics and space use for a data set
<b>ML</b>	Changes to a data set and builds a MEMLIST of all members
<b>REN</b>	Renames a data set
<b>U</b>	Changes to a data set and displays USAGE information
<b>X</b>	Drops a data set from the table

Some very powerful primary commands make the LISTC/LISTF function a file management facility. From the primary command line, **FIND** or **REPLACE** may be entered to activate a global function which will search or update all partitioned data sets in the table.

## Managing Member Lists

To obtain a member list containing all members in a data set similar to an ISPF member list, enter **ALL** as a primary command. An equivalent command is **ML :** since the colon specifies no start or end range and all members would be included. If this were a load library, the processing to build the initial member list display would take slightly longer than a source member list because each member is read to obtain the linkage-edit date.

## STARWARP User Guide

To refer to all members from the current member list with a subcommand, use = as the member group specification. This has the effect of making the current member group equal to the members in the member list before performing the operation.

MEMLIST tables can be saved and recalled. Use the **SAVE** and **MERGE** commands to manage saved member list tables. MEMLIST table names may contain one to six alphanumeric characters.

Member ranges are specified by using a starting member name followed by a : and then an ending member name. To limit the member list displayed, a selected range may be used as follows: **ML ABC:BCD RESET**. This example would request a member list to be built starting with members named ABC... and ending with members named BCD... The RESET keyword indicates that the member list is to be discarded and rebuilt with the specified member group; otherwise, MEMLIST is an additive process.

It is also possible to build a member list of members whose names contain a common string. For example, **ML IST/** would build a member list containing all members with the character string "IST" somewhere in their name. Further combinations are possible which limit the member list to members whose names begin with a common string such as "IEB". In this case, the command would be **ML IEB\***.

Member names are not the only potential method for obtaining member lists. The criteria varies with over 80 selectable options from linkage-edit date and module attributes to SSI and IDR data contained within the members. This flexibility expands the ease in identifying the members you want to manipulate. See the User Reference Manual, Appendix A for member group specification rules and the MEMLIST function for additional information on filtering members.

A simple approach to managing members consists of building a member list based on the last update or linkage-edit date of members in a library. For example, you can enter either **ML : WEEK** or **WEEK** to build a list of members modified or added in the last week. The commands **TODAY** and **MONTH** can also be used for simple date filtering. Another approach for source members is the **MINE** command, which builds a member list of all members whose ID is equal to your USERID.

MEMLIST supports line commands with up to four character names. A line command is entered in the "CMD" column next to a member to be manipulated. Multiple line commands may be entered on the same member list screen together. Below are several commonly used member line commands:

<b>B</b>	Browses a member using ISPF services
<b>COMP</b>	Compares a member with another member which can be in another data set
<b>CS</b>	Builds an ISPF table of all CSECT and ENTRY symbols in a module
<b>E</b>	Edits a source member using ISPF services
<b>MAP</b>	Maps the CSECT structure of a load member
<b>PR</b>	Prints a member
<b>S</b>	Defaults to Edit in a source data set; otherwise, it defaults to Browse for load
<b>SUB</b>	Submits a member for background processing
<b>X</b>	Drops a member from the table
<b>XREF</b>	Displays intra-module references

## Modifying Member Statistics

The recommended way to modify a source member's ISPF statistics is with the **ATTR** line command from a MEMLIST table. You are presented with a prefilled entry assist panel, which reflects the member's current statistics. Individual attribute entries can be changed as desired; and if a member has no ISPF statistics, they can be added.

## Obtaining Assistance

There are many forms of assistance available in STARWARP.

- For an index to tutorials on all STARWARP topics, enter **INDEX**
- For a tutorial on the current subcommand or function, press **HELP**
- For a list of topics managed in a CUA action bar, tab or position the cursor over the area and press **ENTER**
- For a functional description of a CUA action bar menu item, position the cursor next to the item and press **HELP**
- For a functional description of any other area in a table panel, position the cursor over the area and press **HELP**
- For a syntax assist panel for a item in a CUA action bar, position the cursor next to the item and press **ENTER**
- To obtain an explanation just after receiving warning (PDSnnnW) or error (PDSnnnE) messages, enter **?**
- To obtain an explanation of any STARWARP message (PDSnnn) in the log, position the cursor over the message identifier (the "PDSnnn") and press **RCHANGE** (this is normally PF key 6 or 18).
- For a directory of available commands, enter **O** (for OPTIONS) as a line command or primary command. This facility will provide a syntax assist panel for the command chosen. As a short-cut you may chain options; for example, **O.LI** requests a syntax assist panel for the LIST subcommand.
- For a menu of available subcommands and functions, enter **M** (for MENU) as a line command or primary command. This facility will provide a entry assist panel for the command chosen. As a short-cut you may chain options; for example, **M.LI** requests an entry assist panel for the LIST subcommand.
- From either a syntax assist or entry assist panel, for a subcommand tutorial, press **HELP**

## Processing Multiple Data Sets

Global commands may be used to process multiple data sets in a LISTA/DDNAME, LISTC/LISTF or WORKPAD table. The **GLOBAL** command allows any STARWARP subcommand to be used on a global basis (against each data set in a table) and the other commands are understood to have a global scope.

The following commands are global in LISTA/DDNAME, LISTC/LISTF and WORKPAD tables.

