Redbooks
ibm.com/redbooks

IBM z/OS V2R2: JES2, JES3, and SDSF

Keith Winnard

Jose Gilberto Biondo Jr

Jaqueline Mouro

Alvaro Salla

Ewerton Waki

z Systems

IBM

Redbooks

International Technical Support Organization

IBM z/OS V2R2: JES2, JES3, and SDSF

December 2015

SG24-8287-00

Note: Before using this information and the product it supports, read the information in “Notices” on page v.

First Edition (December 2015)

This edition applies to Version 2, Release 2, of z/OS (product number5650-ZOS).

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

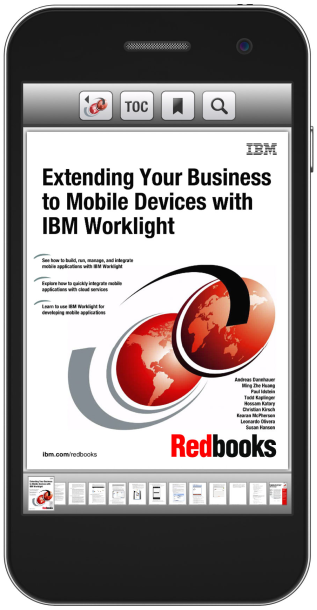
DB2®
DS8000®
GDPS®
IBM®
MVS™
Parallel Sysplex®
RACF®
Redbooks®
Redbooks (logo) ®
RMF™
z/OS®

The following terms are trademarks of other companies:

Other company, product, or service names may be trademarks or service marks of others.

Find and read thousands of IBM Redbooks publications

- ► Search, bookmark, save and organize favorites
- ► Get up-to-the-minute Redbooks news and announcements
- ► Link to the latest Redbooks blogs and videos



IBM

Extending Your Business to Mobile Devices with IBM Worklight

See how to build, run, manage, and integrate mobile applications with IBM Worklight

Explore how to quickly integrate mobile applications with cloud services

Learn to use IBM Worklight for developing mobile applications

Andreas Dannhauer
Ming Zhe Huang
Paul Idstein
Todd Kaplinger
Hossam Katory
Christian Kirsch
Kearan McPherson
Leonardo Olivera
Susan Hanson

Redbooks

ibm.com/redbooks

Get the latest version of the Redbooks Mobile App

iOS ◄ Download Now ► Android

Promote your business in an IBM Redbooks publication

Place a Sponsorship Promotion in an IBM® Redbooks® publication, featuring your business or solution with a link to your web site.

Qualified IBM Business Partners may place a full page promotion in the most popular Redbooks publications. Imagine the power of being seen by users who download millions of Redbooks publications each year!

ibm.com/Redbooks
About Redbooks → Business Partner Programs

THIS PAGE INTENTIONALLY LEFT BLANK

Preface

This IBM® Redbooks® publication helps you to become familiar with the technical changes that were introduced into the JES2 JES3, and SDSF areas with IBM z/OS® V2R2.

This book includes the following chapters:

- Chapter 1, “JES2” on page 1

  This chapter describes the JES2 updates that are related to z/OS V2R2.

- Chapter 2, “JES3” on page 29

  This chapter describes the enhancements that are provided in JES3 by z/OS V2R.

- Chapter 3, “System Display and Search Facility” on page 39

  This chapter describes the changes in SDSF in z/OS V2R2.

This book is one of a series of IBM Redbooks publications that take a modular approach to providing information about the updates that are included with z/OS V2R2. This approach has the following goals:

- Provide modular content
- Group the technical changes into a topic
- Provide a more streamlined way of finding relevant information that is based on the topic

We hope you find this approach useful and we welcome your feedback.

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Keith Winnard is the z/OS Project Leader at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and is keen to engage with customers to understand what they want from IBM Redbooks Publications. Before joining the ITSO in 2014, Keith worked for clients and Business Partners in the UK and Europe in various technical and account management roles. He is experienced with blending and integrating new technologies into the traditional landscape of mainframes.

Jose Gilberto Biondo is an IT Specialist in Integrated Technology Delivery, ServerSystems Operations/Storage Management in IBM Brazil. He has seven years of experience inz/OS, working with storage management since 2007. Jose works mainly with IBM storage products (DFSMSdfp, DFSMSdss, DFSMShsm, and DFSMSrmm), but he also works with OEM software products. Jose’s areas of expertise include installing and maintaining storage products and process automation.

Jaqueline Mouro is a Senior Systems Programmer at Banco do Brasil, a government bank in Brazil. She has 13 years of mainframe experience and holds a degree in Information Systems. Her areas of expertise include z/OS installation and maintenance, IBM Parallel Sysplex®, and security.

Alvaro Salla is a Senior IT Consultant for the ITSO. He has more than 40 years of teaching and developing educational material experience that includes the z/OS mainframe platform and consulting services focusing on performance. Alvaro has co-authored Redbooks about IBM DS8000®, DFSMS, WLM, SYSPLEX, ABCs, IBM RMF™, and disaster recovery. He started with IBM in 1969.

Ewerton Waki is an IT Analyst at Banco do Brasil, a government bank in Brazil. He has four years of experience in mainframe systems, including IBM GDPS®, DFSMS, high-end storage systems, and remote copy solutions. He holds a Bachelor in Information Systems from Universidade Estadual de São Paulo. His area of expertise includes DFSMS and storage performance for mainframe environments.

Thanks to the following people for their contributions to this project:

Bob Haimowitz (DST, Poughkeepsie Center) for setting up and maintaining the systems and providing valuable advice, guidance, and assistance throughout the creation of this IBM Redbooks publication.

Rich Conway (DST, Poughkeepsie Center) for setting up and maintaining the systems and providing valuable advice, guidance, and assistance throughout the creation of this IBM Redbooks publication.

Peter Bertolozzi (Systems Management specialist, IBM Redbooks residency support, Poughkeepsie Center) for setting up and maintaining the environments in which residents worked.

John Gierloff (Operations, Poughkeepsie Center) for residency setup and support.

Don Brennan (DST, Poughkeepsie Center) for setting up and maintaining the systems hardware that is used in the creation of this IBM Redbooks publication.

Ella Buslovich (Graphics specialist, location) for providing guidance and specialist graphics for this IBM Redbooks publication.

Ann Lund (ITSO Administration, Poughkeepsie Center) for administrative support to enable the residency.

Cheryl Gera (ITSO Administration, Poughkeepsie Center) for managing the business operations for this IBM Redbooks publication.

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

JES2

This chapter describes the JES2 updates that are related to z/OS V2R2 and includes the following topics:

- 1.1, “JES2 overview of changes” on page 2
- 1.2, “JOBGROUP” on page 2
- 1.3, “EVENTLOG” on page 16
- 1.4, “Deadline scheduling” on page 17
- 1.5, “JES3 JECL statements overview” on page 19
- 1.6, “RAS enhancements” on page 21
- 1.7, “Checkpoint improvements” on page 25
- 1.8, “EXIT updates” on page 27

1.1 JES2 overview of changes

New functions were added to JES2. These functions also are reflected in System Display and Search Facility (SDSF), which is described in Chapter 3, “System Display and Search Facility” on page 39.

The following new functions were introduced with z/OS V2R2:

- JOBGROUP
- EVENTLOG (JOB STEP COMPLETION CODE)
- DEADLINE SCHEDULING
- JECL
- INCREASED NUMBER OF RUNNING JOBS
- GROWTH CHECKPOINT WITHOUT COLDSTART
- DYNAMIC CHECKPOINT TUNING

These functions are described next.

1.2 JOBGROUP

The resource usage that is required to process increasing workloads and meet business service level agreements (SLAs) demands efficiency in all areas. Increasing CPU resources might not always be the answer. One area to examine is the batch workload. A shorter batch window can increase the resources that are available to the online workload after the critical batch work is complete.

The z/OS Workload Manager (WLM) plays a significant role in apportioning resources in line with performance policies to optimize and prioritize workloads. However, other options and techniques also are useful, such as batch parallelism and conditional job execution.

Batch parallelism might involve breaking long jobs into smaller jobs and allowing these jobs to run concurrently or conditionally, if the correct dependencies are in place to maintain data integrity.

A typical business has a batch workload and the work streams are managed by the operational schedules. Consider these other areas of batch work that are used across IT and users for the following tasks:

- Application development
- Application development test cycles
- Environment set-up for projects
- Operational ad hoc jobs
- User submitted batch work
- Systems software updates and roll outs
- Testing corrective maintenance to an application
- Jobs that are triggered by automation
- On-demand housekeeping that is driven by events
- Information gathering for problem determination
- Unit tests that are submitted by IT staff

In z/OS V2R2, JOBGROUP new functions enable users and programmers to improve batch parallelism and conditional job processing by defining a basic execution control sequence.

JOBGROUP provides a simple, flexible way to control job execution through JES. It uses syntax that is similar to JCL to define jobs to the group, including conditional job processing.

Although it provides an extra layer for controlling job execution, it is *not* intended to replace or act as batch job scheduler.

Assume that you have a single job with five steps with step names Step1 - Step 5. The first step creates an output data set that is used by Step 2. Step 3 creates output data for Step 4, but is not directly dependent on data from steps 1 or 2, and Step 5 uses the input from the four previous steps.

By using JOBGROUP, you can split the steps into five different jobs and make non-dependent jobs to run concurrently, which reduces overall elapsed time. Figure 1-1 shows this scenario and its advantages.

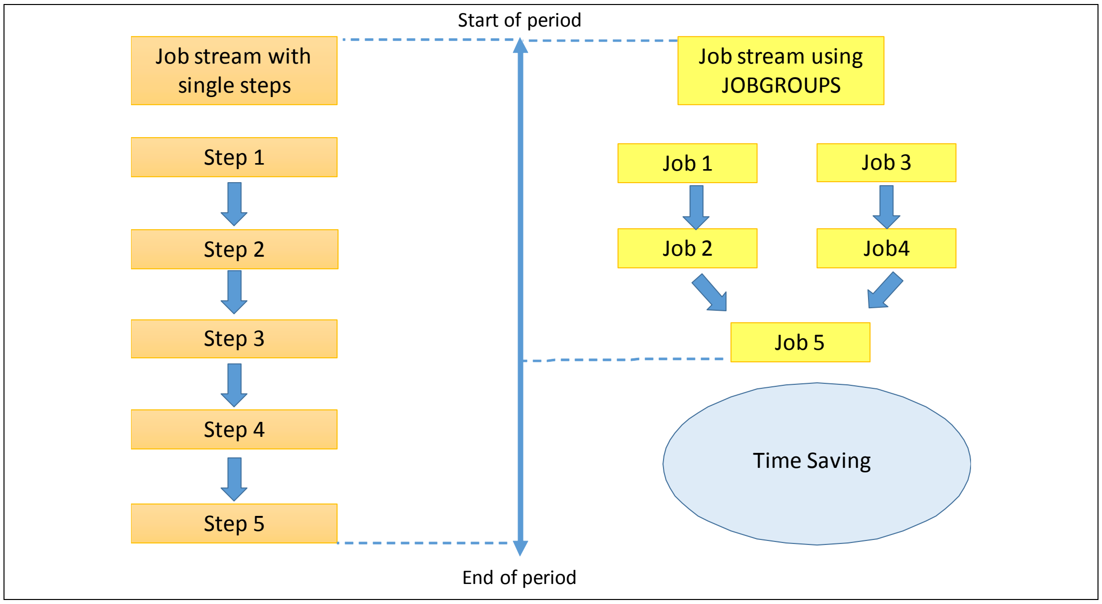
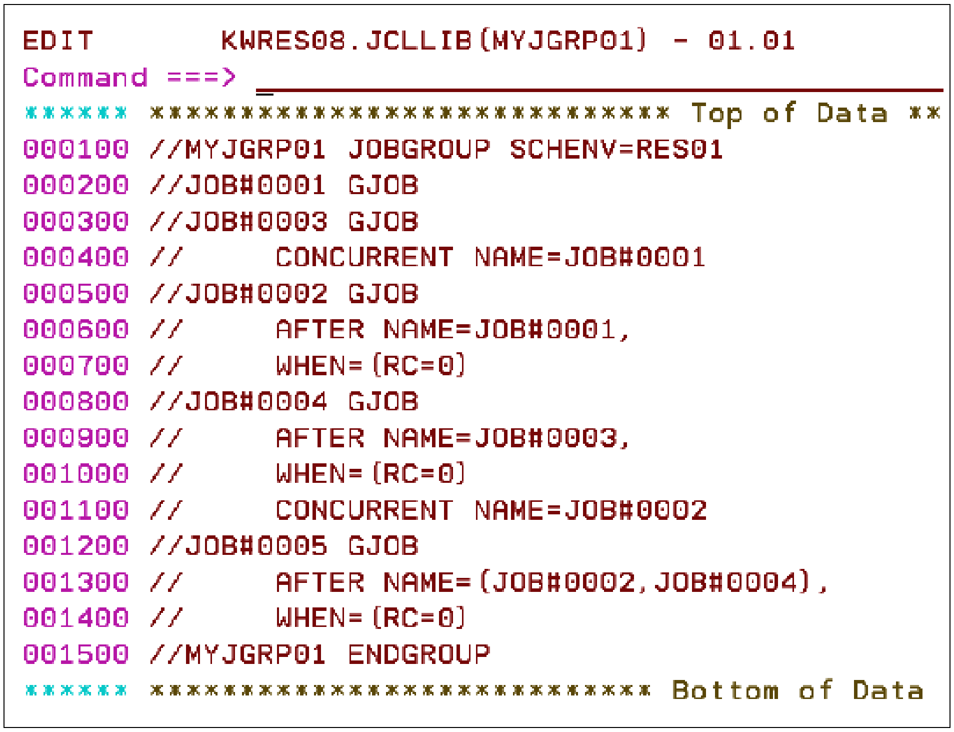


Figure 1-1   Sample JOBGROUP scenario

The JOBGROUPs that are shown in Figure 1-1 show how the elapsed time can be reduced by using batch parallel processing.

Figure 1-2 shows the statements that are used to achieve this scenario.



```
EDIT       KWRES08.JCLLIB(MYJGRP01) - 01.01
Command ===> _________________________________________________
****** ***************************** Top of Data **
000100 //MYJGRP01 JOBGROUP SCHENV=RES01
000200 //JOB#0001 GJOB
000300 //JOB#0003 GJOB
000400 //      CONCURRENT NAME=JOB#0001
000500 //JOB#0002 GJOB
000600 //      AFTER NAME=JOB#0001,
000700 //      WHEN=(RC=0)
000800 //JOB#0004 GJOB
000900 //      AFTER NAME=JOB#0003,
001000 //      WHEN=(RC=0)
001100 //      CONCURRENT NAME=JOB#0002
001200 //JOB#0005 GJOB
001300 //      AFTER NAME=(JOB#0002,JOB#0004),
001400 //      WHEN=(RC=0)
001500 //MYJGRP01 ENDGROUP
****** **************************** Bottom of Data
```

Figure 1-2   Statements to define the JOBGROUPs scenario

The JES checkpoint data contains new areas that are dedicated to JOBGROUP information. A new data area that is called Zone Job Container (ZJC), is used to store JOBGROUP information. The following ZJCs are needed to represent the group:

- One per group (ZOD)
- One per job in the group (ZJI)
- One per dependency (ZDB)

These data areas are allocated when a JOBGROUP is submitted and freed when the JOBGROUP logging job is purged after all jobs from the group are completed and purged.

Attention: Because the use of JOBGROUPs requires new section and data area within checkpoint data sets, it is possible to use this function only when running on new checkpoint level z22. For more information about the new checkpoint level, see 1.6.1, “New $ACTIVATE level” on page 21.

You can use the **`$T GRPDEF`** command to manage the JOBGROUP definition parameters. This command can be used to define values to the following attributes:

- ZJCNUM

  This attribute defines the number of ZJC objects that are allocated in checkpoint for defining parts of a job group. One ZJC is used for each Job group information, dependent job information, and information about dependencies between jobs. The range is 1 - 500000**;** the default is 1000.

- CONCURRENT_MAX

  This attribute defines the maximum number of dependent jobs that can be defined in a single concurrent job set. The range is 0 - 200; default is 0.

- ZJCWARN

  This attribute defines the percentage of ZJC objects that are used before a warning message is issued. The range is 1 - 99; default is 80.

- JOBGROUP_JOB_MAX

  This attribute defines the maximum number of JOBs that can be defined within a JOBGROUP. The range is 10 - 2000.

Example 1-1 shows a sample **$T GRPDEF** command to modify the ZJC number and warning percentage.

Example 1-1 Update number of ZJC objects and warnings

```
$T GRPDEF,ZJCNUM=5000,ZJCWARN=90
```

You can also use the **$D GRPDEF** and **$T GRPDEF** commands to validate the values that are assigned to each attribute. The **$D GRPDEF,ZJCUSE** command also provides more information about ZJC usage, as shown in Example 1-2.

Example 1-2 Output from $D GRPDEF,ZJCUSE

```
$HASP732 GRPDEF 799
$HASP732 GRPDEF  CURRENT ZJC UTILIZATION
$HASP732         TYPE         COUNT
$HASP732         -------- ---------
$HASP732         FREE          988
$HASP732         JOBGROUP        2
$HASP732         DEP JOB         6
$HASP732         DEPENDNT        4
```

To assist identifying, analyzing, and monitoring JOBGROUPs, a Job Group panel was added to SDSF. The SDSF changes are described in Chapter 3, “System Display and Search Facility” on page 39.

JOBGROUP definition

The JOBGROUP has a structure similar to the structure in JCL. There is a main card to define the JOBGROUP name and control parameters, and one or more GJOBs to specify the jobs that are part of the group. You can also define JOBSETs that are a group of jobs inside the main JOBGROUP. This configuration can be compared to PROCs in JCL statements.

Example 1-3 shows a basic JOBGROUP definition with all of these characteristics. JOBGROUP and JOBSET cards have an ENDing card to identify the end of the JOBSET or JOBGROUP.

Example 1-3 Basic structure of a JOBGROUP

```
//JOBGRP01 JOBGROUP
//GRPJOB1  GJOB
//JOBSET1  JOBSET
//SETJOB1  SJOB
//SETJOB2  SJOB
//SETJOB3  SJOB
//JOBSET1  ENDSET
//GRPJOB2  GJOB
//GRPJOB3  GJOB
//JOBGRP01 ENDGROUP
```

Note: Example 1-3 on page 5 shows only the basic JOBGROUP processing structure and syntax. It does not provide any conditional tests or dependencies. These topics are covered next.

In Example 1-3 on page 5, the first line creates a JOBGROUP that is named JOBGRP01 and identifies the line as being a JOBGROUP card. Lines 2, 8, and 9 identify the jobs that belong to the JOBGRP01 and the names next to // are the job names in the environment. Lines 3 and 7 identify the beginning and end of a JOBSET. Lines 4 - 6 identify the jobs that are assigned to JOBSET1. The last line ends the JOBGROUP definition.

JOBGROUP keywords

Several keywords can be used when JOBGROUP is defined to control job execution. The keywords define how JES handles all the jobs that are part of the group. If any jobs within the group require a different configuration, it must be specified on job card.

Example 1-4 provide the sample syntax for JOBGROUP keywords, which is described next. Default values are underscored in this example.

Example 1-4 JOBGROUP parameters

```
//grpname JOBGROUP accounting information
// programmer name,
// OWNER=userid,
// GROUP=racf group id,
// PASSWORD=password,
// SECLABEL=seclabel,
// TYPE=SCAN,
// HOLD=NO|YES,
// ERROR=(condition),
// ONERROR=(STOP|SUSPEND|FLUSH),
// SYSAFF=(affinity_list),
// SYSTEM=(system_list),
// SCHENV=scheduling_environment
```

The JOBGROUP includes the following parameters:

- `grpname`

  This parameter is the name that is associated with the job group. It is the job name of the logging job that is associated with the group. This field is required.

- `accounting information`

  Use the accounting information parameter to enter an account number or other accounting information, as with use as JOB statement. This field is optional.

- `programmer name`

  Use the programmer's name parameter to identify the person or group who is responsible for a job group as with use as JOB statement. This field is optional.

- `OWNER`

  This parameter is the user ID that is to be associated with the job group. This information propagates by using the same rules as a batch job. This field is optional.

- `GROUP`

  This parameter is the IBM RACF® group that is to be associated with job group. This information propagates by using the same rules as a batch job. This field is optional.

- `PASSWORD`

  This parameter is the password (if required) for the user ID that is associated with the job group. This information propagates by using the same rules as a batch job. This field is optional.

- `SECLABEL`

  This parameter is the security label that is to be associated with the job group. This information propagates by using the same rules as a batch job. This field is optional.

- `TYPE=SCAN`

  The JOBGROUP is checked for validity, but not processed. Any error is recorded in the logging job. Because the internal structures for this job group are never created, any jobs that are submitted for this job group fail. This field is optional.

- `HOLD=NO|YES`

  The JOBGROUP can be submitted in a held or non-held state. If the job is submitted in the held state, none of the jobs that are associated with this job group run until the job group is released. If you do not code the HOLD parameter on your JOBGROUP, the default is used.

- `ERROR`

  This parameter defines conditions that cause the group to be placed in an error state if encountered by any job in the job group. The syntax is the same as the WHEN= keyword. The effect of placing a job in an error state depends on the setting of the ONERROR= keyword. This field is optional.

- `ONERROR=STOP|SUSPEND|FLUSH`

  This action is the action to take when a JOBGROUP is determined to be in error. This rule applies when the condition that is defined on the ERROR= keyword is encountered or when a dependency is considered to fail. If you do not ONERROR on your JOBGROUP, the default is used. The following ONERROR actions are possible:

  - STOP

    No new jobs in the JOBGROUP are run. Running jobs can complete. Jobs that are determined to be in error (based on the JOBGROUP ERROR= keyword or the condition in a dependency) can be resubmitted and the error state cleared if they run successfully. This action is the default.

  - SUSPEND

    New jobs that have their dependencies satisfied can run. Jobs that are determined to be in error (based on the JOBGROUP ERROR= keyword or the condition in a dependency) are considered to not run. These jobs can be resubmitted and the error state cleared if the jobs run successfully.