<b>FIND</b>	See <b>Searching Multiple Data Sets</b> on page 89 for more information.
<b>MODEL</b>	Changes to each data set in the table and performs a <b>MODEL</b> command to create a data set based on the data set in the table. <b>MODEL</b> may be used as a front end to the <b>CREATE</b> subcommand or the <b>IDCAMS</b> subcommand for VSAM data sets.
<b>REPLACE</b>	See <b>Updating Multiple Data Sets</b> on page 91 for more information.
<b>SEEK</b>	Changes to each data set in the table and checks for a member.
<b>WHOHAS</b>	Performs an <b>ENQUEUE</b> check on all data sets in the table.

## Saving a LISTC table

To build a LISTC/LISTF table, enter **LISTC** to obtain data sets from a catalog source or enter **LISTF** to obtain data sets from a volume (VTOC) source. A prompt panel will ask you to specify one or more data set name masks; the LISTF prompt panel also requires either a volume name or a volume name mask. Once you have entered the LISTC/LISTF function, you can add to the table by entering LISTC and/or LISTF to search for additional data sets.

You may trim the table with the **X** line command. One very effective way to trim the LISTC/LISTF table, is to first sort the table on some criteria (such as **SORT TYPE** to sort by DSORG), position to any desired point in the table (by scrolling or the **F** command), and entering a primary command such as **X ABOVE**

You can name the table with a command such as **ID B27** for table B27 and enter **SAVE** to save the table in your profile data set. In a later session, this table may be retrieved by entering **LISTC B27**, and data set statistics can be updated by entering the **REFRESH** command.

## Saving a MEMLIST Table

MEMLIST tables can be saved and restored. You may want to use saved MEMLIST tables to checkpoint work in progress (perhaps to save where you were working at the end of a day) or to provide a list of members for another application (one example might be a batch process performing **VERIFY : MEMLIST** commands to select members with errors and saving the list for interactive review).

If you want to save a MEMLIST table for later reference, enter a command such as **SAVE mydata**. Names of saved MEMLIST tables may contain one to six alphanumeric characters.

To recall a previously saved table, enter a command such as: **MERGE mydata**

Note that this command will add (or merge) members into any existing MEMLIST table. You may add a **RESET** operand on the command to nullify the current MEMLIST table before adding the saved members.

## Searching a Data Set

The **FIND** subcommand can find members containing a search string and take a conditional action. For example, if all members containing the string '//STEPLIB' were to be copied to a different data set and deleted you could perform the following:

1. **FIND '//steplib' display then(memlist)** (selects the proper members)

2. **COPY \* new.data.set new** (copies the selected members into a clone of the current data set)
3. **DELETE \*** (deletes the selected members)

Note the use of the "display" operand on the FIND subcommand above; it should be added to a FIND subcommand if FIND results should be displayed in the log in addition to taking a conditional action (such as "then(memlist)"). DISPLAY is particularly useful if you are selecting members for inclusion in a MEMLIST or SUBLIST.

## Searching Multiple Data Sets

To find data in multiple data sets, it is suggested that you use the global **FIND** command. First, build a LISTA/DDNAME, LISTC/LISTF or WORKPAD table containing the data sets to be searched. All data sets should be either partitioned or non-partitioned.

In a LISTC/LISTF table, the global FIND command bypasses non-partitioned data sets. If you actually want to search non-partitioned data sets, enter a command similar to the following:

```
GLOBAL FIND 'string' ...
```

Finally, from the command line in that table, enter a **FIND** command to transit to the global FIND syntax assist panel. If all of the data sets are partitioned, add a member group specification such as **:** to search all members. Otherwise, be sure to omit this positional parameter.

## Selecting Members

The **FIND** subcommand filters members by contents and the **IF** subcommand filters members by attributes. For either of these subcommands, the first operand is a member group (see "Specifying a Member Group" below).

For the IF subcommand, desired attributes should be entered next (for example, you could enter RENT, AMODE24 or AUTH for load members). All specified conditions must be met to select a member; these are and conditions. Finally, you should specify a action to take if true (with "THEN(command)") and/or an action to take if false (with "ELSE(command)"). Normally, THEN(SUBLIST) is specified so that following subcommands can be specified with operands on the members selected. For example,

```
IF copy* ID(myid) LAST(7) THEN(sublist)
PRINT * SYSOUT(a) DEST(rmt21)
COPY * other.data.set
```

could be used to print selected members updated recently by "myid" and copy them to another data set

For the FIND subcommand, the character string being searched for is entered after the member group specification. It is specified as a delimited string with 1 to 32 characters; the first character is used as the final delimiter and the delimiter must not be used within the string itself. In practice, the single quote (') or double quote (") are used as FIND string delimiters. If **X** is used as the string delimiter, the data is assumed to be hexadecimal digits and the string may contain 2 to 64 hexadecimal digits.

Following the string specification, other parameters may be in any order. FIND boundary conditions (WORD, SUFFIX or PREFIX), FIND limits (MAXIN, MAXOUT, MAXFIND, MODULE, OFFSET, ...) and FIND list format (NUM, NONUM, SNUM, LDUMP, LBLOCK, DUMP or BLOCK) may be specified.

To select members with a given string and certain conditions, you should specify an action to take if found (with "THEN(command)") and/or an action to take if not found (with "ELSE(command)"). FIND limits (MAXIN, MAXOUT, MAXFIND, MODULE, OFFSET, ...) and FIND list format (NUM, NONUM, SNUM, LDUMP, LBLOCK, DUMP or BLOCK) may be specified. Normally, THEN(SUBLIST) is specified so that following subcommands can be specified with operands on the members selected. For example,

```
FIND COPY* 'PGM=IEBCOPY' DISPLAY THEN(SUBLIST)
PRINT * SYSOUT(A) DEST(RMT21)
```

**DELETE \***

could be used to print JCL members which invoke IEBCOPY and delete them.

## Specifying a Member Group

All subcommands, which allow a member group specification (except for DISPLAY and PATTERN), also modify the current member group. These subcommands may specify lists of members and/or member groups. If a list of members is specified, it must be enclosed in parentheses and individual items in the list must be separated with one or more blanks and/or a comma.

Individual member group items may be specified using the following entry types:

- Range** Example: **MEMBERS start:end**  
**AB:C** would include all members beginning with AB through all members beginning with C.  
**C:** would include all members beginning with C through the end of the members.  
**:F** would include all members up through members beginning with F.  
**:** would include all members.
- Pattern** Example: **MEMBERS part1/part2**  
**AB/C** would include all members whose names contain AB anywhere and C anywhere.  
**ABC/** would include all members containing ABC anywhere.  
**/ABC** would include all members containing ABC anywhere.  
Note that placeholders (% or ? symbols) which match any single character may be used anywhere in a pattern specification.
- Combination** Example: **MEMBERS start\*end**  
**AB\*C** would include all members whose names start with AB and contain C anywhere else.  
**ABC\*** would include all members whose names start with ABC.  
**\*ABC** would include all members containing ABC anywhere.  
Note that placeholders (% or ? symbols) which match any single character may be used anywhere in a combination specification.
- Placeholder** Example: **MEMBERS start%end**  
**A%?C** would include all members with four character names beginning with A and ending with E.  
**A%** would include all members with two character names beginning with A.  
**?F** would include all members with two character names ending with F.  
Note that % and ? are interchangeable; these placeholder symbols may be used anywhere in a pattern or combination name.
- Asterisk** Example: **MEMBERS \***  
A single asterisk may be used to refer to the current member group.
- Equal** Example: **MEMBERS =**  
An equal symbol may be used to indicate that the current member group is to be set equal to the list of members in the current MEMLIST before passing control to the subcommand.

The current member group is maintained across CHANGE subcommands until a subcommand specifies a different member group. An example which includes several of the elements above would be

**MEMBERS (AB:C, ABC/FX?Z, CO%Y\*, A%?E)**

A MEMBERS subcommand such as the above would cause the associated members in the data set to be listed. To convert this to a sublist (a list of actual member names), you could follow this with a subcommand such as **SUBLIST \* ALIAS** (where ALIAS indicates that associated members should also be included).

## Trimming Dialog Tables

All STARWARP dialog tables can be trimmed with the **X** primary command. The supported operands (ABOVE, BELOW and ALL) trim the table in relation to the top line in the current table display. For those tables that support line commands, the **X** line command or a **XX** block line command pair can be used to drop individual table lines.

**REMOVE** and **EXCLUDE** facilities are available for all function tables. From a log or MEMLIST table, the **EXCLUDE** subcommand may be used to trim members from the MEMLIST table. The **EXCLUDE** subcommand supports the same attribute filtering operands that are available on a MEMLIST subcommand. From any table, the **REMOVE** command may be used to trim all table elements that match a search string (or do not match a search string with the **NOT** keyword). The **REMOVE** command supports the same operands as available on the **F** command with the exception of the directional keywords (**NEXT**, **FIRST** and **LAST**) because the **REMOVE** command always processes an entire table.

For those tables that can be sorted, it is often very effective to sort the table in a particular order, for example with load members you might enter **SORT DATE** (to sort by linkage-edit date). Position to some point in the table by scrolling, with the **F** command or with the **LOCATE** command and enter an **X** primary command to trim the entire table. Finally, you should probably enter **SORT** (with no operands) to sort the table back into its default order.

## Updating a Data Set

To update multiple members in a data set, it is suggested that you use the **REPLACE** subcommand; it performs an update in place. Enter a command similar to the following: **REPLACE memgroup 'datanow' 'datanew'** or if you wish to use delimited strings, enter a command similar to the following:  
**REPLACE memgroup /datanow/datanew/**

For the menu system, enter

<b>M.REPLACE</b> option:	<b>Member or member group</b>	==>	<b>memgroup</b>
	<b>Find data</b>	==>	<b>datanow</b>
	<b>Replacement data</b>	==>	<b>datanew</b>
	<b>Update data on disk</b>	==>	<b>NO</b>

By default, the above subcommand would only perform a trial update. After examining the output from the **REPLACE** subcommand, you should recall the subcommand and add the **WRITE** keyword or change the **M.REPLACE** panel to

<b>Update data on disk</b>	==>	<b>YES</b>
----------------------------	-----	------------

to actually update the members.

To enter hexadecimal data in the **REPLACE** subcommand, a special format with X delimiters is used:

**REPLACE memgroup XhexbeforeXhexafterX.**

## Updating Multiple Data Sets

To update data in multiple data sets, it is suggested that you use the global **REPLACE** command. First, build a **LISTA/DDNAME**, **LISTC/LISTF** or **WORKPAD** table containing the data sets to be updated. All data sets should be either partitioned or non-partitioned.