  - FLUSH

    All jobs that are not yet run are canceled (flushed). No new jobs are started. After there are no longer any jobs running, the job group is marked completed.

- `SYSAFF`

  Base system affinity for all jobs that are associated with this JOBGROUP. Syntax is the same as SYSAFF= on the JOB card. This affinity is ANDed with any affinity specification for each job in the group. This field is optional.

- `SYSTEM`

  Base list of systems where jobs that are associated with this job group can run. Syntax is the same as SYSTEM= on the job card. This list is ANDed with any affinity specification for each job in the group. This field is optional.

- `SCHENV`

  Default scheduling environment for all jobs that are associated with this job group. Syntax is the same as SCHENV= on the job card. The systems where this scheduling environment is available is ANDed with the other affinity specifications for the job. The ANDed process implies that a job in a job group can have two scheduling environments: one for the job group and one for the job in the group, which are ANDed together. This field is optional.

JOBSET keyword

In addition to the JOBGROUP parameters, you can set parameters for JOBSET, GJOB, and SJOB. Example 1-5 shows the structure for JOBSET parameters.

Example 1-5 JOBSET parameter

```
//setname JOBSET FLUSHTYP=ALLFLUSH|ANYFLUSH
```

The JOBSET includes the following parameters:

- `setname`

  This parameter is the name that is associated with the job set. This name is the name that is used when referencing the job set. This field is required.

- `FLUSHTYP=ALLFLUSH|ANYFLUSH`

  For each job in the set, the job is flushed if ALL parent jobs are flushed (ALLFLUSH) or if ANY parent jobs are flushed (ANYFLUSH). This field is optional; if it is not coded, the default ALLFLUSH is used.

When JOBSET is used within a JBGROUP, only the JOBSET should be used for BEFORE, AFTER, CONCURRENT, and conditional testings. However, the SJOBs within the JOBSET can reference each other.

GJOB and SJOB keywords

In addition to JOBGROUP and JOBSET, you can use parameters to control single job execution. Throughout these parameters, you can decide the order that the jobs run and error handling. The same parameters apply to GJOB and SJOB, as shown in Example 1-6.

Example 1-6 GJOB and SJOB control parameters

```
//gjobname GJOB|SJOB FLUSHTYP=ALLFLUSH|ANYFLUSH
// CONCURRENT NAME=name|(name,name,...),
// AFTER NAME=name|(name,name,...),
// BEFORE NAME=name|(name,name,...),
// WHEN=(condition),
// ACTION=SATISFY|FLUSH|FAIL,
// OTHERWISE=FLUSH|FAIL|SATISFY
```

The following parameters are used:

- `gjobname`

  The name of a job that is included in the JOBGROUP or JOBSET. This name must match the name of a job that is submitted (via SCHEDULE) after the JOBGROUP is defined. Each job name must be unique within the scope of the JOBGROUP. This field is required.

- `FLUSHTYP=ALLFLUSH|ANYFLUSH`

  This job is flushed if ALL parent jobs are flushed (ALLFLUSH) or if ANY parent jobs are flushed (ANYFLUSH). If you do not code the FLUSHTYP parameter on your JOBGROUP, the default is used.

- `CONCURRENT NAME=`

  This parameter states that the specified job on the GJOB or SJOB statement and the jobs or job sets on the CONCURRENT NAME= must run simultaneously on the same JES2 MAS member. Keywords WHEN, ACTION, and OTHERWISE are not valid with this parameter, but can be used with AFTER and BEFORE statements. All jobs in a concurrent dependency must use the same WLM service class.

- `AFTER NAME=`

  This parameter defines jobs or job sets that the current job must run after the jobs that are specified on NAME. Up to 10 names can be specified. Each name must be unique among all the name values for the job that contains the AFTER statement.

- `BEFORE NAME=`

  This parameter defines jobs or job sets that the current job must run before the jobs that are specified on NAME. Up to 10 names can be specified. Each name must be unique among all the name values for the job that contains the BEFORE statement.

- `WHEN=`

  This parameter is an optional condition that refers to the ending status of the jobs that are specified on BEFORE|AFTER NAME=. This parameter is similar to IF processing on JCL. The following keywords are supported:

  – RC: Indicates a job's return code
  – ABEND: Indicates that an ABEND condition occurred
  – ¬ABEND: Indicates that no ABEND condition occurred
  – ABENDCC: Indicates a specific system or user ABEND code
  – RUN: Indicates that the job was run
  – ¬RUN: Indicates that the job was flushed from the job group

- `ACTION=SATISFY|FLUSH|FAIL`

  This parameter is one of the following actions to take if the WHEN condition is true:

  – SATISFY

    The condition is to be considered satisfied. This setting is the default.

  – FLUSH

    The dependency is to be considered flushed. The dependent job might be flushed, depending on the dependent job's FLUSHTYP=ALLFLUSH/ANYFLUSH value.

  – FAIL

    The failure of this dependency marks the job group in error. The ONERROR= action from the JOBGROUP statement is taken as a result of the failure.

- `OTHERWISE=FLUSH|FAIL|SATISFY`

  This parameter is one of the following actions to take if the WHEN condition is false:

  - FLUSH

    The dependency is to be considered flushed. The dependent job might be flushed, depending on the dependent job's FLUSHTYP=ALLFLUSH/ANYFLUSH value. This setting is the default.

  - FAIL

    The failure of this dependency marks the job group in error. The ONERROR action from the JOBGROUP statement is taken as a result of this failure.

  - SATISFY

    The condition is to be considered satisfied.

Setting up and running jobs under JOBGROUP

After creating the required JOBGROUPs, you must update your current JCL to assign the jobs to a specific JOBGROUP. This update can be done by inserting a new SCHEDULE statement to your job. This statement indicates in what JOBGROUP the job is participating. Example 1-7 shows sample job with SCHEDULE statement.

Example 1-7 Sample JCL that uses SCHEDULE statement

```
//JOBA JOB (XXX),'JOBGRP1',MSGCLASS=A,CLASS=A,NOTIFY=KWRES08
// SCHEDULE JOBGROUP=RUN00001
```

Before submitting your jobs to run under JOBGROUP control, you must submit the JOBGROUP to internal reader for processing. This submission can be done by using an IEBGENER to place the code into internal reader or by submitting it from ISPF panels.

If you do not have your JOBGROUP in place by the time you submit your jobs with SCHEDULE statement, they fail conversion and a corresponding message is displayed. There are *no* messages on the job log that are related to the reason of the failure. The message that is displayed in Example 1-8 was captured from SYSLOG instead.

Example 1-8 `$HASP305` message for failing JOBA

```
$HASP305 FAILURE DURING CONVERSION FOR JOB JOBA - Following job group
         not valid - RUN00001
```

A sample JCL to place JOBGROUP into internal reader is shown in Example 1-9.

Example 1-9 IEBGENER to send JOBGROUP to internal reader

```
//JOBA JOB (XXX),'JOBGRP1',MSGCLASS=A,CLASS=A,NOTIFY=KWRES08
//TEST EXEC PGM=IEBGENER
//SYSUT1 DD DATA,DLM='@@'
//RUN00001 JOBGROUP OWNER=KWRES08
//JOBA GJOB
// AFTER NAME=JOBB
//JOBD GJOB
// AFTER NAME=JOBB
//JOBB GJOB
//RUN00001 ENDGROUP
@@
//SYSUT2 DD SYSOUT=(,INTRDR)
```

```
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
```

The JOBGROUPs do not have the same JOBID format; instead, they use “G” in the first position of JOBID field. Also, the JOBGROUPs are not listed in the ST and DA panels of SDSF, but are included in the new panel JG (for Job Group).

SETUP queue

The SETUP queue is used for logging jobs for a JOBGROUP, jobs within a job group that did not have their dependencies met, and concurrent running jobs that are waiting to be moved onto the running queue.

When jobs are set up to run concurrently, the following process is used to run the jobs:

1. One job in the set is the “master” job (it cannot be predicted which job receives this designation).
2. The master job is placed in the run queue when *all* jobs meet their dependencies.
3. When the master job is at the head of the queue, WLM is called to see where the set must run.
4. The master job returns to the SETUP queue.
5. WLM selects jobs by number (similar to $SJ).
6. Jobs move to the run queue when selected.

Managing JOBGROUPs

The following basic methods can be used for viewing and managing JOBGROUPs:

- JES2 commands
- SDSF panels

In this section, we focus on JES commands. For more information about how to manage JOBGROUPs by SDSF commands, see Chapter 3, “System Display and Search Facility” on page 39.

Displaying JOBGROUP information

When you are working with JOBGROUPs, you must find all of the information about your groups, including running status, delays, and errors. JES2 was enhanced with several JOBGROUP commands that are described in this section.

Although JOBGROUPs accept commands, such as Purge, Hold, and Release, you cannot perform these actions by using the JOB commands, such as $DJ and $AJ. Instead, you must use the correct groups commands.

The following JOBGROUP commands provide full control over your JOBGROUPs:

- $AG

  Releases the JOBGROUP. Jobs that are defined to the job group can be run.

- $CG

  Cancels all the jobs in a job group and the JOBGROUP.

- $DG

  Displays JOBGROUP. Various views are provided.

- $HG

  Holds the JOBGROUPs. Prevents any jobs that are defined to the job group from starting to run.

- $PG

  Purge all of the jobs in the job group. When the last job is purged, the JOBGROUP is scheduled to be purged. Unlike jobs, you cannot purge an active or pending JOBGROUP. You must cancel it by using the **`$CG`** command first.

$DG command

From the display group command, you can use keywords to retrieve information regarding the JOBGROUP, jobs that are associated with this group, and job dependency. This information can be displayed in a short or long form for further analysis.

The JOBGROUP display brings information about the group’s status, the action to take on error processing, error definition, system affinity, hold status, and owner.

Example 1-10 shows the **`$DG`** command and output.

Example 1-10 $DG command output

```
$DG(9618)
$HASP890 JOB(RUN00001) 445
$HASP890 JOB(RUN00001)  JOB_GROUP_STATUS=COMPLETE,ONERROR=STOP,
$HASP890                SYSAFF=(ANY),HOLD=(NO),OWNER=KWRES08
```

JOB_GROUP_STATUS includes the following possible parameters:

- PENDING

  No jobs were registered with the job group.

- ACTIVE,INIT

  There is at least one registered job, not all jobs are registered, and some jobs might be active.

- ACTIVE

  All jobs are registered; jobs might be active.

- FLUSHING

  Unrecoverable error causes job group termination. All jobs that are not run are canceled (flushed), and the job group status moves to COMPLETE when the last running job completes.

- SUSPENDING

  At least one job INERROR. A subset of jobs cannot run, and jobs that can run do run.

- SUSPENDED

  At least one job INERROR. A subset of jobs cannot run and jobs that can run did run.

For more information, see “JOBGROUP keywords” on page 6.

To display the jobs that are associated to a JOBGROUP, you can use the **`$DG,JOBS`** or **`$DG,JOBF`** commands. The first command tables with job name, job ID, and status, the second command provides more information about JOBSET name and FLUSH_ACTION.

Example 1-11 shows the output of the **$DG(xxxx),JOBF** command to display all jobs status.

Example 1-11 Output from $DG(xxxx),JOBF command

```
$DG(9698),JOBF
$HASP890 JOB(RUN00001) 705
$HASP890 JOB(RUN00001)  JOB_NAME=JOBB,JOBID=NONE,
$HASP890                JOB_STAT=NOT_REG,COMP_STAT=PENDING,
$HASP890                JOBSET_NAME=,FLUSH_ACTION=ALLFLUSH,
$HASP890                JOB_NAME=JOBD,JOBID=NONE,
$HASP890                JOB_STAT=NOT_REG,COMP_STAT=PENDING,
$HASP890                JOBSET_NAME=,FLUSH_ACTION=ALLFLUSH,
$HASP890                JOB_NAME=JOBA,JOBID=JOB09699,
$HASP890                JOB_STAT=PEND_DEP,COMP_STAT=PENDING,
$HASP890                JOBSET_NAME=,FLUSH_ACTION=ALLFLUSH
```

JOB_STAT includes the following possible parameters:

- NOT_REG

  The referred job was not yet submitted for processing; therefore, the job is not registered to the JOBGROUP.

- PEND DEP

  Pending dependencies keep this job from running.

- HELD

  The job is held (HOLD=(JOB)).

- ACTIVE

  The job is active in running.

- DELAYED

  The job is eligible for running but features a resource delay. For more information about the delay that is affecting the job, use the **$DJ,DELAY** command.

- NOT ELIG

  Deemed ineligible to run by job group processing.

In addition to job status commands, job dependencies can be retrieved to identify the cause of possible delays on a JOBGROUP. There are commands available to display job dependency definitions and jobs running before, after, and concurrently with any specific job within a JOBGROUP.

Use the **$DG(xxxx),DEP** or **$DG(xxxx),DEPF** commands to display all dependencies in a JOBGROUP. Also, use **$DJ(xxxx),BEFORE** , **$DJ(xxxx),CON** and **$DJ(xxxx),AFTER** commands to display jobs that must run before, concurrently, or after the specified job.

The output of the **$DG(xxxx),DEPF** command is shown in Example 1-12. The DEPF keyword was used because it provides a more detailed output. The **DEP** command displays the job dependency data in a table format.

Example 1-12 $DG(xxxx),DEPF command output

```
$DG(9709),DEPF
$HASP890 JOB(RUN00001)
$HASP890 JOB(RUN00001)  PARENT_JOB=JOBB,DEP_JOB=JOBA,
$HASP890                DEP_STAT=PENDING ,COMP_STAT=SATISFY,
$HASP890                WHEN=RC=0000,ENDACTION=SATISFY,
$HASP890                  OTHERWISE=FLUSH,
$HASP890                  PARENT_JOB=JOBB,DEP_JOB=JOBD,
$HASP890                  DEP_STAT=PENDING ,COMP_STAT=SATISFY,
$HASP890                  ENDACTION=SATISFY,OTHERWISE=FLUSH
```

Only jobs JOBA and JOBD are displayed as DEP_JOB. JOBB is not displayed because it has no dependencies. The following output fields are available:

- PARENT_JOB

  Column indicates the parent job that must complete before the dependency of the job in the DEP JOB column can be evaluated.

- DEP_JOB

  The dependent job whose evaluation depends on the completion of the parent job.

- DEP_STAT

  Status of the dependency between the parent and dependent job. It includes the following possible values:

  - PENDING

    The evaluation of the dependency is pending. The parent job did not run and it was not flushed.

  - COMPLETE

    The evaluation of the dependency is complete. The parent job ran or was flushed and the completion action of the dependency was assigned.

- COMP ACT

  Completion action that is assigned to the dependency. It is used to determine how to process the dependent job when all its dependencies are complete and all dependency completion actions can be evaluated. Includes the following possible values:

  - SATISFY

    This dependency was satisfied by the parent job run. If this completion action alone was evaluated, the dependent job is eligible for running.

  - FLUSH

    Parent job results indicate that the dependent job must be canceled or flushed and not run.

  - FAIL

    Parent job results cause the dependency to be marked as failed and the dependent job does not run. The entire job group is marked in error and the ONERROR setting of the job group determines the processing.

The WHEN field shows the logical condition that is evaluated before the job run. If the logical test returns TRUE (which means the parent job completed with the specified return code), the JOBGROUP takes the action that is specified on ENDACTION. If the condition is not met, it uses OTHERWISE instead.

The **$DG(xxxx),INERROR** command can also be used to display information about jobs that are in error within the JOBGROUP. The **$DG(xxxx),SUMMARY** command provides a summary of all the information regarding job dependency, run, and errors.

Controlling JOBGROUPs

You can manage JOBGROUPS after they are submitted. The following commands are available to manage JOBGROUP attributes:

- Hold
- Release
- Cancel
- Purge
- Change

There are situations in which it is necessary to hold or release JOBGROUPs to prevent jobs from running in an environment because of a scheduled change, outage, upgrades, or to allow critical jobs to process during peak periods.

A JOBGROUP can be held by running a **`$HG(xxxx)`** command. This command prevents any jobs from running, but allows the running jobs to complete. In the same way, run the **`$AG(xxxx)`** command to release the JOBGROUP.

Holding or releasing the JOBGROUP does not change specific job attributes. Therefore, a held job is not released by running a **`$AG(xxxx)`** command, even if this job is part of the JOBGROUP.

Example 1-13 includes the command and output of a hold JOBGROUP. The output includes the changed JOB_GROUP_STATUS=HELD and other group information.

Example 1-13 $HG(xxxx) output

```
$HG(9713)
$HASP890 JOB(RUN00001) 927
$HASP890 JOB(RUN00001)  JOB_GROUP_STATUS=HELD,ONERROR=STOP,
$HASP890                SYSAFF=(ANY),HOLD=(YES),OWNER=KWRES08
```

Under certain circumstances, you might need to cancel a job within the JOBGROUP or the group. There are some special considerations that you must take when a job is canceled from a JOBGROUP.

Canceling a job part of the JOBGROUP causes this job to be considered IN ERROR status. It also causes the group to be in SUSPENDING, SUSPENDED, or COMPLETE state, based on the description that is provided in “$DG command” on page 12.

You must place any necessary changes and resubmit the canceled job to continue JOBGROUP processing.

Your JOBGROUP logging job is not purged from your system until all JOBGROUP jobs are purged. After all jobs are purged from the spool, the logging job also is purged. If you purge a logging JOBGROUP, all the jobs also are purged.

It is possible to purge only COMPLETE JOBGROUPs. If you have a SUSPENDED, ACTIVE, or PENDING JOBGROUP, first issue a cancel command to complete the group and then, run the purge command.

Note: Purging any job in an active concurrent set causes all jobs in the concurrent set to be canceled.

The SCHENV and SYSAFF attributes can also be modified after the JOBGROUP is defined. This ability allows you to redirect your jobs to other environments if a system becomes unavailable for processing.

Run the **$TG(xxxx)** command to update SCHENV and SYSAFF attributes. Example 1-14 shows a **$TG** command to update SCHENV attribute, along with its output.

Example 1-14 Change JOBGROUP SCHENV attribute

```
$TG(9752),SCHENV=RES01
$HASP890 JOB(RUN00001) 321
$HASP890 JOB(RUN00001)  JOB_GROUP_STATUS=PENDING,ONERROR=STOP,
$HASP890                SYSAFF=(ANY),HOLD=(NO),CMDAUTH=(LOCAL)
$HASP890                SECLABEL=,OWNER=KWRES08,
$HASP890                SPOOL=(VOLUMES=(BH6SP1),TGS=1,
$HASP890                PERCENT=0.0370),CARDS=6,REBUILD=NO,
$HASP890                SCHENV=RES01(SET BY OPERATOR),
$HASP890                SCHENV_AFF=(SC76),CC=(),DELAY=(),
$HASP890                CRTIME=(2015.173,15:47:38),
$HASP890                JOBGROUP=RUN00001
```

SSI updates

There are updates on exits 2 and 52 to handle new JOBGROUP information and other enhancements. For more information about SSI updates, see 1.8, “EXIT updates” on page 27.

1.3 EVENTLOG

In the past, the job completion return code was used to determine whether a job failed and if any other actions must be taken to correct an error. A single job can have many steps and be complex; therefore, it might be necessary to take actions that are based on single-step outcome.

With the enhancements in z/OS V2R2, JES2 logs data about each step in a new data set that is called EVENTLOG. This log provides customers with the granular step information that they need to fully evaluate the running of their jobs.

1.3.1 Implementation and usage

The new JES2 spool data set EVENTLOG is allocated automatically for batch jobs, started tasks, and TSO users. It contains machine-readable records that are non-printable, non-spinnable, and are viewed via SPOOL data set browse.

The EVENTLOG data set contains data about three subtypes of SMF type 30 records. It includes Job start or start of other work unit (subtype 1), step total (subtype 4), and Job termination or termination of other work unit (subtype 5).

There also is a new JOBDEF attribute to control if SMF data is written to EVENTLOG called SUP_EVENTLOG_SMF. You can use the **$T JOBDEF,SUP_EVENTLOG_SMF** command to suppress (YES) or write (NO) SMF records to EVENTLOG. The default value NO means that the SMF records are *not* suppressed. Example 1-15 on page 17 shows the JOBDEF command to process SMF data.

Example 1-15 JOBDEF command to process SMF data

```
$T JOBDEF,SUP_EVENTLOG_SMF=NO
```

SMF Type 30 subtype 4 records cause a STEPDATA record to be written to EVENTLOG. STEPDATA and RESTART records always are written, even if SUP_EVENTLOG_SMF=YES, SMF is not active, or SMF Type 30 records are suppressed. Suppressing SMF EVENTLOG records might affect the amount of information that is provided by job display panel on SDSF. For more information about job display panel, see 3.5.1, “Job step panel” on page 49.

You can access EVENTLOG by using SPOOL data set browse. You can read all EVENTLOG records by using the fully qualified data set name `userid.jobname.jobID.D0000008.EVENTLOG`, or logical data set name `userid.jobname.jobID.EVENTLOG`.

The following logical data set names can be used to read specific record types:

- `userid.jobname.jobID.EVENTLOG.STEPDATA`

  Reads STEPDATA records only.
- `userid.jobname.jobID.EVENTLOG.SMF`

  Reads SMF records only.
- `userid.jobname.jobID.EVENTLOG.SMFSTEP`

  Reads SMF Type 30 subtype 4 records only.
- `userid.jobname.jobID.EVENTLOG.RESTART`

  Reads RESTART records only.

EVENTLOG data set is eligible for transmitting and receiving via NJE, if both JES nodes include the new feature bit NCCINOS in the NCCIFEAT bytes set on. This bit indicates that this NJE node supports transmitting and receiving non-printable SYSOUT data sets. If any of the transmitting or receiving hosts do not have this option on, the EVENTLOG is lost.

You can also view step completion information via SDSF panels. For more information about how to use SDSF panels to display step completion data, see 3.5, “Job displays” on page 48.

1.4 Deadline scheduling

A common requirement of job processing is that some jobs must start at a specific time to perform critical tasks, such as housekeeping and system maintenance. To assist users and programmers with basic scheduling controls in JES2, the date and time a job must run now can be set.