In a **LISTC/LISTF** table, the global **REPLACE** command bypasses non-partitioned data sets. If you actually want to update non-partitioned data sets, enter a command similar to the following:

**GLOBAL REPLACE 'string1' 'string2' ...**

Finally, from the command line in that table, enter a **REPLACE** command to transit to the global **REPLACE** syntax assist panel. If all of the data sets are partitioned, add a member group specification such as **:**. Otherwise, be sure to omit this positional parameter. If you do not enter the "WRITE" keyword, **REPLACE** will only perform a trial update; you should then repeat the **REPLACE** command with the **WRITE** operand to actually update the data sets.

## Validating Data

To validate members or an entire data set you should use the **VERIFY** subcommand. For a PDS, if the member group name is : (a colon), the entire data set is checked and error members are identified; otherwise, members are checked individually and statistics are generated for each member.

To check individual members, enter **VERIFY A:B** (where "A:B" is the group of members to be checked).

To validate an entire PDS, enter **VERIFY :**

For the menu system, enter

**M.VERIFY** option: **Member or member group ==> A:B**

You might want to add the **MEMLIST** option to select members with errors.

By default for VSAM data sets, **VERIFY** will perform an IDCAMS **VERIFY** operation on the data set followed by an IDCAMS **EXAMINE** for KSDS data sets. Then, the **VERIFY** subcommand will access a KSDS data set by key and in addressed mode to verify that all records are accessible.

## Edit Macros

---

Several edit macros and a CLIST are provided for your use during an edit session to assist you in recovering from edit problems and in using edit effectively. Note: during an edit session, you may use the **O** edit macro for help on the STARWARP edit macros; to get help on a specific edit macro, you can also enter **O ATTRIB**

### ATTRIB Edit Macro

This edit macro is used to display a member's ISPF statistics as notes in an edit session on a line just after the cursor. If no operand is entered, ATTRIB formats attribute information on the current member; otherwise, the operand is taken to be a member group name and attribute information is formatted for all corresponding members in the current data set.

Output from the STARWARP ATTRIB subcommand is displayed as edit notes, which are not saved when edit terminates. However, you can enter a **MD** line command over any note line to convert it to a data line.

To display attributes for a member group, enter a command like: **ATTRIB pds/ux**  
An ATTRIB command with no operand would produce the following edit notes:

=NOTE=	ISPF Stats:	VER.MOD	CREATED	LAST MODIFIED	SIZE	INIT	ID
=NOTE=	PDS4ED3	01.00	93/03/04	93/03/04 7:38	28	28	SER07

### CUT and PASTE Edit Macros

The CUT and PASTE edit macros provide a simple method of moving data from one edit session to another. The CUT macro saves lines from a edit session into a temporary ISPF table and the PASTE macro places these lines into another edit session or the same session.

The CUT macro can combine lines with previously saved lines. The PASTE macro can place these lines into one or more edit sessions. The edit sessions can be on different screens and/or edit sessions started at a later time provided you do not exit ISPF.

The CUT macro works with the edit line commands for MOVE/COPY in their various forms (M, MM Mnn, C, CC, Cnn) to specify which group of lines to save. The CUT macro supports several optional operands:

1. The operands P or PROMPT, D or DEFAULTS provide a prompting panel like the following sample. The H or HELP operand can be used to provide TUTORIAL panels.
2. The table name will use a default value if it is not provided or the name can be changed in the prompting panel to any desired two-character name.
3. The operand A or ADD for the CUT macro specifies that the selected lines are to be added to already saved lines if any. The operand R or REPLACE for the CUT macro specifies that the lines selected are to replace or overlay the lines already saved. The default, if not changed, is ADD.

Following is the prompting panel for the CUT edit macro from a **CUT PROMPT** command in edit.

```

----- CUT prompt panel -----
COMMAND ==>

Action      ==>>>          ADD, REPLACE, DEFAULTS, or blank for normal action

CUT table name==>>      any two character name for this one time
                        blank for default

CUT size limit==>> 1000  any number 100 to 99999

    ADD will add CUT lines to the end of previous CUT lines if any.
    REPLACE will clear any previous CUT lines and replace them with
        newly CUT lines.
    DEFAULTS will display a panel to set defaults.
    blank will request the default action.

```

**Figure 77. Sample CUT Prompt Panel**

The PASTE macro works with the edit line commands After or Before (A or B). This specifies where all lines previously saved are to be inserted.

The operand Z or ZERO specifies that the table is to be cleared after the lines have been inserted.

The operand K or KEEP specifies that the table is not to be cleared and that the saved lines are to be kept for future PASTE requests or to be added to by other CUT requests. The default, if not changed, is ZERO.

The use of the operands ADD, REPLACE, ZERO, and KEEP define the mode for the CUT and PASTE macros. The most common mode could be called the "gather mode" with operands ADD and ZERO. This is where several groups of lines are gathered from here and there and finally pasted into one member.

The other main mode could be called the "distribution mode" where one group of lines is CUT and then placed in several members or in several spots in one member. This uses the operands REPLACE and KEEP. The defaults can be set for the mode you like most.

Either mode works when working in a "one for one" basis where each CUT is followed by one PASTE. If you set the defaults to ADD and KEEP you will have to keep track when to override with either REPLACE or ZERO or you will accumulate lines.

Following is the prompting panel for the PASTE edit macro from a **PASTE PROMPT** command in edit.

```

----- PASTE prompt panel -----
COMMAND ==>

Action      ==>>>          ZERO, KEEP, DEFAULTS, or blank for normal action

CUT table name==>>      any two character name for this one time
                        blank for default

    ZERO will delete previously CUT lines after the PASTE operation.
    KEEP will keep previously CUT lines after the PASTE operation to
        be used by a later PASTE or added to by later CUT operations.
    DEFAULTS will display a panel to set defaults.
    blank will request the default action

```

**Figure 78. Sample PASTE Prompt Panel**

## EQUAL Edit Macro and REEQUAL CLIST

The EQUAL macro provides for setting up and executing edit finds and changes. The FIND/CHANGE options are saved for reuse in later edits. If the FIND/CHANGE options are invalid or can not be serviced, the user can reinvoke EQUAL to correct or continue the request.

This macro can be used as an initial MACRO or can be called directly. If used as an initial MACRO, a set of options control if and when prompting is to be provided. Prompting can be forced with the RESET option or by using the REEQUAL TSO clist.

If called directly, one of the following options may be specified:

- C/CHANGE - to override the previously set CHANGE processing option without prompting. This override is for a single request.
- F/FIND - to override the previously set FIND processing option without prompting. This override is for a single request.
- N/NOPROMPT - execute current options without prompting.
- R/RESET - to force prompting if EQUAL is used as an initial macro.
- H/HELP - to display TUTORIAL panels.
- P/PROMPT - to display a prompt panel for the above options.

Following is the prompting panel for the EQUAL edit macro from an **EDIT member MACRO(EQUAL)** subcommand in STARWARP.

```

----- EQUAL MACRO Prompt -----
OPTION ==>

Process option ==> FIND    FIND/CHANGE

FIND and CHANGE TO Strings:  (use QUOTES if necessary for imbedded blanks)
FIND    ==>
CHANGE ==>

Column range (optional)  See Edit TUTORIAL on FIND for more information
Starting  ==>              (blank or 1-255, not to exceed record length)
Ending    ==>              (blank or 1-255, not to exceed record length)

Embedded search options ==>          (blank, CHARS, PREFIX, SUFFIX, WORD)

Control options:
EXCLUDE ALL before search    ==> YES      (Yes or No)
Set to RCHANGE after find    ==> NO       (Yes or No) for Find processing
Set to AUTOSAVE OFF PROMPT   ==> NO       (Yes or No) for Change processing

Options used when macro used as initial macro
Prompt before any action     ==> YES      (Yes or No)
Display after changes        ==> YES      (Yes or No)
Action if change errors      ==> DISPLAY  (DISPLAY, CANCEL, SAVE)

```

**Figure 79. Sample EQUAL Prompt Panel**

EXCLUDE ALL before search of YES will direct the EQUAL macro to first exclude all edit lines before processing the FIND or CHANGE. This will allow the matched or changed lines to stand out.

The EQUAL macro can setup the RCHANGE edit function when used in a find mode. PF6 is normally assigned a RCHANGE function. By specifying YES to Set to RCHANGE after find and providing a CHANGE TO string the PF5 and PF6 keys can be used to make selected changes.

## STARWARP User Guide

When using the CHANGE function you may want to verify that changes are appropriate. Set the AUTOSAVE OFF PROMPT option to YES to prevent the changes from taking effect if the END key is used by accident. Note that this setting may be retained in the edit profile.

This macro can be used for processing a group of members without intervention. In this case, no prompting is necessary and Prompt before any action and Display after changes can be set to NO. Otherwise, specify YES on either as needed. Prompt before any action can be reset to YES by using the REQUAL TSO CLIST at any time a TSO command can be entered.

The Action if change errors option controls the edit action when processing a group of members with PROMPT and DISPLAY set to NO and an error occurs. Use CANCEL to continue processing by skipping the updating of members with change errors. Use SAVE to continue processing and save any changes even with errors. Use DISPLAY to delay processing while reviewing any errors resulting from the change operation.

The REEQUAL CLIST provides for setting or resetting processing options for the EQUAL macro. This CLIST can be called before or between edit sessions that use the EQUAL macro. If the EQUAL processing options are setup for no prompting and the EQUAL macro is being used as a initial macro, the RESET option of this CLIST can be used to force prompting at the next edit session. If no options are used, the REEQUAL CLIST will display the EQUAL processing options panel.

The REEQUAL CLIST can be called with the following options:

- R/RESET - to force prompting if EQUAL is used as an initial macro.
- H/HELP - to display TUTORIAL panels.
- P/PROMPT - prompt panel for the above options.

### PACKOFF Edit Macro

This edit macro can be used to unpack members using ISPF edit and a **PACK OFF** statement. It performs its operation and ends so you do not actually enter edit.

The following example could be used to find all packed members and then unpack them:

```
IF : packed then(sublist)
EDIT * macro(packoff)
```

### PACKON Edit Macro

This edit macro can be used to pack members using ISPF edit and a **PACK ON** statement. It performs its operation and ends so you do not actually enter edit.