1.4.1 Implementation and usage

Starting with z/OS V2R2, there are new keywords available for use with SCHEDULE JCL statement. These new keywords assist the user to keep a job in a held state until a specified time, a wanted time for a job to start, or that a job should run on the same system where a reference job is running.

The new HOLDUNTL and STARTBY keywords have a similar syntax and can be used with your JCL to define the time that a job is to be released from hold status and by what time the job must run. If both keywords are used, STARTBY *must* be a later time than HOLDUNTL, and both must use compatible date and time formats.

The following syntax can be used for HOLDUNTL and STARTBY. If you use both keywords in your JCL, both must use the same syntax:

- HOLDUNTL='+hh:mm'

  A delta time from when the job entered the system. This time is not subject to time offset changes.

- HOLDUNTL=('hh:mm',mm/dd/yyyy) or HOLDUNTL=('hh:mm',yyyy/ddd)

  A specific time in the future when job should be released (the date is optional). This time is a local system time and is subject to time offset changes.

HOLDUNTL

This parameter is used to specify until what date and time a job must be on hold. After the specified date and time, the job is released from hold status. If the JOB JCL code includes TYPRUN=HOLD, the job is still released. If you use a past time with no date, the next day is assumed to be the date. If you use a past date and time, the job is never held.

You can use HOLDUNTL keyword with JOBGROUP to increase the flexibility of your batch schedule. Example 1-16 shows a simple JCL with HOLDUNTL keyword to prevent the job from processing until 06/06/2016 at 15:40.

Example 1-16 JCL with HOLDUNTL keyword

```
//JOBB JOB TIME=NOLIMIT,REGION=0K,MSGCLASS=A,CLASS=A,NOTIFY=KWRES08
// SCHEDULE HOLDUNTL=('15:40',06/06/2016)
//STEP1 EXEC PGM=IEFBR14
//DD1 DD DSN=KWRES08.JGROUP1,DISP=(NEW,CATLG),SPACE=(TRK,(1,1)),
//   RECFM=FB,LRECL=80,BLKSIZE=0
```

You can use the **$DJ(xxxx),HOLDUNTL** command to display HOLDUNTL information regarding a specific job. HOLDUNTL information is *not* displayed by using a display delay command or SDSF panels. You can identify a job that is held by using the HOLDUNTL command by performing the command **$DJ(xxxx)**, which provides job information and includes the HOLDUNTL keyword in the HOLD attribute.

STARTBY

The STARTBY keyword can be used to specify a date and time when the job must start running. This specification does *not* mean that the job always starts at the specified time. Instead, JES2 gradually moves a job to the top of the run queue to give the job a better chance to be selected for running.

Note: Unlike HOLDUNTL, STARTBY does *not* place the job on hold or release it. If the job was put on hold, it must be released for STARTBY to work.

When a job is coded with STARTBY, the job is run if all considerations (initiators, system affinity, and so on) are satisfied. Therefore, if a job is submitted by 16:40 with a STARTBY='18:00', and all resources are available, the job runs immediately, and it does not wait until 18:00 to run. To achieve this status, use STARTBY with HOLDUNTL.

If the job is not selected for processing by the time that is specified on STARTBY, JES2 uses the value that is set on PROMO_RATE on JOBCLASS definition to move the job up on the run queue in 1 minute cycles. The default value for PROMO_RATE is 0, meaning that no job promotion is performed.

Because STARTBY can change job priority on the run queue, it is mutually exclusive with JOBGROUP keyword.

Use the **$JOBCLASS(x),PROMO_RATE=x** command to change the default values for PROMO_RATE. Also, update JES2 parmlib with PROMO_RATE parameter to keep the changes after the systems are initially loaded.

> **Note:** STARTBY cannot enforce the jobs to start at the specified time. It should not be used with the intention of meeting required SLAs.

Example 1-17 shows a sample job that was submitted at 11:00 and uses the STARTBY keyword to get the job running by 13:00 or earlier if the resources are available.

Example 1-17 Sample job that uses STARTBY keyword

```
//JOBB JOB TIME=NOLIMIT,REGION=OK,MSGCLASS=A,CLASS=A,NOTIFY=KWRES08
// SCHEDULE STARTBY=('13:00')
//STEP1 EXEC PGM=IEFBR14
//DD1 DD DSN=KWRES08.JGROUP1,DISP=(NEW,CATLG),SPACE=(TRK,(1,1)),
//   RECFM=FB,LRECL=80,BLKSIZE=0
```

WITH command

The **WITH** command is introduced in z/OS V2R2 to limit jobs to run only when the job or task that is specified on WITH keyword is active and on the same JES2 MAS member. This parameter assists users to make sure their jobs run on the same LPAR of their application and only if they are active.

Only one job or task name can be assigned to the WITH keyword. Coding more than one name on WITH parameter causes your job to fail with a JCL error. The job that uses the WITH keyword can be submitted before or after the reference job becomes active or even submitted. We suggest that you submit your job only after the reference job is active because other sequences cause JES2 to perform more processing.

In Example 1-18, we use the WITH keyword to ensure that our maintenance job runs on the same LPAR of our IBM DB2® started task.

Example 1-18 Use of WITH to run maintenance job on the same LPAR of DB2

```
//JOBA JOB TIME=NOLIMIT,REGION=OK,MSGCLASS=A,CLASS=A,NOTIFY=KWRES08
//   SCHEDULE WITH=(DB2AMSTR)
//STEP1 EXEC PGM=IKJEFT1A
```

The **WITH** command can be combined with JOBGROUP to improve scheduling capabilities.

1.5 JES3 JECL statements overview

Starting in z/OS V2R2, JES2 introduces the ability to process JES3 JECL control statements. A JCL/JECL language can be independent of the JES, simplify management, and provide the ability to turn off JECL statements when required.

JES2 now provides a new set of commands to control how JES2 treats JES3 JECL. You can ignore all JES3 JECL statements or selectively decide which commands are processed, ignored, or failed.

To control if JES2 processes or ignores JES3 JECL commands, you can use the new **$T INPUTDEF** command, as shown on Example 1-19. The default value is to IGNORE JES3 JECL. It also can be included on JES2 PARMLIB to activate JECL after an initial program load (IPL).

Example 1-19 INPUTDEF command syntax

```
$T INPUTDEF,JES3JECL=PROCESS | IGNORE
```

You can also define what specific control statements must be processed by JES2 by using **$T JECLDEF JES3** command. You also include this information in the JES2 PARMLIB to make it available after an IPL. All default values are underlined in Example 1-20.

Example 1-20 JECLDEF for JES3 commands

```
JECLDEF JES3=(
MAIN = PROCESS | IGNORE | WARN | FAIL
FORMAT = IGNORE | WARN | FAIL
ROUTE = IGNORE | WARN | FAIL
OPERATOR = IGNORE | WARN | FAIL
DATASET = IGNORE | WARN | FAIL
ENDDATASET = IGNORE | WARN | FAIL
PROCESS = IGNORE | WARN | FAIL
ENDPROCESS = IGNORE | WARN | FAIL
NET = IGNORE | WARN | FAIL
NETACCT = IGNORE | WARN | FAIL
PAUSE = IGNORE | WARN | FAIL
)
```

Note: At the time of this writing, only JES3 MAIN JECL is supported by JES2. Setting any other attribute to WARN has no effect and JES2 ignores the statements.

Optionally, you also can process or ignore JES2 control statements by using the **JECLDEF JES2** command, as shown in Example 1-21.

Example 1-21 JECLDEF for JES2

```
JECLDEF JES2=(
JOBPARM = PROCESS | IGNORE | WARN | FAIL
MESSAGE = PROCESS | IGNORE | WARN | FAIL
NETACCT = PROCESS | IGNORE | WARN | FAIL
NOTIFY = PROCESS | IGNORE | WARN | FAIL
OUTPUT = PROCESS | IGNORE | WARN | FAIL
PRIORITY = PROCESS | IGNORE | WARN | FAIL
ROUTE = PROCESS | IGNORE | WARN | FAIL
SETUP = PROCESS | IGNORE | WARN | FAIL
XEQ = PROCESS | IGNORE | WARN | FAIL
XMIT = PROCESS | IGNORE | WARN | FAIL
)
```

In both examples, the following options are available where applicable:

- PROCESS

  Statement is processed.

- IGNORE

  Statement is ignored (treated as a comment).

- WARN

  Statement is processed. However, a warning message is issued to record the occurrence of a JECL statement.

- FAIL

  An error message is issued and job is failed with a JCL error.

Example 1-22 shows the use of MAIN control card to change the specified CLASS in JCL.

Example 1-22 Sample MAIN control card

```
//JOBA JOB TIME=NOLIMIT,REGION=OK,MSGCLASS=A,CLASS=A,NOTIFY=KWRES08
//*MAIN CLASS=B
//STEP1 EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *

JOB00544  IRR010I  USERID KWRES08  IS ASSIGNED TO THIS JOB.
JOB00544  ICH70001I KWRES08  LAST ACCESS AT 12:55:03 ON TUESDAY, JUNE 2
JOB00544  $HASP373 JOBA     STARTED - INIT 1    - CLASS B        - SYS
JOB00544  Jobname  Procstep Stepname  CPU Time      EXCPs     RC
JOB00544  JOBA     --None-- STEP1     00:00:00         46     00
```

In addition to JECL control statements, you can define how JES2 must process null JCL statements. You can use the **$T INPUTDEF,NULLJCL=IGNORE|EOF** command to define whether JES2 must maintain its traditional behavior or consider it EOF, as JES3 does.

For exit updates, you can update EXIT 2 to control new JECL processing. For more information about EXIT updates for z/OS V2R2, see 1.8, “EXIT updates” on page 27.

1.6 RAS enhancements

There are several updates in JES2 to support system growth, increase the number of active jobs, and new functions. These changes are described next.

1.6.1 New $ACTIVATE level

Starting in z/OS V2R2, a new level of the JES2 checkpoint function can be activated. This new level is required for some new functions that are available in z/OS V2R2, including JOBGROUP and dynamic checkpoint data set resize.

Implementation and usage

With the introduction of z22 level, the old z2 level is no longer supported by JES2. If you use z2 level, we suggest you move to z11 before loading any systems on z/OS V2R2.

Before activating the new z22 checkpoint level, you must ensure that system requirements are met. You can check the current mode and the restrictions for the use of z22 by running the **$D ACTIVATE** command. Example 1-23 on page 22 shows the output of the **$D ACTIVATE** command, including the necessary actions before z22 is used.

Example 1-23   Sample $D ACTIVATE command

```
$HASP895 $DACTIVATE
$HASP895 JES2 CHECKPOINT MODE IS CURRENTLY Z11
$HASP895 THE CURRENT CHECKPOINT:
$HASP895  -- CONTAINS 2100 BERTS AND BERT UTILIZATION IS 14
$HASP895     PERCENT.
$HASP895  -- CONTAINS 409 4K RECORDS.
$HASP895 z22 CHECKPOINT MODE ACTIVATION WILL:
$HASP895  -- EXPAND CHECKPOINT SIZE TO 479 4K RECORDS.
$HASP895 z22 ACTIVATION WILL FAIL IF ISSUED FROM THIS MEMBER.
$HASP895     THE FOLLOWING ISSUES PREVENT ACTIVATION:
$HASP895  -- CYL_MANAGED SUPPORT MUST BE ACTIVATED.
```

To activate z22, you must check whether SPOOLDEF CYL_MANAGED keyword is ALLOWED; otherwise, you must change it to ALLOWED before the conversion is performed. You can change SPOOLDEF CYL_MANAGED status by running the **`$T SPOODEF,CYL_MANAGED=ALLOWED`** command.

After all the dependencies are satisfied, you can activate the new JES2 level. We suggest you perform this change in a maintenance window because any errors in the conversion process can result in outages. To activate the new level, run the **`$ACTIVATE,LEVEL=Z22`** command. Example 1-24 shows the output from **`ACTIVATE`** command.

Example 1-24   ACTIVATE command output

```
$ACTIVATE,LEVEL=Z22
$HASP895 z22 CHECKPOINT MODE IS NOW ACTIVE
$HASP895 $ACTIVATE,LEVEL=Z22 660
$HASP895 JES2 CHECKPOINT MODE IS CURRENTLY Z22
$HASP895 THE CURRENT CHECKPOINT:
$HASP895  -- CONTAINS 2100 BERTS AND BERT UTILIZATION IS 11
$HASP895     PERCENT.
$HASP895  -- CONTAINS 397 4K RECORDS.
$HASP260 MEMBER SC76 IS NOW IN z22 CHECKPOINT MODE 661
         NOTE: OPTSDEF COLD_START_MODE=z11 DOES NOT AGREE WITH THE
         CURRENT CHECKPOINT MODE
```

1.6.2 Increased number of jobs

As the systems grow, the JES2 must handle the increased number of jobs and tasks on production and test systems. To handle these needs, JES2 is enhanced to support up to 1,000,000 active jobs, handle up to 2,500,000 job output elements, and define up to 2,500,000 BERTs. The cache of available track groups increased to 1024.

Implementation and usage

To use the new limits that are available in JES2, you must first activate the new JES2 level, z22. After the system is activated on the new level, run the **`$T DEFJOB,JOBNUM=`** command to change the number of jobs that can be active at the same time. Example 1-25 on page 23 shows the output from the specified command.

Example 1-25 Sample $T JOBDEF,JOBNUM= output

```
$T JOBDEF,JOBNUM=2000
$HASP835 JOBDEF
$HASP835 JOBDEF  ACCTFLD=OPTIONAL,BAD_JOBNAME_CHAR=?,
$HASP835         CNVT_ENQ=FAIL,DEF_CLASS=A,INTERPRET=INIT,
$HASP835         CISUB_PER_AS=5,CNVT_SCHENV=IGNORE,JNUMBASE=552,
$HASP835         JNUMFREE=9627,JNUMWARN=80,JOBFREE=1628,
$HASP835         JOBNUM=2000,JOBWARN=80,PRTYHIGH=10,
$HASP835         PRTYJECL=YES,PRTYJOB=NO,PRTYLOW=5,PRTYRATE=0,
$HASP835         RANGE=(1,9999),RASSIGN=YES,JOBRBLDQ=NONE,
$HASP835         DUPL_JOB=DELAY,LOGMSG=ASIS,SUP_EVENTLOG_SMF=NO
```

> **Note:** Depending on the number that you define on JOBNUM, JOENUM, and BERTNUM, it might be necessary to increase the size of your checkpoint data set.

To change the JOENUM attribute, run the **$T OUTDEF,JOENUM=** command, as show in Example 1-26.

Example 1-26 Sample $T OUTDEF,JOENUM= output

```
$T OUTDEF,JOENUM=2000
$HASP836 OUTDEF
$HASP836 OUTDEF  COPIES=255,DMNDSET=NO,JOENUM=2000,JOEFREE=1664,
$HASP836         JOEWARN=80,OUTTIME=CREATE,PRTYLOW=0,
$HASP836         PRTYHIGH=255,PRTYOUT=NO,PRYORATE=0,SEGLIM=100,
$HASP836         STDFORM=STD,USERSET=NO,JOERBLDQ=NONE,
$HASP836         DSLIMIT=10M,LDEV_OPT=NO,SAPI_OPT=NO,WS_OPT=NO
```

If you want to increase your BERTNUM, run the **$T CKPTSPACE,BERTNUM=** command, as shown in Example 1-27.

Example 1-27 $T CKPTSPACE,BERTNUM= command

```
$T CKPTSPACE,BERTNUM=3000
$HASP852 CKPTSPACE
$HASP852 CKPTSPACE  BERTNUM=3000,BERTFREE=2746,BERTWARN=80,
$HASP852            CKPT1=(CAPACITY=888,UNUSED=489,TRACKS=75)
```

1.6.3 New DEBUG option

Starting with z/OS V2R2, a new DEBUG option is available. The option QVERIFY provides an option to drive regular job queue verifications. Although this option can cause high overhead, it can result in timely detection of queue errors and can be useful for debugging job, output, and Block Extension Reuse Table (BERT) queue errors in tests.

This new option is *not* activated by DEBUG=YES option. You must manually set it on by running the **QVERIFY=YES** command. The output of the **T DEBUG** command is shown in Example 1-28.

Example 1-28 Changing DEBUG QVERIFY to YES

```
$T DEBUG,QVERIFY=YES
$HASP827 DEBUG
$HASP827 DEBUG  BERT=YES,CKPT=NO,MISC=NO,SECURITY=NO,STORAGE=NO,
$HASP827        SYMREC=NO,VERSION=NO,VERBOSE=NO,
$HASP827        MEMBER_STATUS=NO,QVERIFY=YES,TIMECLOCK=NO
```

1.6.4 Updated $D PERFDATA(CKPTSTAT) command

The **$D PERFDATA(CKPTSTAT)** command was updated to provide more performance metrics. These metrics can be used to monitor and tune your environment.

Example 1-29 shows the output of the updated command.

Example 1-29 Sample $D PERFDATA(CKPTSTAT) output

```
$D PERFDATA(CKPTSTAT)
$HASP660 $DPERFDATA(CKPTSTAT)
$HASP660 CKPT PERFORMANCE STATISTICS SC74-INTERVAL=
$HASP660 21:28:20.207870,HOLD=0,AVGHOLD=0.000464,DORMANCY=(0,
$HASP660 100),AVGDORM=0.986298,TOT$CKPT=542661,WRITE-4K=160930,
$HASP660 WRITE-CB=0,OPT$CKPT=379528,OPT4K=0,
$HASP660 PAIN=2089575,PAINM=1583416,AVGXHOLD=0.000163,
$HASP660 IO=R1,COUNT=78330,AVGTIME=0.001313,
$HASP660 AVG_SUB_CPU=0.000023,AVG_SUB_TIME=0.001694,
$HASP660 AVG_SUB_IO=2,AVG_SUB_REC=15,
$HASP660 IO=R2,COUNT=76464,AVGTIME=0.000344,TOTAL4K=237619,
$HASP660 TOTALCB=0,
$HASP660 AVG_SUB_CPU=0.000006,AVG_SUB_TIME=0.000339,
$HASP660 AVG_SUB_IO=1,AVG_SUB_REC=1,
$HASP660 IO=PW,COUNT=0,AVGTIME=0.000000,TOTAL4K=0,TOTALCB=0,
$HASP660 AVG_SUB_CPU=0.000000,AVG_SUB_TIME=0.000000,
$HASP660 AVG_SUB_IO=0,AVG_SUB_REC=0,
$HASP660 IO=IW,COUNT=16,AVGTIME=0.001060,TOTAL4K=32,TOTALCB=0,
$HASP660 AVG_SUB_CPU=0.000019,AVG_SUB_TIME=0.001013,
$HASP660 AVG_SUB_IO=1,AVG_SUB_REC=4,
$HASP660 IO=FW,COUNT=78331,AVGTIME=0.000972,TOTAL4K=160898,
$HASP660 TOTALCB=0,
$HASP660 AVG_SUB_CPU=0.000014,AVG_SUB_TIME=0.000956,
$HASP660 AVG_SUB_IO=1,AVG_SUB_REC=4,
$HASP660 IO=FMT,COUNT=0,AVGTIME=0.000000,TOTAL4K=0,TOTALCB=0,
$HASP660 AVG_SUB_CPU=0.000000,AVG_SUB_TIME=0.000000,
$HASP660 AVG_SUB_IO=0,AVG_SUB_REC=0
```

1.7 Checkpoint improvements

There are a few checkpoint improvements to increase checkpoint limits and improve performance. These enhancements are described in this section.

1.7.1 64-bit CKPT processing

To support increased limits, checkpoint processing was reworked. This change should not affect the ability to control blocks (JQE, JOE, and so on). The I/O area was moved to 64-bit storage, which allows the use of larger amounts of memory while freeing memory space that is below the bar. Most DASD I/O processing was moved to a new subtask.

1.7.2 Reconfiguring CKPT data sets

In older releases of JES2, the process to reconfigure checkpoint data sets included reviewing complex dialog panels, even for simple changes. The size of a checkpoint data set cannot be altered without deleting it.

z/OS V2R2 introduces a new fast path checkpoint reconfiguration process and support for dynamic changing of checkpoint sizes, including ALTER processing on a Coupling Facility (CF), and extending a DASD data set into *adjacent free space*.

To validate the current checkpoint size, the **$D CKPTSPACE** command now includes several tracks for DASD data sets or current and maximum size of CF structure in 1 K blocks. Example 1-30 shows the new CKPTSPACE output, including space information.

Example 1-30 Sample $D CKPTSPACE output

```
$HASP852 CKPTSPACE
$HASP852 CKPTSPACE  BERTNUM=3000,BERTFREE=2746,BERTWARN=80,
$HASP852            CKPT1=(CAPACITY=888,UNUSED=489,TRACKS=75)
```

The **$T CKPTDEF** command was updated to include DSNAME, VOLSER, and STRNAME to define the checkpoint data set that is resized and SIZE and SPACE attributes to define the new checkpoint size.

When a checkpoint reconfiguration is performed, Fast Path reconfiguration is used, if necessary. It is not used if INUSE=NO is specified. This configuration change works with down-level members in the MAS and no writes to operator with reply (WTORs) are issued for the changes.

The **$T CKPTDEF** command is used in Example 1-31 to change the volume that the checkpoint data set is on. Although this change is *not* reflected in the command response, it is processed asynchronously. If the checkpoint is not in use, the change can occur without reconfiguration.

Example 1-31 $T CKPTDEF to change checkpoint data set volume

```
$T CKPTDEF,CKPT1=(VOL=CKPTPK)
$HASP829 CKPTDEF
$HASP829 CKPTDEF CKPT1=(DSNAME=SYS1.JESCKPT1,VOLSER=SPOOL1,
$HASP829 INUSE=YES,VOLATILE=NO),
:
$HASP285 JES2 CHECKPOINT RECONFIGURATION STARTING
$HASP233 REASON FOR JES2 CHECKPOINT RECONFIGURATION IS OPERATOR
REQUESTED SET COMMAND
```

```
$HASP285 JES2 CHECKPOINT RECONFIGURATION STARTED - DRIVEN BY
MEMBER IBM1
$HASP280 JES2 CKPT1 DATA SET (SYS1.JESCKPT1 ON CKPTPK) IS NOW IN USE
$HASP255 JES2 CHECKPOINT RECONFIGURATION COMPLETE
```

When CF structures are set INUSE=NO, the CKPT reconfiguration was altered to force the structure, which deletes it from CF.