The following example could be used to find all unpacked members and then pack them:

```
IF : nopacked then(sublist)
EDIT * macro(packon)
```

### PDSEDSN Edit Macro

This edit macro is used to initiate a STARWARP session on a data set name in edit. The name of the data set must be fully qualified and it must be cataloged. To use PDSEDSN from an edit session, enter **KEYS** as a primary command and assign **PDSEDSN** to one of the PF keys. Exit from the KEYS function with an END command.

Position the cursor over any data set name and press the PF key assigned to PDSEDSN. This will invoke STARWARP for that data set. When you are finished, terminate STARWARP and you will be placed back in the edit session.

### TRAP Edit Macro

This edit macro is used to display output from a TSO command as notes in an edit session on a line just after the cursor. This CLIST uses TSO/E SYSOUTTRAP facilities so it can only capture data from TSO commands, which

use PUTLINE output. Output from the TSO command is displayed as edit notes, which are not saved when edit terminates. However, you can enter a **MD** line command over any note line to convert it to a data line. For example, to capture output from the DSAT command, enter a command like:

**TRAP DSAT sdsf**

The above command would produce the following edit notes:

```
=NOTE= SERIAL  ALLOC  FREE  EX  DSORG  -DCB  ATTRIBUTES-  CR.  DATE  -DSNAME-
=NOTE= SER007      6      2  1  A-PS  FBA  23474  121 03/04/92 SER7.S#
=NOTE= SER006     15      4  1  A-PO  FB   13680   80 05/03/93 SER7.S5
=NOTE=              21      6                          2 DATA SETS
```

### VOLUME Edit Macro

This edit macro is used to display the current data set and volume name as a note in an edit session on a line just after the cursor. This edit macro may be used with uncataloged data sets to verify that the correct data set is being edited. An edit note is not saved when edit terminates; however, you can enter a **MD** line command over any note line to convert it to a data line.

To display the data set and volume name for the current edit data set, enter: **VOLUME**

The above command would produce the following edit note:

```
=NOTE= DATA SET 'SER07.PDSE311.PANELS' IS ON VOLUME TSO006
```



## Appendix A. IBM Batch Utilities Comparison

IBM Utility	Function	STARWARP	Description
AMBLIST	List load member attributes	ATTRIB	List all attributes of a module including the linkage-edit date
"	"	CSECTS	Display CSECTS in a selectable table
"	"	HISTORY	List IDR data (zap, userdata and translator information)
"	"	MAP	Map internal CSECT structure or rebuild linkage-edit statements
"	"	MEMLIST	Display members in a selectable table
"	"	XREF	Cross-reference CSECT and EXTERNAL symbols
IEHLIST	VTOC list utility	LISTF	Data sets in a selectable table with statistics Delete or rename data sets Catalog or uncatalog data sets
"	"	LISTV	List attributes and available space on disk volumes
"	"	USAGE	Map the Format 1 DSCB in the log (with the ALL operand)
IEWL	Linkage-edit	ATTRIB	Alter most load member linkage attributes
IDCAMS	IDCAMS Utilities	IDCAMS	Format IDCAMS commands using a menu approach with results in the log
"	"	LISTC	List data sets from the MVS catalog in a selectable, sortable table
"	"	IDC	LISTC line command to display an IDCAMS LISTCAT with ENTRY and ALL
"	"	INFO	LISTC line command to display VSAM space use and attributes
"	"	VERIFY	Perform IDCAMS VERIFY and EXAMINE, also validate data



## Appendix B. ISPF Utilities Comparison

ISPF	Function	STARWARP	Description
=1	Browse	BROWSE	Browse a member, data set or VSAM file
=2	Edit	EDIT	Edit a member, data set or VSAM file (SAVE is not supported for VSAM)
=3.1	Library Applications	ATTRIB	List or alter member statistics or module attributes
"	"	BROWSE	Browse members
"	"	VERIFY	Validate members
=3.2	Data Set Applications	CREATE	Create a new data set
"	"	DSNAME	Display summary data set information
"	"	MODEL	Create a new data set with overtyping (including VSAM)
"	"	USAGE	Display additional data set information
"	"	USAGE ALL	Display detailed data set information
=3.4	Data Set List	PLIST	List a saved data set table from anywhere in ISPF
"	"	LISTC	List data sets using a catalog source
"	"	LISTF	List data sets using a volume source
"	"	MASK	Add selected data sets to a LISTC/LISTF table
"	"	LISTA	List data sets allocated to a TSO session by attributes
"	"	DDNAME	List data sets allocated to a TSO session by DDNAME mask
"	"	WORKPAD	Display selected data set names and/or commands in a selectable table
=3.5	Reset Stats	ATTRIB	List or alter member statistics or module attributes
=3.8	Command Table	CMDBL	Display and modify an ISPF command tables
=3.13	Compare Members	COMPARE	Compare two members using SUPERC, COMPAREX or COMPARE\$
"	"	COMPDIR	Compare multiple members for similarities or differences
=3.14	Search for	FIND	Search members and perform conditional processing

Note that STARWARP offers functions equivalent to most of the ISPF functions through its menu system. For example, if you wanted to use the equivalent of ISPF 3.4 in STARWARP, you could enter **M.3.4** or **3.4**. If you actually wanted to perform an ISPF function and return to STARWARP, you could stack an ISPF session with a command like the following: **ISPF 3.4**.