If the reconfiguration fails (as shown in Example 1-32), no WTORs are issued that request alternative actions. The reconfiguration is not performed. If there is a typographical error in the data set name, the new data set is created.

Example 1-32 Failed CKPT reconfiguration

```
$TCKPTDEF,CKPT1=VOL=UNKNOW
$HASP829 CKPTDEF
$HASP829 CKPTDEF CKPT1=(DSNAME=SYS1.JESCKPT1,VOLSER=CKPTPK,
$HASP829 INUSE=YES,VOLATILE=NO),
:
$HASP285 JES2 CHECKPOINT RECONFIGURATION STARTING
$HASP233 REASON FOR JES2 CHECKPOINT RECONFIGURATION IS OPERATOR
REQUESTED SET COMMAND
$HASP285 JES2 CHECKPOINT RECONFIGURATION STARTED - DRIVEN BY
MEMBER IBM1
$HASP424 MEMBER IBM1 -- UNKNOW IS NOT MOUNTED
$HASP255 JES2 CHECKPOINT RECONFIGURATION FAILED
```

1.7.3 Resizing checkpoint data set

To alter the checkpoint size, you can use new keywords SPACE and SIZE for DASD and CF structures. The checkpoint resize function is available only when JES2 level z22 is run. The following keyword options are available:

- SPACE=(TRK|CYL|MAX,nnnn):
  - TRK: Define the space allocation is specific in tracks
  - CYL: Define the space allocation is specific in cylinder
  - MAX: Specify to use the largest CKPT or all free space available, whichever is smaller
  - nnnn: The size of the CKPT data set, in CYLs or TRKs
- SIZE=nnnn|MAX:
  - nnnn: Size of the CKPT CF structure, in 1K blocks
  - MAX: Largest CKPT or all the policy limit, whichever is smaller

A CKTP DASD data set resize is shown in Example 1-33.

Example 1-33 Sample CKTP DASD data set resize

```
$DCKPTSPACE,CKPT1
$HASP852 CKPTSPACE CKPT1=(CAPACITY=1788,UNUSED=1566,TRACKS=150)
$TCKPTDEF,CKPT1=SPACE=(TRK,160)
$HASP829 CKPTDEF
$HASP829 CKPTDEF CKPT1=(DSNAME=SYS1.JESCKPT1,VOLSER=CKPTPK,
$HASP829 INUSE=YES,VOLATILE=NO),
:
$HASP740 Volume CKPTPK Data set SYS1.JESCKPT1 Extend successful.
$DCKPTSPACE,CKPT1
```

```
$HASP852 CKPTSPACE CKPT1=(CAPACITY=1908,UNUSED=1686,TRACKS=160)
```

> **Note:** Ensure that the INITSIZE in the new policy for the structure is at least large enough to hold the current usage. If it is not large enough, you cannot bring it INUSE.

1.7.4 JES2 checkpoint tuning

JES2 MASDEF parameters DORMANCY and HOLD are defined by system programmers to determine the checkpoint usage by MAS members. This definition prevents a single MAS member from monopolizing the checkpoint data set. These options often are not tuned, unless a performance issue is identified.

z/OS V2R2 introduces a new management capability for these two parameters where JES2 is responsible for maintaining its values. Allowing JES2 to control the values for DORMANCY and HOLD can result in overall performance improvements that are based on system workload and configuration.

By allowing JES2 to control these parameters, you no longer can modify them from SDSF MAS panels. To enable automatic checkpoint cycle management, run the **`$T MASDEF,CYCLEMGT=AUTO`** command. The output from the **`MASDEF`** command is shown in Example 1-34.

Example 1-34 Change cycle management to manual

```
RESPONSE=SC74
 $HASP843 MASDEF
 $HASP843 MASDEF  OWNMEMB=SC74,AUTOEMEM=ON,CKPTLOCK=ACTION,
 $HASP843         COLDTIME=(2005.182,17:42:57),COLDVRSN=z/OS 1.7,
 $HASP843         ENFSCOPE=SYSPLEX,DORMANCY=(20,300),HOLD=40,
 $HASP843         LOCKOUT=1000,QUESHELD=SC74,RESTART=YES,
 $HASP843         SHARED=NOCHECK,SYNCTOL=120,
 $HASP843         WARMTIME=(2015.174,19:40:58),XCFGRPNM=WTSCPLX7,
 $HASP843         QREBUILD=0,CYCLEMGT=AUTO
```

You can turn off automatic management at any time. If automatic management is turned off, the DORMANCY and HOLD values are reset to the values that were used before automatic management was enabled. To turn off this feature, run the **`$T MASDEF,CYCLEMGT=MANUAL`** command.

1.8 EXIT updates

There are several changes to JES2 exits to accommodate new functions. These changes are described in this section.

1.8.1 EXIT 2 and EXIT 52

EXIT 2 and EXIT 52 were updated to include new 12-byte fields in the $XPL. The traditional 8-byte fields are still available and are the first 8 bytes. This change is necessary to accommodate longer JES2 and JECL statements, such as CONCURRENT. This JCL is *not* passed to the converter.

A new JOBGROUP job type can be processed by EXIT 2 and EXIT 52. The JOBGROUP acts as a job in these exits. Exits see the new JCL and must act. Bits JQE (JQE3DFJG), JCT (JCT6DFJG), and JRW (JRW1GROP) were updated to reflect these changes. For more information about JOBGROUPs, see 1.2, “JOBGROUP” on page 2.

Changes to EXIT 2 and EXIT 52 also might be necessary to control JES2 processing of JES2 and JES3 JECL statements. You can use these exits to control JECL processing job by job. For more information about JECL processing, see 1.5, “JES3 JECL statements overview” on page 19.

1.8.2 EXIT 4 and EXIT 54

EXIT 4 and EXIT 54 were updated to include 12-byte fields in $XPL. This change allows longer JES2 and JECL statements, such as CONCURRENT. The traditional 8-byte fields are still available and are the first 8 bytes.

If JES JECL processing is activated on your environment, EXIT 4 and EXIT 54 might need to be updated to treat //*MAIN as JECL. Operands can be processed by these exits. For more information about JECL processing, see 1.5, “JES3 JECL statements overview” on page 19.

1.8.3 EXIT 32 and EXIT 49

These exits can be affected by the use of concurrent execution from JOBGROUPs. Exits cannot reject one job of a concurrent set. This issue can limit what EXIT 32 can do and affect EXIT 49. For more information about concurrent execution and JOBGROUPs, see 1.2, “JOBGROUP” on page 2.

1.8.4 EXIT 51

Because the SETUP queue is now used by JES2, there might be instances in which a job follows a different order of processing, as shown in the following example:

SETUP → EXEC → SETUP → EXEC

This order can be caused by concurrent jobs that include unmatched affinity, concurrent jobs that WLM is delaying execution, and a master job in a concurrent set. ENF 78 is not affected by this change because it is issued after the job moves beyond running and cannot return.

If JOBGROUPs are in use, update EXIT 51 to handle these situations. For more information about SETUP queue, see “SETUP queue” on page 11.

2

JES3

This chapter describes the enhancements that are provided in JES3 by z/OS V2R2 and includes the following topics:

- 2.1, “JES3 support for OUTDISP statement” on page 30
- 2.2, “JES3 modifications because of OUTDISP option” on page 31
- 2.3, “JES3 enhancements for symbols” on page 34
- 2.4, “JES3 improvements in use of symbols” on page 35
- 2.5, “Data Set Integrity for JES3 data sets” on page 36

2.1 JES3 support for OUTDISP statement

The OUTDISP JES2 option is implemented at JES3 that is running in a z/OS V2R2 system.

Allocating a data set to a task allows that task's programs to access the data set.

For allocation purposes in JCL, you must set the disposition (DISP) of the data set, which is to describe the status of a data set (before the step is run) and to set at deallocation (step end) what is the final status. You can specify one disposition for normal step end and another for abnormal step end.

This logic also applies to sysout data sets. The OUTDISP option was used to determine JES2 about the output disposition for a sysout data set when it was created. The OUTDISP option for JES2 can be declared by using one of the following options:

- OUTDISP parameter on the OUTPUT JCL statement
- OUTDISP parameter of the TSO/E OUTDES command

  This method is used to create an output description that is used by the **`TSO/E ALLOCATE`** command.
- OUTADD macro option.

  This option is used to create an output description that is used with the DYNALLOC macro.

The OUTADD syntax is described next.

2.1.1 OUTDISP=(normal disposition, abnormal disposition)

The following subparameters are used:

- The normal disposition subparameter specifies the disposition for the output if the job completes normally. If not specified, the default is WRITE.
- The abnormal disposition subparameter specifies the disposition for the output if the job does not complete successfully. If not specified, the default is equivalent to normal disposition.

The following options are available:

- WRITE: Prints the sysout data set. After printing, JES2 purges it.
- HOLD: Holds the sysout data set until the user or operator releases it. Releasing the sysout data set changes its disposition to WRITE.
- KEEP: Prints the sysout data set. After printing the data set, JES2 changes its disposition to LEAVE.
- LEAVE: Output is not available to be processed until you change the disposition to WRITE or KEEP, or release the output. When the output is released, the disposition changes to KEEP.
- PURGE: Deletes the sysout data set without printing it.

2.2 JES3 modifications because of OUTDISP option

Before z/OS V2R2, JES3 handled sysout with no output disposition.

With z/OS V2R2, JEs3 supports the OUTDISP parameter on JES3 SYSOUT initialization statement, as does JES2.

If this option is not declared, there are no changes to the behavior that JES3 handles the sysout data set disposition.

2.2.1 JES3 processing of the OUTDISP option

In this section, we describe the effect of the OUTDISP option at JES3 handling of a sysout data set.

JES3 handling of a sysout data set not held for an external writer

In this case, the following options are available:

- WRITE output disposition, the sysout data set is:
  - Placed on Q=WTR
  - Not made available via the `TSO/E OUTPUT` command
  - Purged after processing by a printer or writer
- KEEP output disposition, the data set:
  - Is placed on Q=WTR
  - Is not made available via the `TSO/E OUTPUT` command
  - After processing by a printer or writer, the disposition of the sysout data set is changed to LEAVE and moved to Q=HOLD
- HOLD output disposition, the data set:
  - Is placed on Q=HOLD
  - Is made available via the `TSO/E OUTPUT` command
  - Can be released to the writer service queue (Q=WTR) or purged

This processing is the same processing as HOLD=TSO for the JES3 sysout class:

- LEAVE output disposition, the data set is:
  - Placed on Q=HOLD
  - Made available via the `TSO/E OUTPUT` command
  - Not available for processing until released and the output disposition is changed to KEEP or WRITE
- PURGE output disposition, the data set is not placed on any service queue and is deleted.

JES3 handling of a sysout data set held for an external writer

In this case, the following options are available:

- WRITE output disposition, the data set is:
  - Not made available via the `TSO/E OUTPUT` command
  - Available for processing by an external writer by using PSO or SAPI
  - Released after processing, it is purged (it is not moved to Q=WTR)

    If no output disposition is available, it is moved to Q=WTR.

- KEEP output disposition, the data set is:
  - Not made available via the **TSO/E OUTPUT** command
  - Available for processing by an external writer by using PSO or SAPI
  - Released after processing; the disposition of the sysout data set is changed to LEAVE and remains on Q=HOLD (data set held for external writer)
- HOLD output disposition, the data set is:
  - Made available via the **TSO/E OUTPUT** command
  - Not available for processing by an external writer until the output disposition is changed to KEEP or WRITE

    Override is possible with new SAPI flag covered later.
- LEAVE output disposition, the data set is:
  - Made available via the **TSO/E OUTPUT** command.
  - Not available for processing by an external writer until the output disposition is changed to KEEP or WRITE

    Override possible with new SAPI flag covered later.
- PURGE output disposition, the data set is not placed on any service queue and is deleted.

2.2.2 New options for JES3 commands supporting OUTDISP option

The following options are available:

- **JES3 *INQUIRY,U** command

  OUTDISP= parameter added:
  - H | HOLD or K | KEEP or L | LEAVE or W | WRITE or N | NONE or ?
  - Displays the output disposition of selected data sets in the output queue (?), or displays information about data sets with the specified output disposition.
- **JES3 *MODIFY,U** command:
  - OUTDISP= parameter added:
    - H | HOLD or K | KEEP or L | LEAVE or W | WRITE or N | NONE or ?
    - Displays the output disposition of selected data sets in the output queue (?), or displays information about data sets with the specified output disposition.
  - NOUTDISP= parameter added:
    - H | HOLD or K | KEEP or L | LEAVE or P | PURGE or W | WRITE
    - Modifies the output disposition of the selected data sets.

2.2.3 Process SYSOUT Data Sets Call (PSO) – SSI 1

The following modifications are available at SSI 1 Process SYSOUT (PSO) because of the new OUTDISP option:

- The PSO interface is stabilized and changes are limited to those changes that are necessary to maintain the behavior for compatibility purposes.

  We recommend that the more robust SSI79 SAPI interface is used for processing sysout data sets. Therefore, any updates that are made because of output disposition does not adversely affect behavior.

- To maintain complete compatibility with the behavior, output disposition is ignored for sysout selection, delete requests (SSSODELC), and release requests (SSSORLSE).

  The only change that is made is that a release request can result in an update of the output disposition.

2.2.4 SYSOUT Application Program Interface – SSI 79

The following modifications are available at SSI 79 SYSOUT Application Program Interface (SAPI) because of the new OUTDISP option:

- The SAPI was updated with the addition of output disposition.

  Updates are based on support that is provided by JES2 so that applications obtain similar behavior with JES3.

- Data set selection flags within input field SSS2SEL1 and SSS2SEL6 are used to determine from which queue the data sets are selected for GET processing.

  Data set selection was updated based on the current output disposition of the data set.

- Data set return flags within output field SSS2RET5 indicate on which queue the returned data set is kept.

  Returned flags were updated to reflect the current output disposition of the returned data set.

- SAPI disposition flags within input field SSS2DSP1 are an indication from the user as to what is to be done with the data set that is being PUT.

  The final disposition of the data set is now based on the SAPI disposition that us specified by the user and the current output disposition of the data set.

- The behavior of all flags is maintained for sysout data sets that do not have an output disposition (that is, data sets that were created with an output disposition of none).

2.2.5 Extended Status Function Call – SSI 80

The following modifications are available at SSI 80 because of the new OUTDISP option:

- JES3 return the output disposition (STSTDISP) for sysout data sets that have an output disposition.
- JES3 handling of sysout data set filters in STATSSL1 is unchanged (that is, not affected by output disposition).
- JES3 added support for all sysout data set filters in STATSSL3.

  Includes selection of data sets based on output disposition.

2.2.6 NJE header flags

The following modifications are available at NJE Header Flags because of the new OUTDISP option:

- JES3 added support for flags in the networking data set header that specify output disposition NDHGF2HB and NDHGF2HA.
- Flags are set for outgoing networked sysout data sets that have an output disposition.
- Flags are used to establish the output disposition of incoming networked sysout data sets.

  OUTDISP is always set for incoming NJE sysout.

2.2.7 JES3 installation exits

The following modifications are available at JES3 Installation exits because of the new OUTDISP option:

- Output disposition is maintained in the data set entries of the Output Service Element (OSE) and is available to installation exits that provide a pointer to the OSE.
- The following exits provide OSE data via a parameter list that is mapped by IATYSSX:
  - IATUX58 = Modify Security Information Before JES3 Security Processing
  - IATUX59 = Modify Security Information After JES3 Security Processing

  Exits are used for PSO and SAPI processing.

  IATYSSX was updated to include the output disposition of the sysout data set.

2.2.8 Migration and coexistence considerations

Complete JES3 OUTDISP support is available with the JES3 global main and any CIFSSes at the V2R2 level only.

OUTDISP-related parameters are ignored or invalid at previous levels.

JES3 release toleration APAR OA43563 is required for systems in the JESplex that are at JES3 V2R1 or V1R13.

The OUTDISP parameter is tolerated on the SYSOUT statement in the JES3 initialization stream.

2.2.9 Benefits and value

Instead of cluttering spools with test and development batch sysout, OUTDISP support allows sysout data sets to be purged or printed as necessary.

2.3 JES3 enhancements for symbols

In this section, we describe the symbols that are used in a z/OS system. The use of such symbols is improved in JES3 that is running in a z/OS V2R2.

Symbols are strings of up to eight characters that represent variable information. The symbols allow you to modify and customize z/OS statements easily. Symbols act as variables in a program. They also can take on different values based on the input to the program.

The following types of symbols are used:

- System symbols

  These symbols represent values that have a unique value in each z/OS system. z/OS system replaces those symbols with its own values. For example, system symbols allow that z/OS systems can share parmlib definitions while retaining unique values in those definitions.

  When you specify a system symbol in a shared parmlib definition, the system symbol acts as a place holder. Each system that shares the definition replaces the system symbol with a unique value during initialization.

You can use these symbols when it processes a started task JCL and TSO log ons, for example. System Symbols are defined in MVS or by your installation at IEASYMxx parmlib member.

- JCL symbols

  These symbols must be defined before you can use them in that JCL. The JCL symbols that you define are valid for the current job only. By default, JCL symbols are available only to the job at the converter phase and are lost when the job runs. The substitution effect of the JCL symbols is shown in Example 2-1.

Example 2-1 JCL proc with JCL symbols and the final JCL result

```
//EXAMPLE PROC SYM1=”Whats up, Doc?”,SYM2=(DEF),SYM3=&&&&TEMP1,
// SYM4=”&&TEMP2”,SYM5=&&TEMP3,TEMP3=TEMPNAME,
// SYM6=&TEMP3
//S1 EXEC PGM=WTO,PARM= ”&SYM1”,ACCT=&SYM2
//DD1 DD DSN=&SYM3,UNIT=SYSDA,SPACE=(TRK,(1,1))
//DD2 DD DSN=&SYM4,UNIT=SYSDA,SPACE=(TRK,(1,1))
//DD3 DD DSN=&SYM5,UNIT=SYSDA,SPACE=(TRK,(1,1))
//DD4 DD DSN=&SYM6,UNIT=SYSDA,SPACE=(TRK,(1,1))
// PEND

The final JCL substitution:
//S1 EXEC PGM=WTO,PARM=”Whats up, Doc?”,ACCT=(DEF)
//DD1 DD DSN=&&TEMP1,UNIT=SYSDA,SPACE=(TRK,(1,1))
//DD2 DD DSN=&&TEMP2,UNIT=SYSDA,SPACE=(TRK,(1,1))
//DD3 DD DSN=&TEMP3,UNIT=SYSDA,SPACE=(TRK,(1,1))
//DD4 DD DSN=&TEMP3,UNIT=SYSDA,SPACE=(TRK,(1,1))
```

  However, by using the EXPORT and SET JCL statements, JCL symbols can be made available to the job running phase. Exported JCL Symbols can be accessed during the job running phase by using the JCL Symbol Service (IEFSJSYM) or the JES Symbol Service (IAZSYMBL).

- JES symbols

  These symbols are dynamic symbols that can be managed by using the JES Symbol Service (IAZSYMBL).

2.4 JES3 improvements in use of symbols

With this new JES3 support at z/OS V2R2, the following tasks can be performed:

- Use symbols in the in-stream (DD * data set) data in the same manner as they are used in JCL stream of the job. In this way, the JCL of the job and the in-stream data that is passed to the application can view the same set of symbols.
- Programmatically access JCL symbols that are defined in job’s JCL stream at job run time.
- Dynamically create JES symbols via the JES Symbol Service.
- Instruct the internal reader to pass JCL and JES symbols to a submitted job. Such symbols can be used for symbol substitution in the JCL stream of the submitted job.
- Provide a JES-independent interface to identify when a job finishes running. The application can request by defining a special JES symbol that JES must send notification when a particular job completes or is no longer eligible for running. This notification is provided by Event Notification facility (ENF) 78.

All of these z/OS V2R2 JES3 symbol enhancements were available for JES2 at z/OS V2R1, except for JES3, which does not support Job Correlator functions or the following JES symbols:

- SYS_CORR_USERDATA
- SYS_CORR_CURRJOB
- SYS_CORR_LASTJOB

For more information about these enhancements, see the following resources:

- *Application Programming Guide*, SA32-0987
- *MVS JCL Reference*, SA23-1386

2.5 Data Set Integrity for JES3 data sets

In this chapter, we review the capability installations include with z/OS JES3 V2R2 to enable Data Set Integrity (DSI) for the JES3 data sets, which is not to be confused with the JES3 DSI facility; that is, Dynamic System Interchange.

There is a data integrity and availability issue with the major JES3 data sets, such as SPOOL, JCT, and CKPT. Only SAF authorized users on the system can inadvertently scratch these data sets or write over the data set space while JES3 is running.

The simple solution at z/OS V2R2 is to use the Program Properties Table (PPT) in a SCHEDxx parmlib member to specify DSI for JES3 programs. The JES3 data sets then are protected by the ENQ serialization mechanism.

2.5.1 DSI on z/OS overview

DSI means that the Job step holds an ENQ shared or exclusive for the data sets it allocates. The DSI/NODSI option at SCHEDxx applies to batch allocation only. The DSI is the default.

Dynamic allocation uses its input parameters to determine whether to enqueue on data sets.

If DSI is specified, the initiator through allocation acquires an ENQ for all data sets that are requested by the program in the step. The ENQ major name is SYSDSN and the minor name is the name of the data set. The ENQ is exclusive or shared, depending on the disposition on the DD request. Consider the following points:

- Exclusive for OLD, NEW, and MOD
- Shared for SHR

The installation can specify WAITALLOC(NO/YES) on the SDSN_WAIT keyword in the ALLOCxx member of parmlib to specify the installation policy for batch jobs that must wait to enqueue.

If NODSI is specified, the initiator still issues an ENQ for all data sets that are requested by the program. However, the ENQ is released before the problem program in the step is started.

The justification for NDSI is clarified by the output that is shown Example 2-2 on page 37, in which the Master Scheduler JCL procedure is shown. If in PPT the option DSI is activated for such program, no other code can update such a data set.

Example 2-2 Portion of the Master Scheduler JCL procedure

```
//MSTJCL00 JOB MSGLEVEL=(1,1),TIME=1440
// EXEC PGM=IEEMB860
//IEFPDSI   DD DSN=SYS1.PROCLIB,DISP=SHR
```

2.5.2 Activating DSI in JES3

To protect the JES3 data by using ENQs, complete the following tasks:

- Specify DSI for JES3 programs in a SCHEDxx member:
  - PPT PGMNAME(IATINTK) NOCANCEL NOSWAP SYST DSI KEY(1)
  - PPT PGMNAME(IATINTKF) NOCANCEL NOSWAP SYST DSI KEY(1)
- Review the JES3 cataloged start procedure to make sure all data sets that are allocated use DD statements specify DISP=SHR.

  If DD allocations are used in the JES3 procedure (not recommended), DISP=OLD should not be specified. Otherwise, the first JES3 that is started locks out other JES3 address spaces in the same Sysplex.
- Check your JES3 start procedure and your DSN specification on the DYNALLOC statement in your initialization deck for duplicate data set names.

  Because an ENQ is held on the data set names, it is recommended that you use unique spool data set names to allow for offline allocation, deallocation, and formatting.
- Use one of the following tasks to activate DSI:
  - Initially load with the updated SCHEDxx member in use and start JES3.
  - Use the SET SCH command with the updated SCHEDxx member to change the settings. Restart JES3 to pick up the changes. Consider the following points:
    - Any JES3 restart picks up the changes. In this case, no IPL is required.
    - Backout plans to return to NODSI without an IPL.

In z/OS V2R2, JES3 code is updated to set DSI for ALL dynamic allocations if DSI is set in the PPT. Shared ENQ for data sets that are identified in the initialization deck and dynamically allocated by JES3 always allocate as DISP=SHR.

2.5.3 New JES3 health check

At z/OS V2R2, a health check (IBMJES3,JES3_DATASET_INTEGRITY) determines whether DSI or NODSI was specified on the JES3 entries in the PPT. The check generates an exception message when the current DSI setting does not match the specified setting.

The following default keywords are available for the check that you can override on a POLICY statement in the HZSPRMxx parmlib member or on a **`MODIFY`** command:

- UPDATE
- CHECK(IBMJES3,JES3_DATASET_INTEGRITY)
- SEVERITY(LOW)
- INTERVAL(ONETIME)
- PARM('DSI(YES)')
- DATE('date_of_the_change')
- REASON('your update reason')

3

System Display and Search Facility

This chapter describes the changes in the System Display and Search Facility (SDSF) in z/OS V2R2 and includes the following topics:

3.1 SDSF enhancements

The following SDSF enhancements are available in z/OS V2R2:

- JJE component elimination
- zIIP exploitation
- System command improvements
- UI enhancements
- JOB Display enhancements
- Batch parallelism new panels
- REXX enhancements
- Miscellaneous changes

Since z/OS V1R10, SDSF had a second component (JJExxxS) that included parts that required JES2 control blocks, such as LOG, non-RMF DA, and JES2 offset table. This configuration increased the complexity of SDSF installations.

Starting with z/OS V2R2, the need for a second JJExxxS component was eliminated. The SDSF V2R2 is installed as a single FMID; that is, HQX77A0. All JES data is now obtained via interfaces (such as the SSI) rather than by traversing JES2 control blocks.

3.1.1 Implementation and usage

SDSF is now required to be installed in the SMP/E BCP zone. The ServerPac option for SDSF SMP/E zone only was removed.

Also, the JES2 dependent feature JJE77xS was removed in SDSF V2R2. It was used to provide means for reassembling SDSF when JES2 macros were changed. SDSF no longer uses these macros, and reassembly is no longer necessary. As the reassembly is no longer necessary, UCLIN and reassembly sample jobs were deleted.

The following JJE77xS related data sets also are no longer necessary and can be removed when systems are moved to z/OS V2R2:

- ISF.SISFJCL1/ISF.AISFJCL1
- ISF.SISFMOD1/ISF.AISFMOD1
- ISF.SISFSRC1/ISF.AISFSRC1

The SISFMOD1 also must be removed from the linklist.

To support the JJE77xS data set removal, the ISFPARMS were moved from JJE77xS to HQX77A0. If you modify ISFPARMS, update your SMP/E apply job to specify the correct FMID. We suggest you use ISFPRMxx instead of ISFPARMS.

HASPINDX removal

HASPINDX was used to chronologically order JES2 syslog data sets, and it is no longer used in SDSF V2R2. The HASPINDX keywords are now obsolete and are ignored. Verify that your logon procedure or initial clist does not ALLOC FI(HASPINDX) because it is no longer used.

If you share the HASPINDX data set, you can delete it after all z/OS images are running V2R2.

3.2 zIIP use

Some SDSF activities can become CPU-intensive, depending on the activity that is performed and the amount of data that is involved. Performing a sort on SDSF panels with tens of thousands jobs can increase CPU usage. Now, SDSF uses zIIP processor to perform some CPU-intensive tasks.

There is no user action necessary to use zIIP on large sorts. SDSF determines whether the action is eligible for zIIP processing that is based on the number of entries to be sorted. A number of 1,000 entries or more is eligible for zIIP processing. If zIIP processor is not available, regular CPU processor is used.

3.3 System command improvements

The System Command Extension pop-up window provides a simple way to retrieve commands that were issued by the user. To access this panel, issue a slash from any SDSF panel.

The window was enhanced to provide a better user experience while increasing the number of saved commands and allowing grouping and commenting commands. In this section, we describe these features with examples and provide suggestions about how to use these enhancements to improve support activities.

3.3.1 Increasing the number of the saved slash commands

The number of 20 saved commands might not be helpful for users who constantly issue different slash commands. For this reason, the standard number of saved slash commands was increased to 50, which is the default value on z/OS V2R2. There is no user action necessary to use this feature.

If you must store more than 50 commands, you can define a data set and allocate it to ISFTABL on your logon procedure to increase the number of saved commands.

The ISFTABL data set must be a partitioned data set (PDS) or PDS-e with logical record of 80 bytes and record format FB. Each saved command uses up to 500 bytes of space. The default number of saved commands is 1,000, with a limit of 2,000.

Example 3-1 uses an IEFBR14 to allocate a sample ISFTABL data set. Define space attributes according to your needs.

Example 3-1 Define ISFTABL data set

```
//JOBD JOB TIME=NOLIMIT,REGION=OK,MSGCLASS=A,CLASS=A,NOTIFY=&SYSUID
//STEP1 EXEC PGM=IEFBR14
//ALOC  DD  DSN=KWRES08.ISFTABL,
//      DISP=(,CATLG),SPACE=(TRK,(60,15,50)),RECFM=FB,
//      LRECL=80,BLKSIZE=0
```

Migration and coexistence

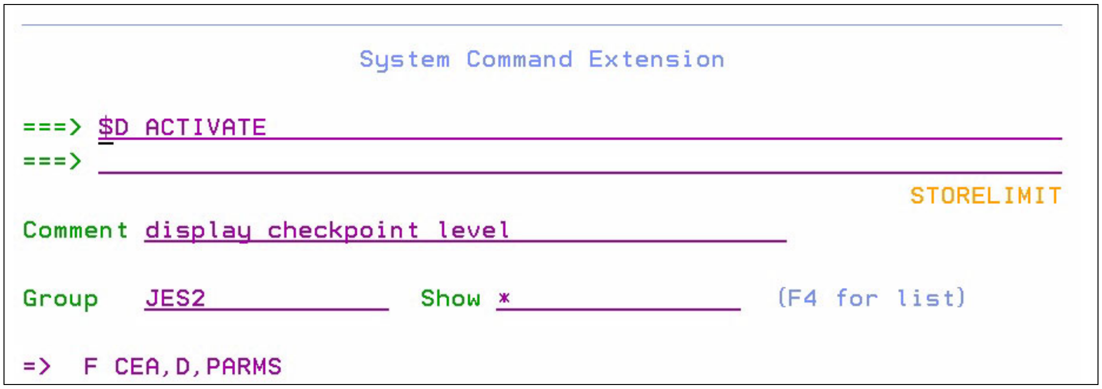
If you have a mixed environment with previous z/OS releases, only the last 20 commands are available in these z/OS images. If a new command is issued from these LPARs and the 20th command is excluded from the list on all LPARs.

3.3.2 Grouping and viewing saved commands

With the increase of saved commands, it is necessary to provide a way to manage and use the saved commands. SDSF is enhanced to allow grouping commands to improve management and usability. Grouping is available by default and is optional. Any command that does not have a group name that is associated with it is considered ungrouped.

You can use grouping options to associate a set of commands that perform a similar function, or control the same application. Then, you can use the group name to display only the commands that are associated with that group. You can create multiple groups,

Figure 3-1 shows the system command panel that is used to define a command to JES2 group.

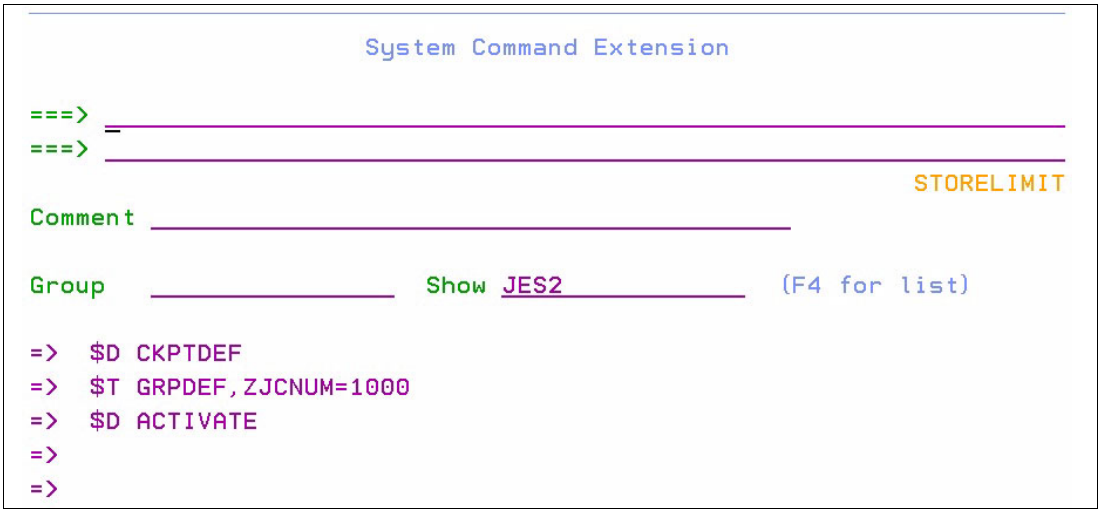


```
                     System Command Extension

===> $D ACTIVATE
===>
                                                          STORELIMIT
Comment display checkpoint level

Group   JES2             Show *                (F4 for list)

=>  F CEA,D,PARMS
```

Figure 3-1 Define a command to JES2 group

> **Note:** It is no longer necessary to submit the command to save it. You can press F10 to save the command without submitting it. Submitting the command also saves it on your list.

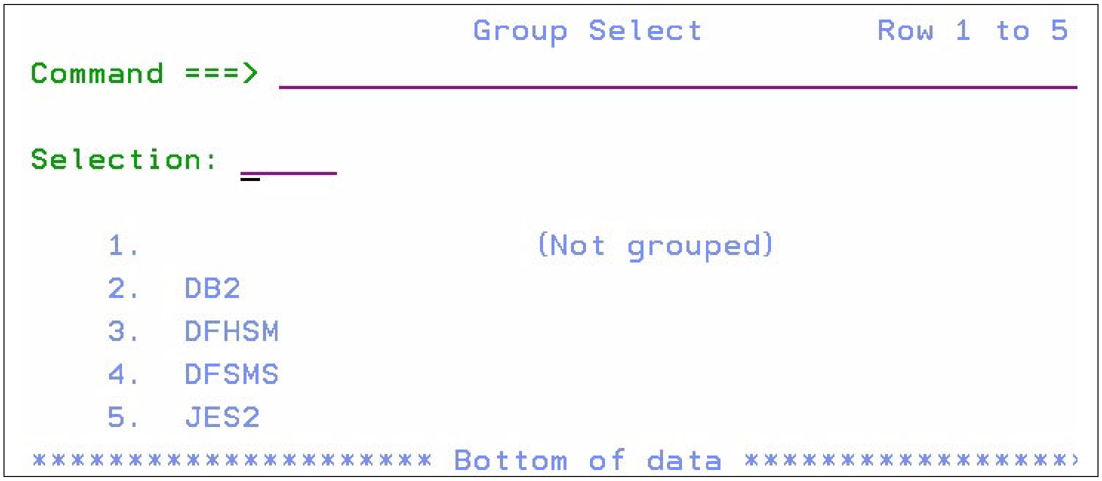
After the command is saved, you can use the Search field to display only the commands that are associated to this group. As shown in Figure 3-2, we used the keyword JES2 in the Search field to retrieve JES2 group commands only.



```
                     System Command Extension

===>
===>
                                                          STORELIMIT
Comment

Group                    Show JES2             (F4 for list)

=>  $D CKPTDEF
=>  $T GRPDEF,ZJCNUM=1000
=>  $D ACTIVATE
=>
=>
```

Figure 3-2 Display JES2 grouped commands only

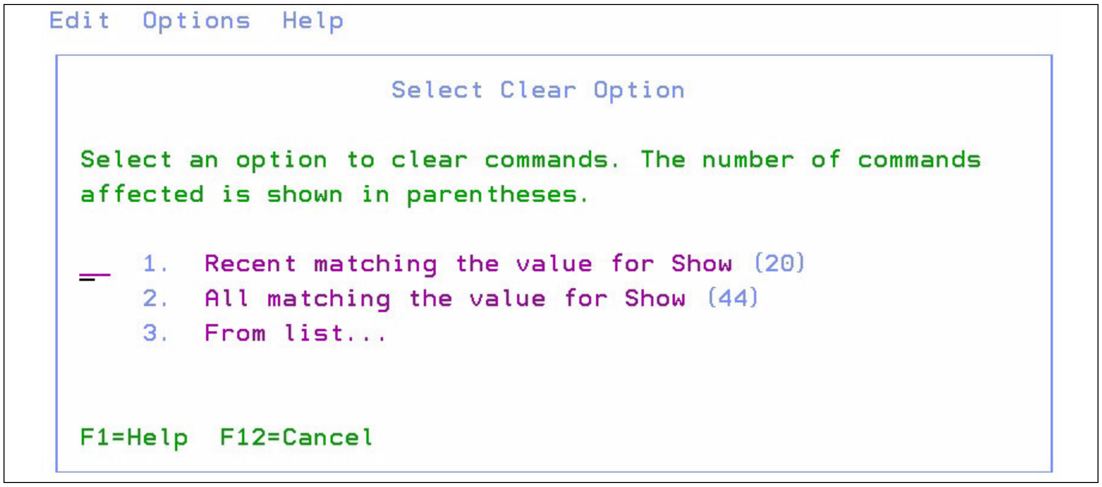
Pressing F4 when the cursor is at the group or show options opens a pop-up window in which a list of all groups is defined for selection. Within this panel, option 1 is used for non-grouped commands, as shown in Figure 3-3.



```
                        Group Select          Row 1 to 5
Command ===> __________________________________________

Selection: _____

    1.                    (Not grouped)
    2.   DB2
    3.   DFHSM
    4.   DFSMS
    5.   JES2
********************* Bottom of data ****************
```

Figure 3-3 Display groups

Commands also can be deleted from a saved list. You can press F11 to clear these entries. A new pop-up panel prompts you to select between the 20 most recent commands, all commands, or to select the commands from a list, as shown in Figure 3-4.



```
                  Select Clear Option

Select an option to clear commands. The number of commands
affected is shown in parentheses.

__  1.  Recent matching the value for Show (20)
    2.  All matching the value for Show (44)
    3.  From list...


F1=Help  F12=Cancel
```

Figure 3-4 Clear commands from display

We suggest that you create different groups for each type of commands. You can create a group that is named D-JES to store display JES commands, such as the commands that described in Chapter 1, “JES2” on page 1. You can also create groups for other applications or purposes, such as ‘stop-applx’, and save all commands that are required to correctly stop application X.

We also encourage you to include comments to your commands. The steps to include these comments are described next.

Migration and coexistence

Slash commands that are saved on early systems are considered ungrouped when processed by V2R2 system. Ungrouped commands on V2R2 system visible to early systems.

3.3.3 Including comments to saved commands

In addition to increasing the number of saved commands and grouping and searching commands, SDSF now also gives you the ability to include comments to your saved commands. This function can be useful to collect more information when a command is retrieved from the saved list.

The comments about saved commands are displayed when the command is retrieved for use. The comment is not displayed on syslog or the user log when the command is issued. You can use comments and groups to create a more meaningful view of groups and commands and improve commands management.

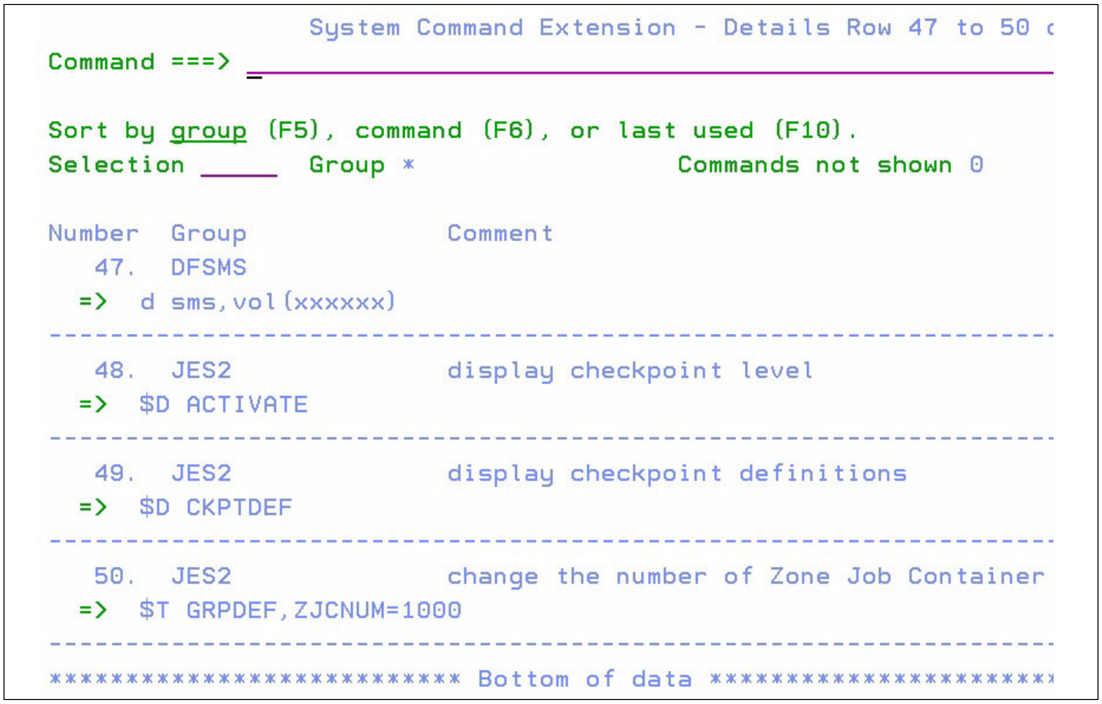
In addition to main panel, the comments are displayed in detailed pop-up panels, which is described next.

Detailed pop-up panel

The main commands panel lists only the last few commands that were issued by the user, even if a larger limit of save commands are allowed or if ISFTABL is allocated and in use by users. The new details pop-up panel allows users to see all saved commands with a group and comments while enabling sort options for ease of use.

To access the Details pop-up panel, press F6 from the system commands extension panel. All of your saved commands are shown and are sorted by group name. Ungrouped commands are displayed first.