---



---

## GLOSSARY

---

**Alias member**

A member whose directory entry has an indicator value which indicates that the member entry is an alternate name for a main member; a main member and its aliases are associated with the same recorded data in a data set.

**Apparent alias member**

Data in a PDS which is pointed to by more than one main directory entry.

**Associated members**

Members with the same TTR address. For a main member, the associated members could be alias members or an apparent alias member; for an alias member, the associated members could be other aliases or a main member.

**Batch mode**

A mode of STARWARP operation. In batch mode, all user communication is performed with PUTGET, GETLINE and PUTLINE TSO services. ISPF services are available. In this mode of operation, STARWARP operates under control of the batch Terminal Monitor Program (the TMP, or IKJEFT01).

**BLDL**

A BPAM function which is used to check for the existence of members.

**BPAM**

An acronym for Basic Partitioned Access Method; BPAM is actually very similar in use to BSAM which is for sequential data. STARWARP often uses BPAM for member check operations and always uses BPAM for directory updates and reading PDSE data sets.

**BSAM**

An acronym for Basic Sequential Access Method; STARWARP uses BSAM for several utility functions.

**CCHHR**

An acronym for Cylinder, Cylinder, Head, Head, Record which is a ten byte hexadecimal disk address relative to the start of the volume.

**Command**

A request for an operation. For example, STARWARP is a command and HISTORY is a STARWARP subcommand. The distinction between a command and a subcommand is not often important.

**Compress**

A process that removes deleted members from a PDS. Disk space formerly occupied by these deleted members is made available for new members. In a PDSE data set, space from deleted members is managed dynamically; you can not compress a PDSE.

**Deleted member**

Member data which is not pointed to by any directory entry; deleted members remain in a PDS until the data set is compressed. In a PDSE data set, space from deleted members is managed dynamically; you can not restore deleted members.

**Directory**

A segment of a PDS which is a sequential data set with eight byte keys and 256 byte data records containing pointers to data in the PDS member portion of the data set. For a PDSE, PDS directory and member data is created in a virtual PDS data set.

**Directory entry**

A logical entry in a PDS or PDSE directory which points to its associated data in the member data; it consists of an eight character member name field and one or more three byte relative address (TTR) fields.

**DS1LSTAR**

A pointer in the data set control block (DSCB) which indicates the last used disk address for a data set. This pointer is updated after a member is added to a PDS data set or after a PDS is compressed.

**EXCP**

An acronym for EXecute Channel Program. This access method is used extensively by STARWARP to read an entire disk track with a single operation.

**Function**

A request for an ISPF (or ISPMODE) operation. Most ISPMODE functions support their own commands.

**ISPMODE**

A mode of STARWARP operation. With ISPMODE, STARWARP operates as an ISPF dialog and normal ISPF services (HELP, TSO, SPLIT, SWAP,...) as well as all STARWARP services are available.

**Line command**

A command entered on a table line in the CMD field. Line commands may be up to four characters long.

**Line mode**

A mode of STARWARP operation. In line mode, all user communication is performed with PUTGET, GETLINE and PUTLINE TSO services. No ISPF services are available.

**Load library**

A PDS or PDSE which has record format U and is commonly used to contain executable modules.

**Main member**

A non-alias directory entry and its associated recorded data.

**Member**

Data in a PDS or PDSE, which is pointed to by one or more directory entries. Members may be alias or main members. Deleted members, which have no associated directory entries. For a PDSE, space from deleted members is managed dynamically; you can not restore deleted members or compress a PDSE.

**Module**

A member contained in a load library. Also known as a load module or load member.

**Orphan member**

A member marked as an alias, which does not have an associated main member.

**PDS or Partitioned data set**

PDS. A data set with fixed, variable or undefined format, which is commonly used as a *library* for related types of information. Partitioned data sets contain two segments of data: a directory of information and member data. This type of library is requested as DSNTYPE(PDS) in JCL or a TSO ALLOCATE.

**PDSE or Partitioned Data Set Extended**

PDSE. A SMS managed data set, which is logically similar to a PDS. This type of library is requested as DSNTYPE(LIBRARY) in JCL or on an ALLOCATE command. A PDSE can be accessed by BPAM or BSAM access methods to obtain a virtual data set that looks like a PDS.

**Primary command**

A subcommand or command entered from the command line of a panel.

**QSAM**

An acronym for Queued Sequential Access Method; STARWARP uses QSAM for several utility functions.

**Source library**

A PDS or PDSE data set that has fixed or variable format data; source libraries are normally used to contain non-executable data.

**STARWARP**

The premier year 2000 workbench for MVS environments. A multipurpose ISPF dialog and TSO command processor which manipulates data sets, members and data records.

**Subcommand**

A request for an operation that is within the scope of work requested by the previously issued command. STARWARP is a command; HISTORY is a STARWARP subcommand. The distinction between a subcommand and a command is not often important.

**TTR**

An acronym for Track, Track, Record (a 1 to 6 digit hexadecimal disk address relative to the start of the data set); this type of address is stored in the member directory entry to indicate the start of a member.