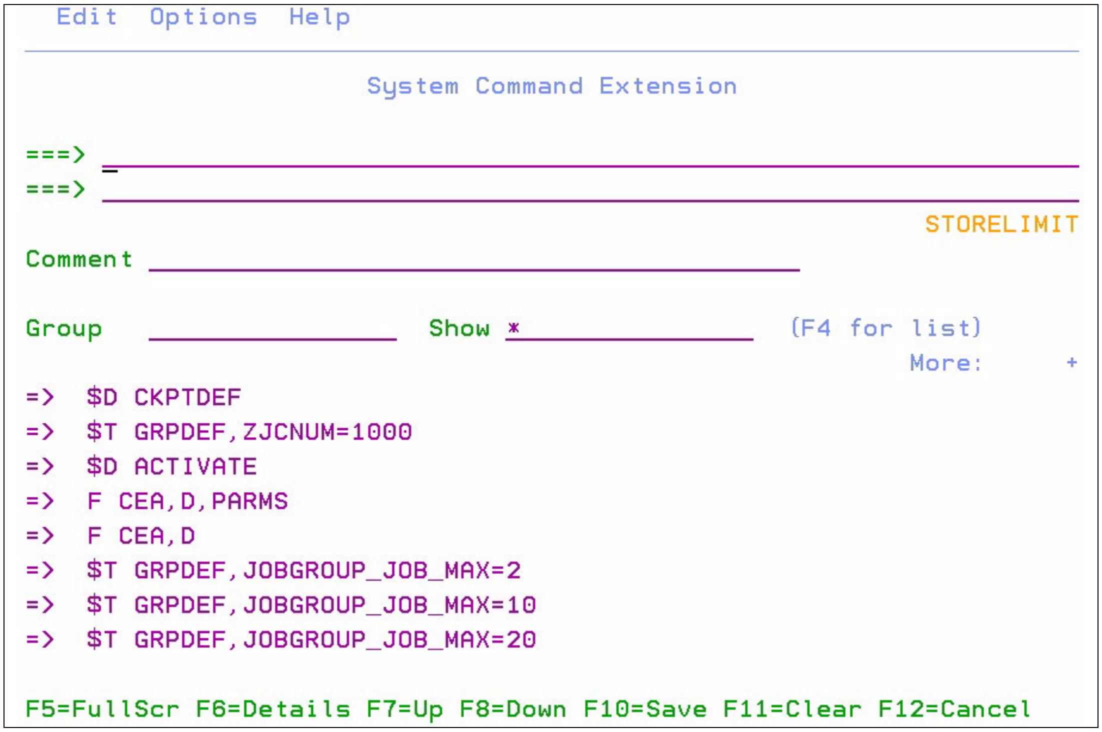
To change the sort option, press F5 to sort by Group, F6 to sort by Command, or F10 to sort by last used. Figure 3-5 shows the Details pop-up panel as sorted by last used commands. The group name information and command comment also is displayed in this panel.



```
                  System Command Extension - Details Row 47 to 50 c
Command ===> _

Sort by group (F5), command (F6), or last used (F10).
Selection _____   Group *                  Commands not shown 0

Number  Group              Comment
   47.  DFSMS
  =>  d sms,vol(xxxxxx)
-------------------------------------------------------------------
   48.  JES2               display checkpoint level
  =>  $D ACTIVATE
-------------------------------------------------------------------
   49.  JES2               display checkpoint definitions
  =>  $D CKPTDEF
-------------------------------------------------------------------
   50.  JES2               change the number of Zone Job Container
  =>  $T GRPDEF,ZJCNUM=1000
-------------------------------------------------------------------
*************************** Bottom of data ***********************
```

Figure 3-5 System Command Extension: Details panel

To select a command for processing, enter the command number in the selection option, and press Ctrl+Enter. You are returned to the System Commands Extension panel, with the required command ready for submission, as shown in Figure 3-6.



```
  Edit  Options  Help
------------------------------------------------------------------------
                      System Command Extension

===> _
===>
                                                            STORELIMIT
Comment

Group                    Show *                      (F4 for list)
                                                        More:      +
=>  $D CKPTDEF
=>  $T GRPDEF,ZJCNUM=1000
=>  $D ACTIVATE
=>  F CEA,D,PARMS
=>  F CEA,D
=>  $T GRPDEF,JOBGROUP_JOB_MAX=2
=>  $T GRPDEF,JOBGROUP_JOB_MAX=10
=>  $T GRPDEF,JOBGROUP_JOB_MAX=20

F5=FullScr F6=Details F7=Up F8=Down F10=Save F11=Clear F12=Cancel
```

Figure 3-6 System Command Extension panel

We encourage you to include comments with your commands to improve command management and usage.

Migration and coexistence

The Details pop-up menu is not available in lower-level releases. In a mixed environment, this option is available on z/OS V2R2 images only.

New action bar options

The system commands extension panel also provides users a menu option to control system commands panel. There are three options available: Edit, Options, and Help.

Edit

The Edit option can be used to clear unused commands from your saved list. When the Clear option is selected, a pop-up panel prompts you for the clear action to take. This panel is the same panel from the F11 option.

From this panel, you can select the following options:

- Recent matching the value for Show

  This option deletes the 20 most recent commands that match the values set on system commands main window.

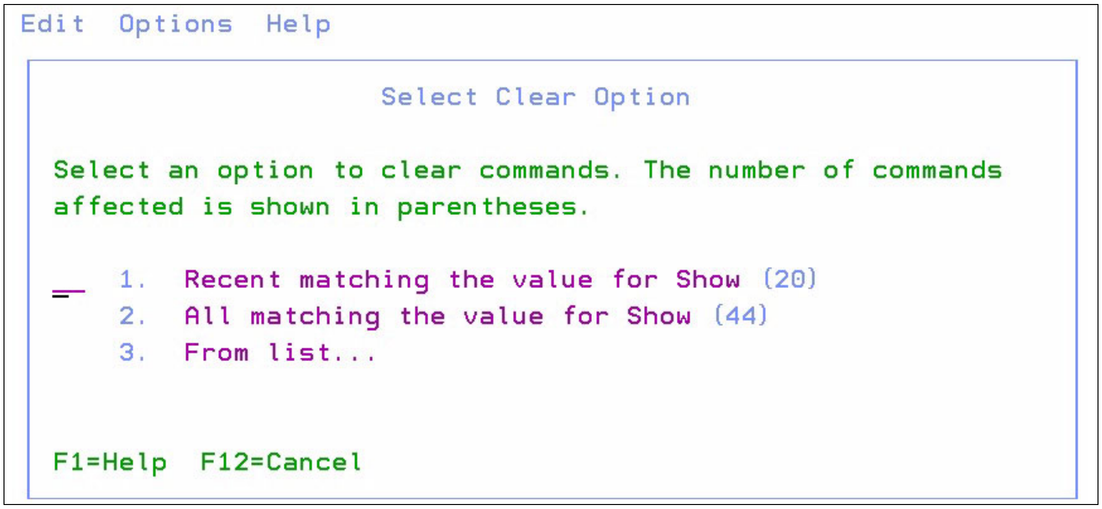
- All matching the value for Show

  This option deletes *all* of the commands that match the values set on system commands main window.

- From list

  Open a list of commands that match the selection criteria. Place the number of the command that you want to delete and press Ctrl+Enter.

Figure 3-7 shows the Select Clear Option panel.



```
Edit  Options  Help

                    Select Clear Option

 Select an option to clear commands. The number of commands
 affected is shown in parentheses.

 __   1.  Recent matching the value for Show (20)
      2.  All matching the value for Show (44)
      3.  From list...


 F1=Help  F12=Cancel
```

Figure 3-7 Select Clear Option panel

Warning: There is no confirmation panel to delete the entries beyond this panel. Be sure that you want to delete the commands before proceeding from the Select Clear Option panel.

Options

From the Options menu, you can control the following system commands:

- Set wait response to ON|OFF

  This option sets the WAIT response to ON or OFF.

- Set Store Commands in ISPF Profile at Exit to ON|OFF

  Sets this ON|OFF to enable or disable saving issued commands.

- Set Store Limit Warning to ON|OFF

  A warning limit notifies you when the limit of saved commands is reached. New commands remove the oldest commands from the list.

Help

Help options provide information about the system commands panel and list system commands manuals on option 2.

3.4 UI enhancements

Starting in z/OS V2R2, many user interfaces enhancements were included. These new options provide you with the means to control your queues and performing commands against multiple entries.

The new features and some examples are described next.

3.4.1 Line shortcut command

Before z/OS V2R2, performing tasks, such as changing job Class of multiple jobs, was time-consuming and required that the user to change the value job-by-job on SDSF panels. Now, such tasks can be performed for multiple jobs from command line, which can significantly reduce the time that is spent to perform repetitive operations.

On the command line, you can specify the line number (or up to three ranges of line numbers) followed by an action to issue against multiple rows. Some examples are shown in Example 3-2.

Example 3-2 Shortcut commands

```
2 D -> issues the Display action against the second row
1-5 P -> issues the Purge action against rows 1 to 5
1-3 6-10 14 C -> issues the Cancel action against rows 1 to 3, 6 to 10, and 14
```

All commands that are issued from the NP column also can be issued from the command line. Multiple actions cannot be performed on the same command. For instance, you cannot select the second output for view and purge it by using the same command.

The value on SET CONFIRM is accepted and only one confirmation is requested for each range. The rows do not have to be on-screen for the command to work and the row number starts on the first entry displayed. For more information about row numbers, see “SET ROWNUM command” on page 47.

You can also type over the value of a column, when allowed. Example 3-3 shows two commands that are used to change the job priority to 13, and job class to B of the second row in the panel.

Example 3-3 Changing job priority and job class

```
2 PRTY=13
2 C=B
```

3.4.2 SET ROWNUM command

The new SET ROWNUM command can be used to enable row numbering on SDSF panels. The row number column appears between the NP column and the fixed field and remains fixed when scrolling left and right.

The column title is ####, with a width of at least four characters (or more where necessary). These numbers are the row numbers that are used online shortcut command.

NP column width

In previous z/OS V2R2 releases, the NP column was a fixed width for each display (often four), with a + action that expanded the column to a larger fixed width (usually six).

In this release, the NP column can be expanded to a specified width by using the `+nn` action, where `nn` is a value 4 - 20. This value is temporary and is reset by using the **`RESET`** command or leaving the display. Use the **`ARRANGE`** command to expand the NP column and save the configuration, as shown in the following example:

```
ARRANGE NP 9
```

The NP width also can be set via the ARRANGE pop-up panel. Each column has its own width. The values are saved in ISPF profile. To open the pop-up panel, click VIEW → **ARRANGE** option, or run the **`ARRANGE ?`** command, as shown in Example 3-4.

Example 3-4 Output of Arrange ? command

```
Arrange           Row 1 to 8 of 55
 Command ===>

 To move a column, select with / (// for a block), then type A
 (after) or B (before). Special function keys:
 F5/17=Refresh list  F11/23=Clear input  F6/18=Default order

      NP width             15

      Column             Width   Description
      StepName              8
      ProcStep              8
      JobID                 8
      Owner                 8
      C                     8
      Pos                   3
      DP                    2
      PGN                   3   Not shown in goal mode
```

3.4.3 Browse locate data set action

Starting in z/OS V2R2, you can browse a specific data set number for a job directly from DA, I, ST, O, H, and JS panels.

Use the `Sn` action to begin the browse action at the data set that is specified by n. It allows you browse the sysout at the beginning of a specified data set where `n` represents the data set number of the job. A negative value `S-n` can be specified to reference an offset from the bottom, as shown in Example 3-5.

Example 3-5 Browse locate data sample

```
S5 –> positions to the fifth data set
S-2 -> positions to the next-to-last data set
```

You can use it with line shortcut commands. If you want to browse the third data set of jobs on rows 1 to 3, you can run the **`1-3 S3`** command.

To browse the jobs that are associated with rows 3 - 5 and positions to the last data set in the job, run the **`3-5 S-1`** command.

3.5 Job displays

The ability to display job information in a concise and meaningful way is important for deciding the actions that are based on a job status, return code, or allocations.

The SDSF enhancements that provide extended display capabilities are described next.

3.5.1 Job step panel

Easy access to job step completion code enables the possibility of taking quick actions upon job failure, or corrective actions while the job is still running.

Starting in z/OS V2R2, SDSF now provides a job step display to consolidate step completion information in a single display, which reduces the time that is spent on searching the required information. This new panel features information about step completion code, abend reason, elapsed time, CPU time, and so on.

To access the job step panel, you can issue a JS next to the job name that you want to check. This action is valid in panels I, ST, O, H, and DA.

This new panel uses new EVENTLOG data set information that is stored by JES2. For more information about EVENTLOG and how to enable or disable it, see 1.3, “EVENTLOG” on page 16.

The amount of information that is available on job step data depends on the EVENTLOG data that is captured. If it is being suppressed, only the data from STEPDATA is retrieved. Otherwise, more data from SMFSTEP is also used. This feature is available for JES2 only.

Table 3-1 lists the new columns of the Job Step display.

Table 3-1 Job Step display

Columns that are displayed when STEPDATA is found				
STEPNAME	ProcStep	Pgm-Name	Step-CC	AbendRsn
StepNum	SysName	Step-Begin	Step-End	
Columns that are displayed when SMF data is available for that job				
Elapsed	CPU-Time	SRB-Time	EXCP-Cnt	Conn
Serv	Workload	Page	Swap	VIO
Swaps	Region	Rgn-Used	MemLimit	Mlim-Used
zIIP-Time	zICP-Time	zIIP-NTime	HiCPU%	HiCPUPgm

Example 3-6 shows the job display panel for a specific job.

Example 3-6 Sample job step completion display

```
Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB STEP DISPLAY - JOB JOBA     (JOB00629) SMF     DATA SET DISPLAYED
COMMAND INPUT ===>                                            SCROLL ===> CSR
NP   #### STEPNAME U-Time    SRB-Time    EXCP-Cnt Conn        Serv  Workload Pa
        1 STEP1    :00:00.00  0:00:00.00       56  0:00:00.00  3525 SYSTEM
        2 STEP2    :00:00.00  0:00:00.00       55  0:00:00.00  3331 SYSTEM
```

From this window, you can interact with job steps by using S, SB, SE (Browse) actions, and X (Print) actions. These actions are processed only to data sets that are associated with the selected step. A ? can be used to view the list of data sets that are associated with the step, and SJ displays the entire job JCL.

3.5.2 Job detail displays

The following new secondary panels were added to provide more information about resources that are used by a job:

- JD (Job Device): Devices and pseudo-devices that are owned or allocated by a job.
- JM (Job Memory): Memory usage by subpool or key.
- JY (Job Delay): Job-related delays.

Job Device panel

By using the Job Device panel, you see information about devices that are used by a job. You display this panel by using the `JD` action character on the DA, I, INIT, NS (for NETSERV address spaces), and ST panels. The ASID, ASIDX, and SYSNAME columns are added to displays in which they were absent.

The rows on JD secondary panel are generated for the following components:

- Active allocations (data sets or devices)
- CF connections
- Connections to remote IP addresses
- Listens on local IP ports

The Type column features one of the following values:

- DD (DD name)

  Columns include:

  - Seq
  - DataSetName
  - Volser
  - Unit
  - Lrecl
  - RecFm
  - Blksize
  - EXCPCt

- CF (CF name)

  Columns include:

  - StrName
  - Volser
  - Policy
  - Status

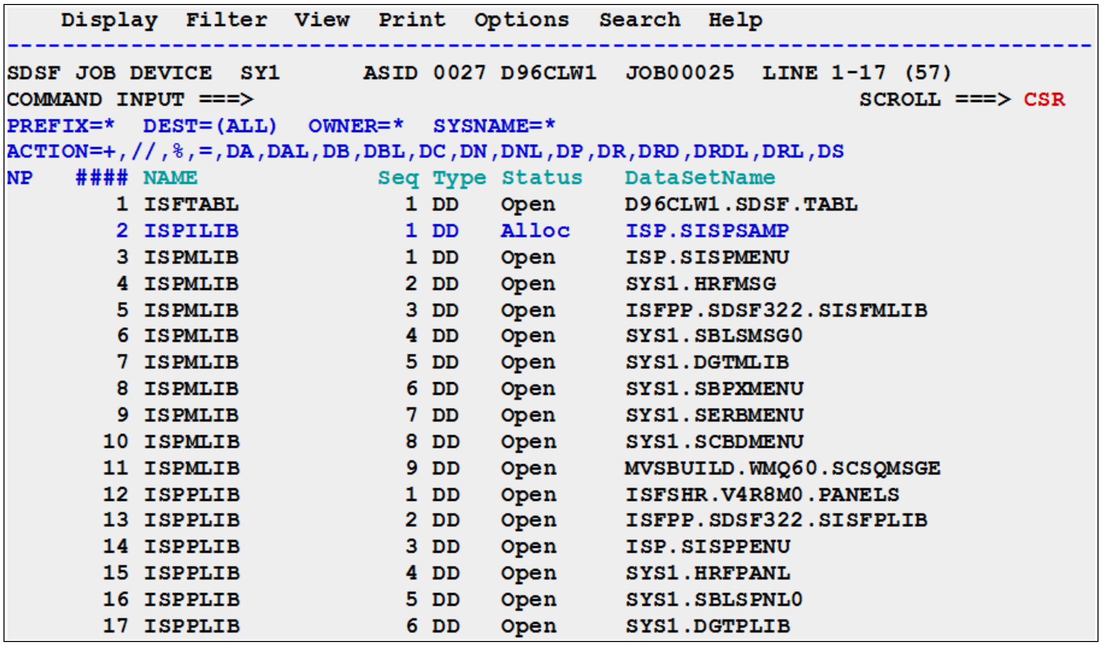
- IP (TCP/IP Server name)

  Columns include:

  - IPAddr
  - Port
  - Status
  - BytesIn
  - BytesOut
  - Start-Time
  - Last-Time
  - Stack
  - Resource-ID
  - ApplData

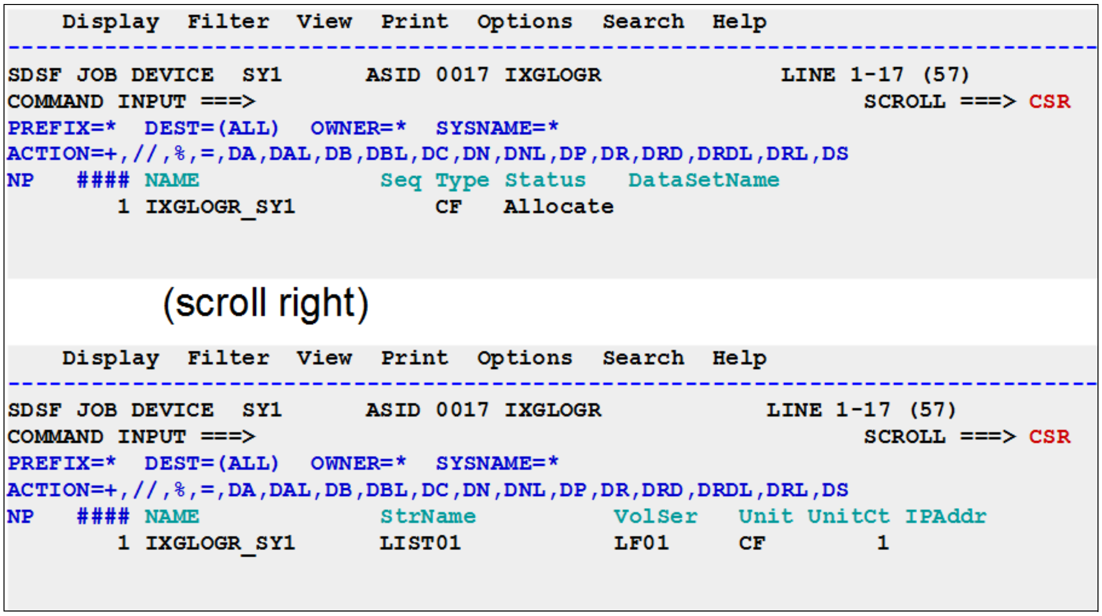
You can use the information that is provided by the job device panel to ensure that the allocations are correct, display CF structure, XCF policy, IP connection information, and so on.

Job Device Detail display examples are shown in Figure 3-8, Figure 3-9, and Figure 3-10 on page 52.



```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB DEVICE  SY1      ASID 0027 D96CLW1  JOB00025  LINE 1-17 (57)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=,DA,DAL,DB,DBL,DC,DN,DNL,DP,DR,DRD,DRDL,DRL,DS
NP    #### NAME             Seq Type Status    DataSetName
         1 ISFTABL            1 DD   Open      D96CLW1.SDSF.TABL
         2 ISPILIB            1 DD   Alloc     ISP.SISPSAMP
         3 ISPMLIB            1 DD   Open      ISP.SISPMENU
         4 ISPMLIB            2 DD   Open      SYS1.HRFMSG
         5 ISPMLIB            3 DD   Open      ISFPP.SDSF322.SISFMLIB
         6 ISPMLIB            4 DD   Open      SYS1.SBLSMSG0
         7 ISPMLIB            5 DD   Open      SYS1.DGTMLIB
         8 ISPMLIB            6 DD   Open      SYS1.SBPXMENU
         9 ISPMLIB            7 DD   Open      SYS1.SERBMENU
        10 ISPMLIB            8 DD   Open      SYS1.SCBDMENU
        11 ISPMLIB            9 DD   Open      MVSBUILD.WMQ60.SCSQMSGE
        12 ISPPLIB            1 DD   Open      ISFSHR.V4R8M0.PANELS
        13 ISPPLIB            2 DD   Open      ISFPP.SDSF322.SISFPLIB
        14 ISPPLIB            3 DD   Open      ISP.SISPPENU
        15 ISPPLIB            4 DD   Open      SYS1.HRFPANL
        16 ISPPLIB            5 DD   Open      SYS1.SBLSPNL0
        17 ISPPLIB            6 DD   Open      SYS1.DGTPLIB
```

Figure 3-8 Job Device Detail display (DD rows)

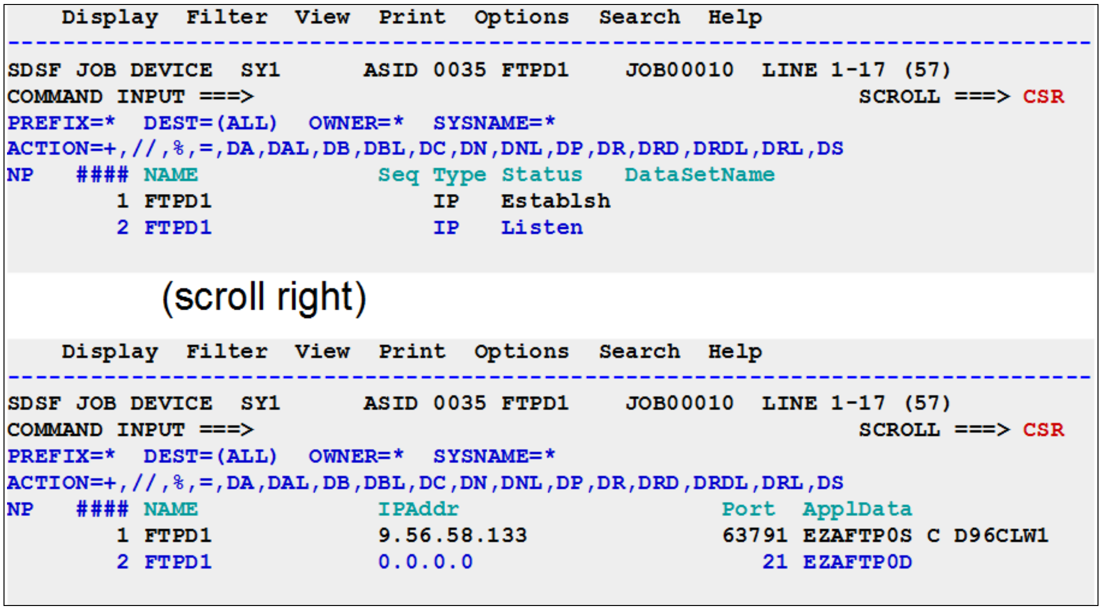


```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB DEVICE  SY1      ASID 0017 IXGLOGR            LINE 1-17 (57)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=,DA,DAL,DB,DBL,DC,DN,DNL,DP,DR,DRD,DRDL,DRL,DS
NP    #### NAME             Seq Type Status    DataSetName
         1 IXGLOGR_SY1          CF   Allocate
```

(scroll right)

```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB DEVICE  SY1      ASID 0017 IXGLOGR           LINE 1-17 (57)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=,DA,DAL,DB,DBL,DC,DN,DNL,DP,DR,DRD,DRDL,DRL,DS
NP    #### NAME             StrName          VolSer   Unit UnitCt IPAddr
         1 IXGLOGR_SY1      LIST01           LF01     CF        1
```

Figure 3-9 Job Device Detail display (CF rows)



```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB DEVICE  SY1      ASID 0035 FTPD1    JOB00010  LINE 1-17 (57)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=,DA,DAL,DB,DBL,DC,DN,DNL,DP,DR,DRD,DRDL,DRL,DS
NP    #### NAME                Seq Type Status    DataSetName
        1 FTPD1                    IP   Establsh
        2 FTPD1                    IP   Listen
```

(scroll right)

```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB DEVICE  SY1      ASID 0035 FTPD1    JOB00010  LINE 1-17 (57)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=,DA,DAL,DB,DBL,DC,DN,DNL,DP,DR,DRD,DRDL,DRL,DS
NP    #### NAME                IPAddr                   Port  ApplData
        1 FTPD1                9.56.58.133              63791 EZAFTP0S C D96CLW1
        2 FTPD1                0.0.0.0                     21 EZAFTP0D
```

Figure 3-10   Job Device Detail display (IP rows)

The following actions are available in Job Device panel. They actions are display actions and vary by row type:

- DD

  No actions defined.

- CF:
  - Display actions are different forms of **D XCF** command
  - **DC** (Display CF) – Displays the CF by using **D XCF** command
  - **DS** (Display Structure) – Displays the structure by using **D XCF**
  - **DP** (Display Policy) – Displays the policy by using **D XCF**
- IP:
  - Display actions are different forms of **D TCPIP** command
  - DA (Display All): D TCPIP,stack,N,ALL,IPP=
  - DN (Display Conn): D TCPIP,stack,N,CO,APPLDATA,IPP=
  - DB (Display Byte info): D TCPIP,stack,N,BYTE,IDLETIME,IPA=
  - DR (Display Route): D TCPIP,stack,N,ROUTE,IPA=

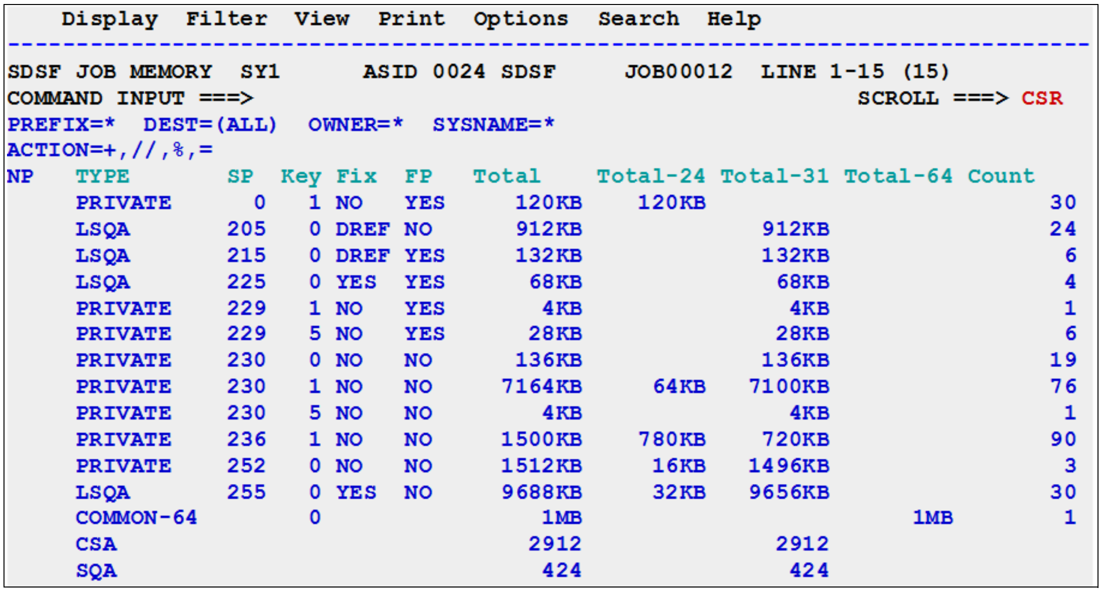
Job Memory panel

This panel is accessible via the JM action from panels where individual rows represent (or can represent) an active address space: DA, I, INIT, NS (for NETSERV address spaces), and ST panels.

Rows on JM secondary panel are generated for the following components:

- Each subpool or key combination for which memory is allocated
- 64-bit private storage (by key)
- 64-bit common storage that is owned by address space (by key)
- CSA and SQA that is owned by address space (if CSA tracking is active)

An example of the Job Memory panel is shown in Figure 3-11 on page 53.



```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB MEMORY  SY1      ASID 0024 SDSF     JOB00012  LINE 1-15 (15)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=
NP    TYPE        SP  Key Fix  FP    Total    Total-24 Total-31 Total-64 Count
      PRIVATE       0   1 NO   YES    120KB    120KB                        30
      LSQA        205   0 DREF NO     912KB             912KB               24
      LSQA        215   0 DREF YES    132KB             132KB                6
      LSQA        225   0 YES  YES     68KB              68KB                4
      PRIVATE     229   1 NO   YES      4KB               4KB                1
      PRIVATE     229   5 NO   YES     28KB              28KB                6
      PRIVATE     230   0 NO   NO     136KB             136KB               19
      PRIVATE     230   1 NO   NO    7164KB     64KB   7100KB               76
      PRIVATE     230   5 NO   NO       4KB               4KB                1
      PRIVATE     236   1 NO   NO    1500KB    780KB    720KB               90
      PRIVATE     252   0 NO   NO    1512KB     16KB   1496KB                3
      LSQA        255   0 YES  NO    9688KB     32KB   9656KB               30
      COMMON-64         0               1MB                         1MB      1
      CSA                              2912              2912
      SQA                               424               424
```

Figure 3-11 Job Memory panel

You can use this panel to help identify the best memory allocation for certain jobs and tasks or to indicate programs in error.

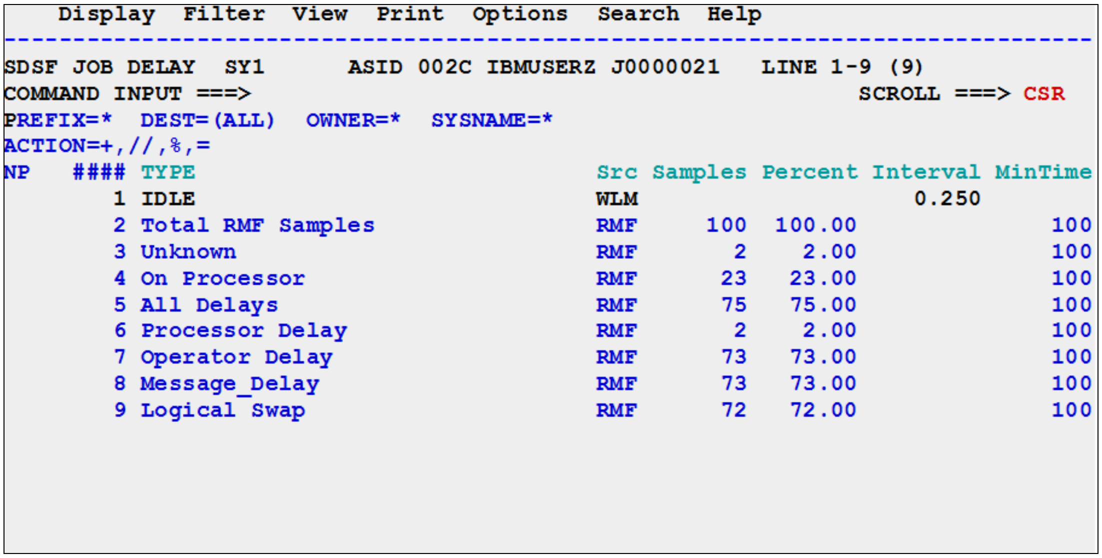
Job Delay panel

This panel is accessible via the `JY` action from DA panel.

Rows on JY secondary panel are generated for the following components:

- Current delay information reported by WLM
- All delays for latest interval as reported via RMF

Because this panel uses RMF information, it is not available from non-RMF version of DA. The Job Delay panel is shown in Figure 3-12.



```
    Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF JOB DELAY  SY1      ASID 002C IBMUSERZ J0000021   LINE 1-9 (9)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=*
ACTION=+,//,%,=
NP    #### TYPE                         Src Samples Percent Interval MinTime
         1 IDLE                         WLM                     0.250
         2 Total RMF Samples            RMF     100  100.00               100
         3 Unknown                      RMF       2    2.00               100
         4 On Processor                 RMF      23   23.00               100
         5 All Delays                   RMF      75   75.00               100
         6 Processor Delay              RMF       2    2.00               100
         7 Operator Delay               RMF      73   73.00               100
         8 Message_Delay                RMF      73   73.00               100
         9 Logical Swap                 RMF      72   72.00               100
```

Figure 3-12 Job Delay panel

Installation and usage

The JD, JM, and JY actions are protected by the following SAF profiles:

- In JESSPOOL class

  For jobs that are known to JES, actions are protected by sysname.userid.jobname.jobid profile. The READ access is required.

- In SDSF class

  Each panel has its own profile and READ access is required:

  – JD: ISFDISP.DEVICES.userid.jobname
  – JM: ISFDISP.STORAGE.userid.jobname
  – JY: ISFDISP.DELAY.userid.jobname

Access is allowed if either profile allows it.

3.5.3 Snapshot command

A new **SNAPSHOT** command was added to capture the contents of a tabular display into a browse or edit session. All columns are captured, including the columns that are not displayed on-screen.

You can use the **PRINT** command (from SDSF Browse) or the **COPY** (from ISPF Edit) command to move the captured data to a more permanent location. The following data is captured:

- Rows and column data that is in the same order as on the display
- Column widths are maximized to prevent data loss and numeric scaling

The following syntax is used:

```
SNAP [S|SB|SE]
```

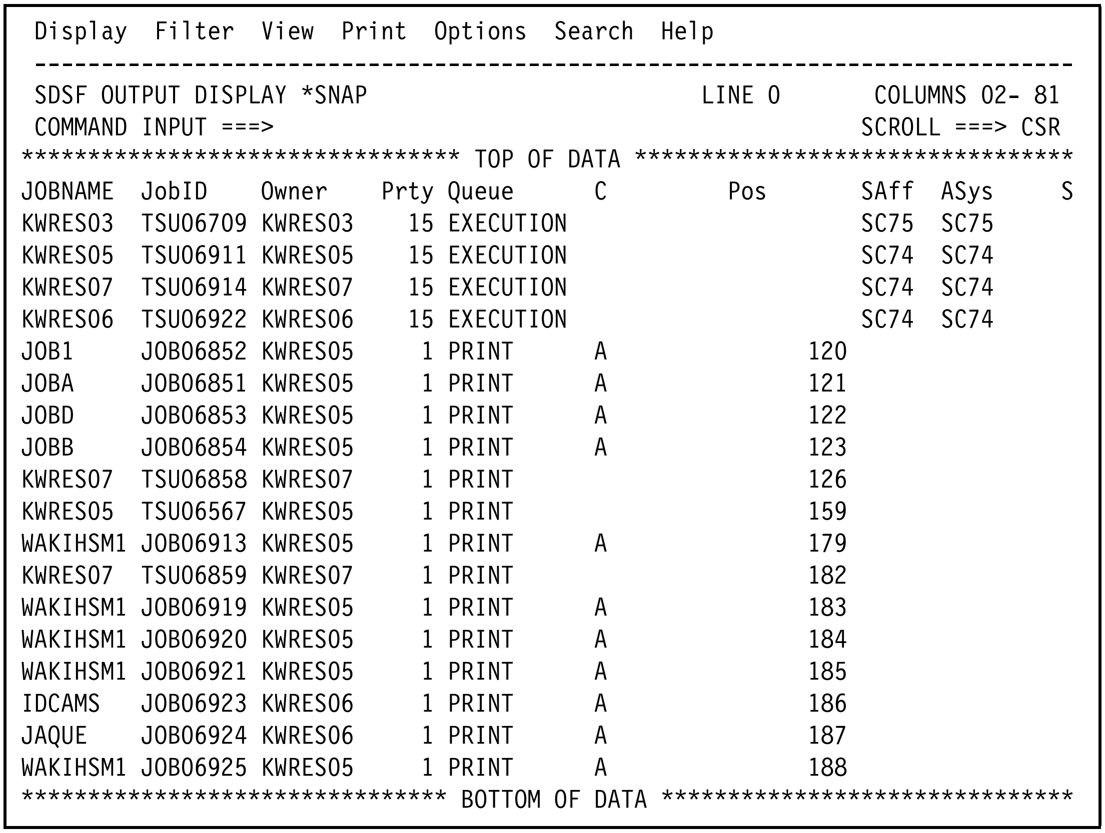
Use the following SET SNAP command to specify the default:

```
SET SNAP [S|SB|SE|?]
```

Where:

- S: Uses SDSF Browse to view data
- SB: Uses ISPF Browse to view data (requires ISPF)
- SE: Uses ISPF Edit to view data (requires ISPF)
- ?: Opens pop-up panel to enter choice

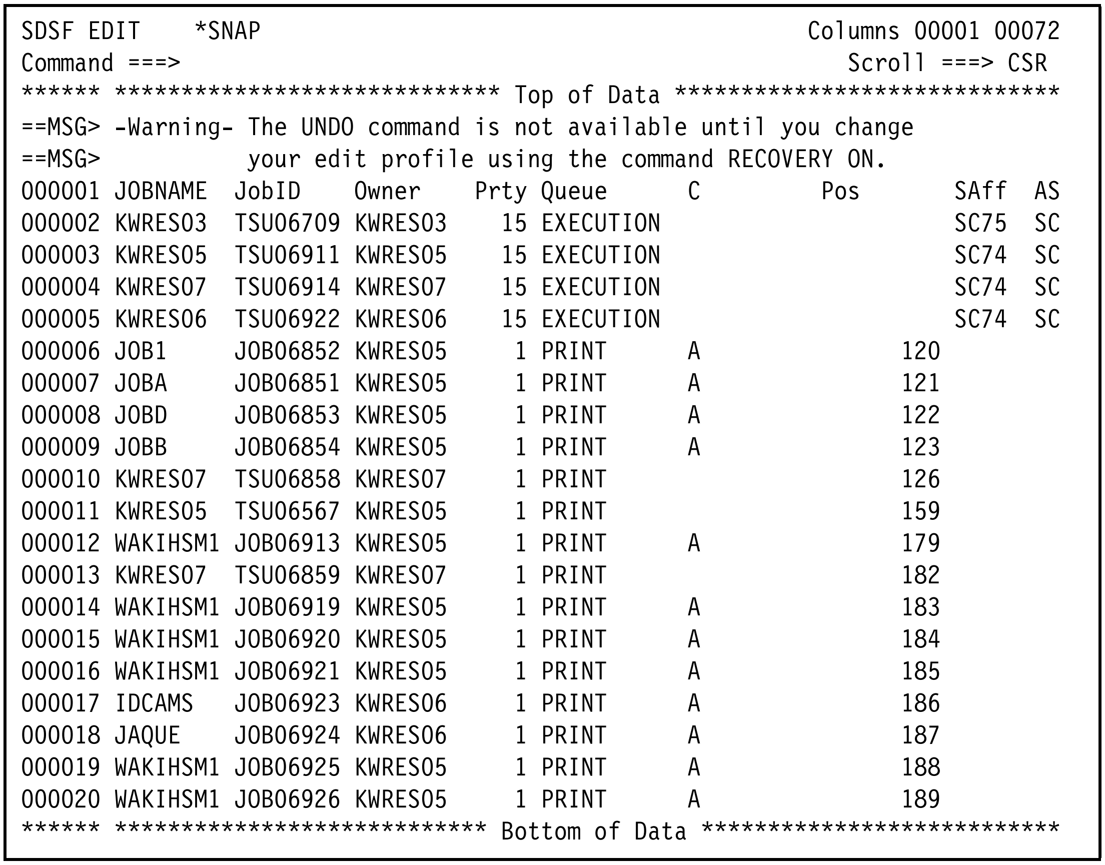
A SNAP command output example from SDSF browse is shown in Figure 3-13 on page 55.



```
 Display  Filter  View  Print  Options  Search  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY *SNAP                         LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> CSR
********************************* TOP OF DATA *********************************
JOBNAME  JobID    Owner    Prty Queue      C         Pos       SAff  ASys     S
KWRES03  TSU06709 KWRES03    15 EXECUTION                      SC75  SC75
KWRES05  TSU06911 KWRES05    15 EXECUTION                      SC74  SC74
KWRES07  TSU06914 KWRES07    15 EXECUTION                      SC74  SC74
KWRES06  TSU06922 KWRES06    15 EXECUTION                      SC74  SC74
JOB1     JOB06852 KWRES05     1 PRINT      A               120
JOBA     JOB06851 KWRES05     1 PRINT      A               121
JOBD     JOB06853 KWRES05     1 PRINT      A               122
JOBB     JOB06854 KWRES05     1 PRINT      A               123
KWRES07  TSU06858 KWRES07     1 PRINT                      126
KWRES05  TSU06567 KWRES05     1 PRINT                      159
WAKIHSM1 JOB06913 KWRES05     1 PRINT      A               179
KWRES07  TSU06859 KWRES07     1 PRINT                      182
WAKIHSM1 JOB06919 KWRES05     1 PRINT      A               183
WAKIHSM1 JOB06920 KWRES05     1 PRINT      A               184
WAKIHSM1 JOB06921 KWRES05     1 PRINT      A               185
IDCAMS   JOB06923 KWRES06     1 PRINT      A               186
JAQUE    JOB06924 KWRES06     1 PRINT      A               187
WAKIHSM1 JOB06925 KWRES05     1 PRINT      A               188
******************************** BOTTOM OF DATA *******************************
```

Figure 3-13 SNAP output SDSF browse display

Figure 3-14 shows a SNAP command output example from an ISPF edit.



```
SDSF EDIT    *SNAP                                        Columns 00001 00072
Command ===>                                                 Scroll ===> CSR
****** ***************************** Top of Data *****************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 JOBNAME  JobID    Owner    Prty Queue      C         Pos       SAff  AS
000002 KWRES03  TSU06709 KWRES03    15 EXECUTION                      SC75  SC
000003 KWRES05  TSU06911 KWRES05    15 EXECUTION                      SC74  SC
000004 KWRES07  TSU06914 KWRES07    15 EXECUTION                      SC74  SC
000005 KWRES06  TSU06922 KWRES06    15 EXECUTION                      SC74  SC
000006 JOB1     JOB06852 KWRES05     1 PRINT      A               120
000007 JOBA     JOB06851 KWRES05     1 PRINT      A               121
000008 JOBD     JOB06853 KWRES05     1 PRINT      A               122
000009 JOBB     JOB06854 KWRES05     1 PRINT      A               123
000010 KWRES07  TSU06858 KWRES07     1 PRINT                      126
000011 KWRES05  TSU06567 KWRES05     1 PRINT                      159
000012 WAKIHSM1 JOB06913 KWRES05     1 PRINT      A               179
000013 KWRES07  TSU06859 KWRES07     1 PRINT                      182
000014 WAKIHSM1 JOB06919 KWRES05     1 PRINT      A               183
000015 WAKIHSM1 JOB06920 KWRES05     1 PRINT      A               184
000016 WAKIHSM1 JOB06921 KWRES05     1 PRINT      A               185
000017 IDCAMS   JOB06923 KWRES06     1 PRINT      A               186
000018 JAQUE    JOB06924 KWRES06     1 PRINT      A               187
000019 WAKIHSM1 JOB06925 KWRES05     1 PRINT      A               188
000020 WAKIHSM1 JOB06926 KWRES05     1 PRINT      A               189
****** **************************** Bottom of Data ***************************
```

Figure 3-14 SNAP output ISPF edit display

3.6 Batch parallelism

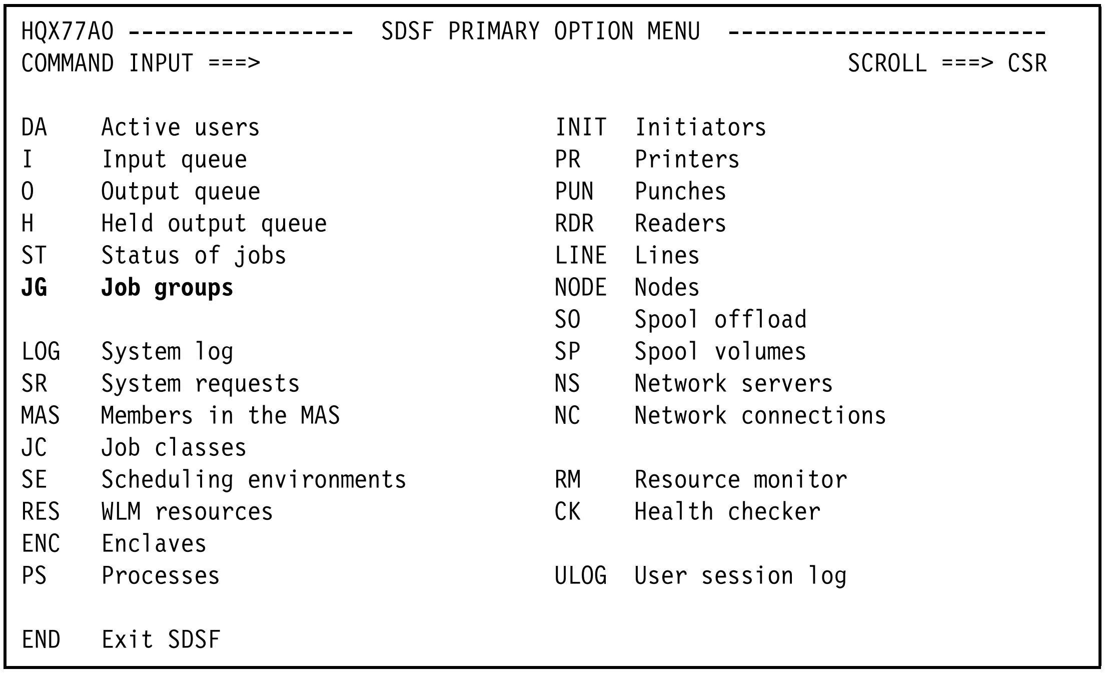
In z/OS V2R2, the JES2 adds support for dependent job control and job groups, which is called *batch parallelism*. Two new panels were added on SDSF for users to manage the new functions.