## Index

### ?

? Command, 87

### =

=, 16

## A

Asterisk member specification, 90  
ATTRIB Edit Macro, 93  
ATTRIB Subcommand, 87

## B

Background, i  
Background JCL, 78, 79  
Batch JCL, 78, 79  
Batch mode, 10, 77  
Batch utilities, 99

## C

Calling STARWARP  
  Batch, 78  
  ISPF, 10  
  ISPMODE in Batch, 78  
  Single subcommand, 10  
chained commands, 9  
Chained commands, 13  
Change Man  
  HISTORY Preload, 75  
  MAP LCT Option, 73  
  MEMLIST Option, 72  
Change Management, 71  
CMNLOAD, 75  
Combination member specification, 90  
commands, 53  
Commands  
  ?, 87  
  chained, 9, 13, 101  
  EXCLUDE, 80, 84, 91  
  F, 79  
  HELP, 87  
  INDEX, 87  
  L, 80, 85  
  LOCATE, 80, 85  
  M, 13  
  MENU, 13  
  MODEL, 82  
  OUTPUT, 80  
  restricted, 11, 12  
  SORT, 80

X, 80, 84, 91

COMPARE Subcommand, 83  
COMPCHK, 44  
COMPRESS global command, 88  
CSECTCHK, 46  
current data set, 10  
CUT Edit Macro, 93

## D

DATECALC, 31  
DATEDIFF, 35

## E

Edit macro  
  ATTRIB, 93  
  CUT, 93  
  EQUAL, 95  
  PACKOFF, 96  
  PACKON, 96  
  PASTE, 93  
  PDSEDSN, 96  
  REEQUAL, 95  
  TRAP, 97  
  VOLUME, 97  
EDIT Subcommand, 84  
Entry assist panels, 8, 13, 17  
EQUAL Edit Macro, 95  
Equal member specification, 90  
EUROCALC, 32  
EXCLUDE Command, 80, 84, 91  
EXCLUDE Subcommand, 91  
expanding a field, 54

## F

F Command, 79  
field level help, 8  
FIND global command, 88, 89  
FIND Subcommand, 84, 88, 89  
Functions  
  LC, 85, 88  
  LF, 85, 88  
  LISTC, 85, 88  
  LISTF, 85, 88  
  MEMLIST, 82, 86, 88  
  ML, 82, 86, 88

## G

Global commands  
  COMPRESS, 88  
  FIND, 88, 89  
  GLOBAL, 88  
  MODEL, 88

## STARWARP User Guide

REPLACE, 88, 91  
SEEK, 88  
WHOHAS, 88  
GLOBAL global command, 88

### H

#### Help

command, 8  
field level, 8  
pop-up, 8  
HELP Command, 87  
Hierarchical mode, 16  
HISTORY Subcommand, 84  
HOLIDAYC, 36

### I

IF Subcommand, 89  
IMPACT, 38  
INDEX Command, 87  
internal jump, 16  
ISPF utilities, 101

### J

JCL, 78, 79  
jump, 16

### L

L Command, 80, 85  
LANGTYPE, 48  
LC Function, 85, 88  
LF Function, 85, 88  
line commands, 53  
List member specification, 90  
LIST Subcommand, 84  
LISTC Function, 85, 88  
LISTF Function, 85, 88  
LOCATE Command, 80, 85

### M

M Command, 13  
MAP LCT, 73  
Member group specification, 90  
Member specification  
\*, 90  
=, 90  
Asterisk, 90  
Combination, 90  
Equal, 90  
List, 90  
Pattern, 90  
Placeholder, 90  
Range, 90  
with %, 90  
with \*, 90  
with /, 90

with :, 90  
with ?, 90  
MEMLIST Function, 82, 86, 88  
MENU Command, 13  
Menu System, 13  
ML Function, 82, 86, 88  
MODEL Command, 82  
MODEL global command, 88  
modeling the input data set, 66  
Modes  
batch, 10, 77  
hierarchical, 16  
parallel, 16  
multiple elements, 56  
MVS utilities, 99

### O

Occurs Depending On, 58  
Option 4 WARP, 20  
OUTPUT Command, 80

### P

PACKOFF Edit Macro, 96  
PACKON Edit Macro, 96  
Parallel mode, 16  
PASTE Edit Macro, 93  
Pattern member specification, 90  
PDSEDSN Edit Macro, 96  
Placeholder member specification, 90  
primary commands, 53  
PROFMAN, 30

### R

Range member specification, 90  
REEQUAL CLIST, 95  
REPLACE global command, 88, 91  
REPLACE Subcommand, 84, 91  
Restricted commands, 11, 12  
Return code, 10

### S

saving a warp table, 61  
Scrolling  
panels, 18  
table panels, 18  
tutorial panels, 18  
scrolling the warp table, 63  
SEEK global command, 88  
SERENA, i, ii  
single subcommand mode, 10  
SORT Command, 80  
stacked ISPF session, 101  
STARBAT batch program, 68  
STARTOOL subset, 11  
Subcommands  
ATTRIB, 87

COMPARE, 83  
EDIT, 84  
EXCLUDE, 91  
FIND, 84, 88, 89  
HISTORY, 84  
IF, 89  
LIST, 84  
REPLACE, 84, 91  
VERIFY, 92

## T

TRAP Edit Macro, 97

## U

Utilities  
Batch, 99  
ISPF, 101

MVS, 99

## V

VERIFY Subcommand, 92  
VOLUME Edit Macro, 97

## W

WARP table views, 63  
WARPCOPY, 21  
WARPSAVE, 29  
WHOHAS global command, 88

## X

X Command, 80, 84, 91