For more information about how to implement JOBGROUPs on your system, see 1.2, “JOBGROUP” on page 2.

3.6.1 Job Group panel

The job group panel allows you to view and control your JOBGROUPs. This panel contains information about JOBGROUP name, ID, owner, status, system affinity, scheduling environment, and so on. This panel is the only panel where it is possible to see JOBGROUPs via SDSF. The other method to display this information is by issuing JES2 commands, as described in Chapter 1, “JES2” on page 1.

You can call the primary JG panel on the SDSF main panel, as shown in Figure 3-15 on page 57.



```
HQX77A0 -----------------  SDSF PRIMARY OPTION MENU  ------------------------
COMMAND INPUT ===>                                              SCROLL ===> CSR

DA    Active users                      INIT  Initiators
I     Input queue                       PR    Printers
O     Output queue                      PUN   Punches
H     Held output queue                 RDR   Readers
ST    Status of jobs                    LINE  Lines
JG    Job groups                        NODE  Nodes
                                        SO    Spool offload
LOG   System log                        SP    Spool volumes
SR    System requests                   NS    Network servers
MAS   Members in the MAS                NC    Network connections
JC    Job classes
SE    Scheduling environments           RM    Resource monitor
RES   WLM resources                     CK    Health checker
ENC   Enclaves
PS    Processes                         ULOG  User session log

END   Exit SDSF
```

Figure 3-15 Job groups option

A SAF profile was created to protect the job group panel. It is protected by `ISFCMD.DSP.JGROUP.jesx` profile in `SDSF` class. READ access is required. The profile is included by default when ISFPARMS specifies **`GROUP AUTH(ALL)`**, **`GROUP AUTH(ALLOPER)`** or **`GROUP AUTH(ALLUSER)`** or includes `JG` in `AUTH` list.

From the job group panel, you can issue JES action commands against JOBGROUPs. The commands A, C, CP, D, H, P, S, X, and ? are available for use. New commands were introduced to provide JOBGROUP specific actions. The following new actions are available:

- DE

  Display the jobs in the JOBGROUP that encountered an error.

- DJ

  Display jobs that are associated with a JOBGROUP.

- DL

  Display JOBGROUP information. This command includes information about creation time, spool usage, and so on.

- DN

  Display network. Shows information about jobs in error, jobs that belong to JOBGROUP, concurrent jobs, and job dependency. This action is an alias for $D GROUP(xxxx),SUMMARY command.

- DP

  Display dependency, including job predecessors and concurrent jobs.

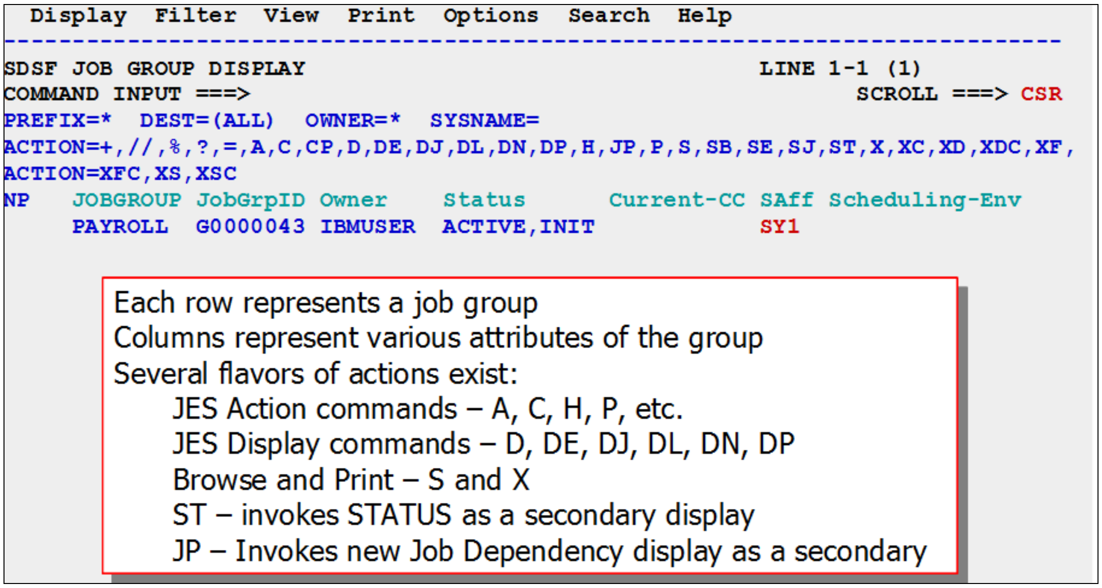
- JP

  Job dependency. Provides a table that is based output for dependent and concurrent jobs.

- ST

  Access to ST panel. This action displays the jobs from the JOBGROUP, including the jobs that are not yet submitted. To see the job names, ensure that your OWNER and PREFIX settings meet job attributes.

The job group display is shown in Figure 3-16.

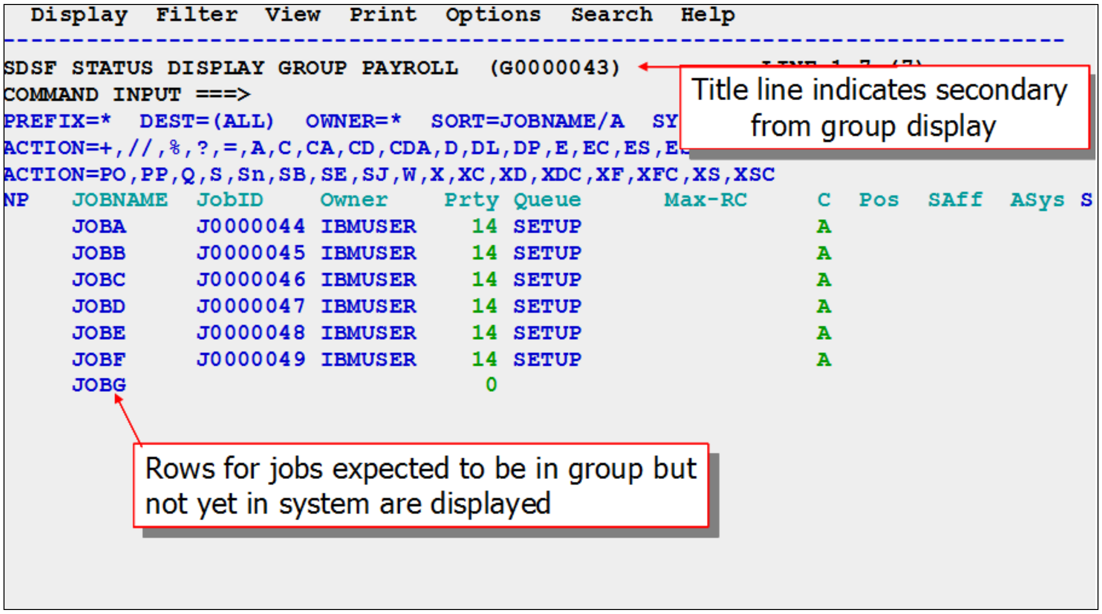


```
  Display  Filter  View  Print  Options  Search  Help
 -----------------------------------------------------------------------------
SDSF JOB GROUP DISPLAY                                      LINE 1-1  (1)
COMMAND INPUT ===>                                                 SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=
ACTION=+,//,%,?,=,A,C,CP,D,DE,DJ,DL,DN,DP,H,JP,P,S,SB,SE,SJ,ST,X,XC,XD,XDC,XF,
ACTION=XFC,XS,XSC
NP    JOBGROUP JobGrpID Owner    Status      Current-CC SAff Scheduling-Env
      PAYROLL  G0000043 IBMUSER  ACTIVE,INIT            SY1
```

Each row represents a job group
Columns represent various attributes of the group
Several flavors of actions exist:
JES Action commands – A, C, H, P, etc.
JES Display commands – D, DE, DJ, DL, DN, DP
Browse and Print – S and X
ST – invokes STATUS as a secondary display
JP – Invokes new Job Dependency display as a secondary

Figure 3-16 Job group display

The job status display from the JOBGROUP display is shown in Figure 3-17.



```
  Display  Filter  View  Print  Options  Search  Help
 -----------------------------------------------------------------------------
SDSF STATUS DISPLAY GROUP PAYROLL  (G0000043)
COMMAND INPUT ===>
PREFIX=*  DEST=(ALL)  OWNER=*  SORT=JOBNAME/A  SY
ACTION=+,//,%,?,=,A,C,CA,CD,CDA,D,DL,DP,E,EC,ES,E
ACTION=PO,PP,Q,S,Sn,SB,SE,SJ,W,X,XC,XD,XDC,XF,XFC,XS,XSC
NP    JOBNAME  JobID    Owner    Prty Queue      Max-RC     C  Pos  SAff  ASys S
      JOBA     J0000044 IBMUSER   14  SETUP                 A
      JOBB     J0000045 IBMUSER   14  SETUP                 A
      JOBC     J0000046 IBMUSER   14  SETUP                 A
      JOBD     J0000047 IBMUSER   14  SETUP                 A
      JOBE     J0000048 IBMUSER   14  SETUP                 A
      JOBF     J0000049 IBMUSER   14  SETUP                 A
      JOBG                         0
```

Title line indicates secondary from group display

Rows for jobs expected to be in group but not yet in system are displayed

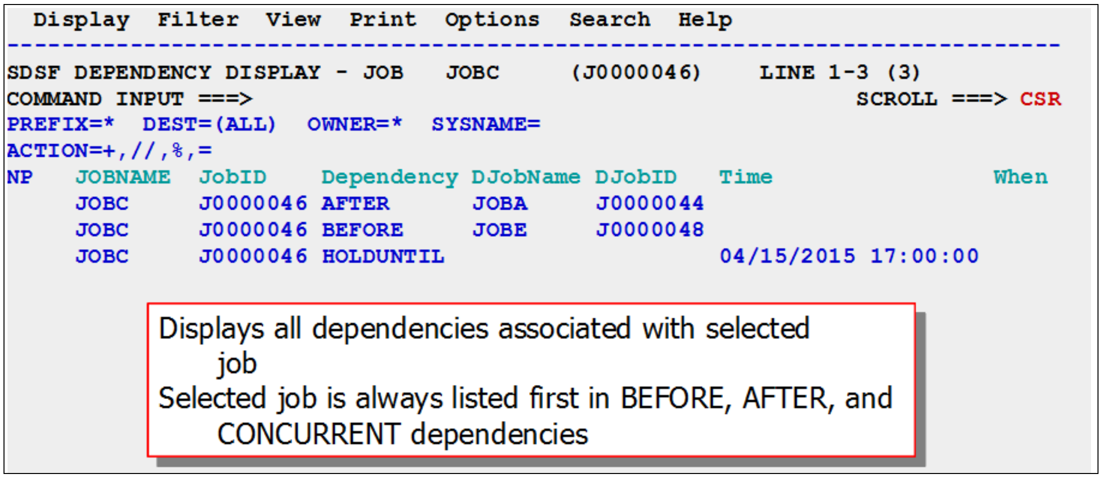
Figure 3-17 Status display from JG display

3.6.2 Job dependency panel

The job dependency panel displays the following information:

- For a selected JOBGROUP, all of the dependency within the group.
- For a selected job:
  - Jobs that have dependencies
  - Jobs that have dependencies on it

A job dependency display is shown in Figure 3-18.

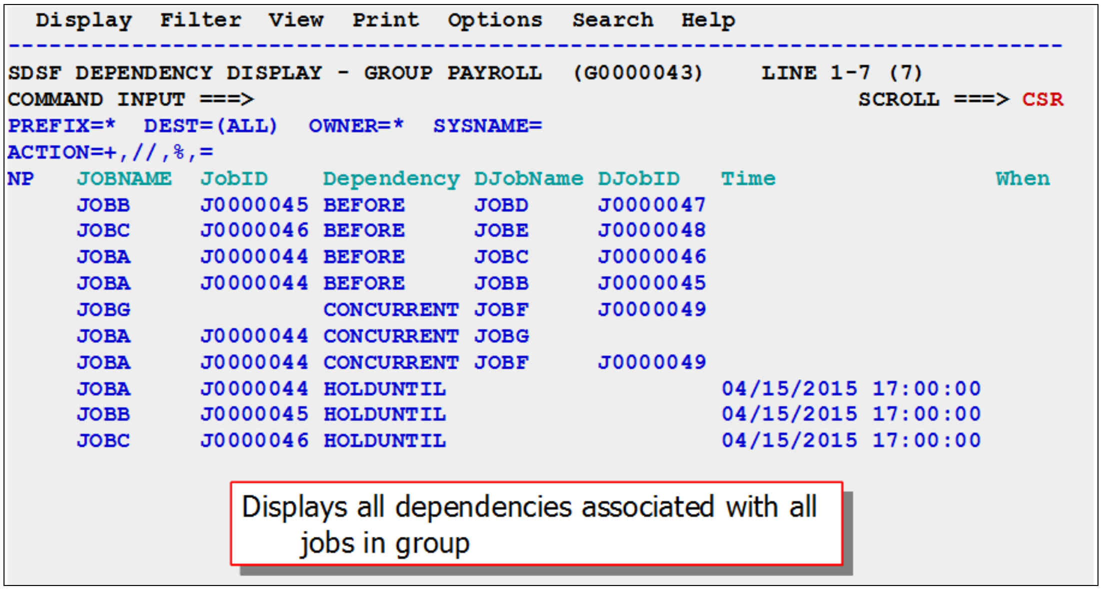


```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF DEPENDENCY DISPLAY - JOB   JOBC     (J0000046)    LINE 1-3  (3)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=
ACTION=+,//,%,=
NP    JOBNAME   JobID     Dependency DJobName DJobID    Time                 When
      JOBC      J0000046 AFTER      JOBA     J0000044
      JOBC      J0000046 BEFORE     JOBE     J0000048
      JOBC      J0000046 HOLDUNTIL                     04/15/2015 17:00:00
```

Displays all dependencies associated with selected job
Selected job is always listed first in BEFORE, AFTER, and CONCURRENT dependencies

Figure 3-18 Job dependency display

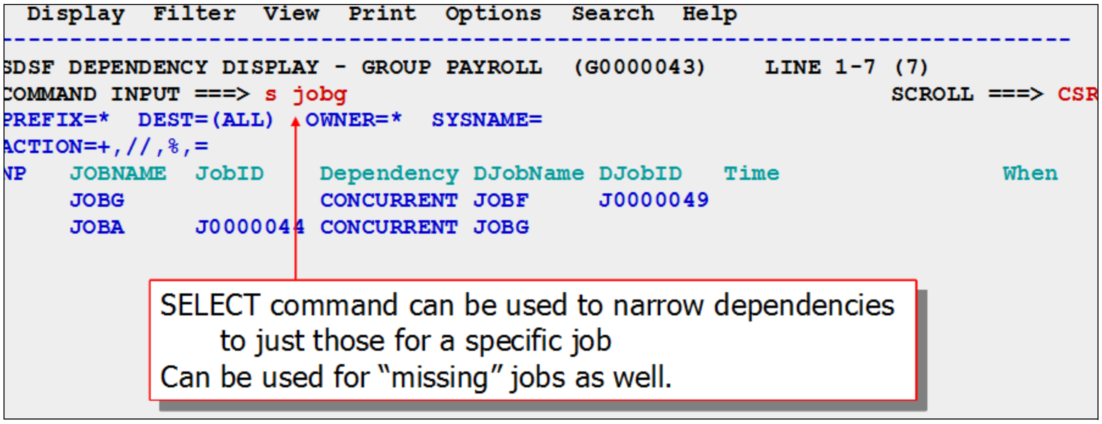
The Job Dependency panel can be accessed with the `JP` action character from the JG panel. The Job Dependency display is shown in Figure 3-19 and Figure 3-20 on page 60.



```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF DEPENDENCY DISPLAY - GROUP PAYROLL  (G0000043)    LINE 1-7  (7)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=
ACTION=+,//,%,=
NP    JOBNAME   JobID     Dependency DJobName DJobID    Time                 When
      JOBB      J0000045 BEFORE     JOBD     J0000047
      JOBC      J0000046 BEFORE     JOBE     J0000048
      JOBA      J0000044 BEFORE     JOBC     J0000046
      JOBA      J0000044 BEFORE     JOBB     J0000045
      JOBG               CONCURRENT JOBF     J0000049
      JOBA      J0000044 CONCURRENT JOBG
      JOBA      J0000044 CONCURRENT JOBF     J0000049
      JOBA      J0000044 HOLDUNTIL                     04/15/2015 17:00:00
      JOBB      J0000045 HOLDUNTIL                     04/15/2015 17:00:00
      JOBC      J0000046 HOLDUNTIL                     04/15/2015 17:00:00
```

Displays all dependencies associated with all jobs in group

Figure 3-19 Dependency display from JG



```
 Display  Filter  View  Print  Options  Search  Help
-----------------------------------------------------------------------------
SDSF DEPENDENCY DISPLAY - GROUP PAYROLL  (G0000043)    LINE 1-7 (7)
COMMAND INPUT ===> s jobg                                    SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=
ACTION=+,//,%,=
NP    JOBNAME  JobID     Dependency DJobName DJobID   Time              When
      JOBG               CONCURRENT JOBF     J0000049
      JOBA     J0000044 CONCURRENT JOBG
```

SELECT command can be used to narrow dependencies to just those for a specific job
Can be used for "missing" jobs as well.

Figure 3-20 Select command

3.7 REXX enhancements

The following new enhancements are available in IBM System Rexx:

- A new **RGEN** command is available to create custom samples that are based on current context. It can be tailored by using current settings, such as PREFIX, OWNER, and FILTER values.
- There is a new % prefix character that allows Rexx execs to be run as an action against a row.

The RGEN commands provide better examples to get started writing Rexx execs. Common tasks can be more easily performed by creating custom Rexx actions.

3.7.1 RGEN command

The **RGEN** command generates a REXX exec for the current panel and displays it with ISPF Edit. The exec includes the statements that you need to add the SDSF host command environment and to access the current panel, and special variables for tasks, such as filtering.

The exec also might include suggested logic for more functions. The generated exec is displayed by using ISPF Edit.

You can use RGEN for the following tasks:

- Display the tabular panel (DA, ST, PR, JDS, and so on) or log panel (SYSLOG, OPERLOG, and ULOG) with which you want to work.
- Issue the **RGEN** command from the command line. SDSF generates the appropriate exec and displays it by using ISPF Edit. The display includes special temporary lines that are visible in ISPF Edit but are not included in the exec. To remove those lines, use the **RESET** command.
- Copy the exec to a data set by using the **CREATE** command. Copying the exec before you begin making any updates ensures that none of your changes are lost.
- Modify the exec to suit your needs.

3.8 Miscellaneous changes

The following enhancements are available for z/OS V2R2:

- JES3 OUTDISP support:
  - The OUTDISP columns on O and H are enabled for JES3, but you cannot type over them.
  - A new OUTDISP column on JDS display was added and can be typed over.
- JES2 Dynamic checkpoint tuning:
  - The HOLD and DORMANCY columns on MAS panel cannot be typed over when dynamic checkpoint tuning is used (MASDEF CYCLEMGT=AUTO). However, the columns are still displayed and show values that are used internally.

    For more information about JES2 Dynamic checkpoint tuning, see 1.7.4, “JES2 checkpoint tuning” on page 27.
  - There is a MAS display title line indicator when it is in effect.
- The user ID is included in the enclave display.

Related publications

The publications that are listed in this section are considered particularly suitable for a more detailed discussion of the topics that are covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide more information about the z/OS V2R2 updates. Some of the publications that are referenced in this list might be available in softcopy only:

- *z/OS V2R2: JES2, JES3, and SDSF,* SG24-8287
- *z/OS V2R2: Security,* SG24-8288
- *z/OS V2R2: Storage Management and Utilities,* SG24-8289
- *z/OS V2R2: Availability Management,* SG24-8290
- *z/OS V2R2: Performance,* SG24-8292
- *z/OS V2R2: Operations,* SG24-8305
- *z/OS V2R2: Diagnostics,* SG24-8306
- *z/OS V2R2: Sysplex,* SG24-8307
- *z/OS V2R2: UNIX System Services* SG24-8310
- *z/OS V2R2: User Interfaces,* SG24-8311
- *z/OS V2R2: ServerPac*, SG24-8500

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

The following publications are also relevant as further information sources:

- *z/OS JES2 Commands,* SA32-0990
- *z/OS JES2 Initialization and Tuning Guide,* SA32-0991
- *z/OS JES2 Initialization and Tuning Reference* SA32-0992
- z/OS JES3 Commands, SA32-1008
- z/OS JES3 Customization, SA32 1006
- *z/OS SDSF Operation and Customization,* SA23-2274

Help from IBM

IBM Support and downloads:

ibm.com/support

IBM Global Services:

ibm.com/services

(0.1”spine)
0.1”<->0.169”
53<->89 pages

SG24-8287-00

ISBN 0738441260

Printed in U.S.A.