New SDSF functions in z/OS 3.2

Rob Scott

rscott@rs.com

Agenda

- Summary Of Recent SDSF Enhancements
- pysdsf
- SDSF Profile Data In zFS
- RACF Log Display
- Sysname Lists
- New SDSF Panels
- Useability Enhancements
- Miscellaneous Enhancements

Recent SDSF Enhancements

- z/OS 2.5
  - SAF-only security
  - 16 new panels including memory browse, detailed common and private storage information
- z/OS 3.1
  - New concept of SDSF features
  - Module fetch monitor and event log
  - Over 30 new panels including RACF classes and profiles
- z/OS 3.2
  - pysdsf
  - z/OS file system for profile data
  - 18 new panels

pysdsf

pysdsf – Python Interface to SDSF

- Easy to use python interface to SDSF
- Initial support for the following primary panels
  - AS, DA, H, I, O, ST
  - LOG (both SYSLOG and OPERLOG)
- Initial support for the following secondary panels
  - JDS (aka '?'), JS
- SYSOUT browse supported
- Operator commands also supported
  - Both generated by actions and freeform "/" style

IBM | Rocket software

pysdsf – Installation And Setup

- pysdsf supplied as part of the standard SDSF SMP/E FMID
- Artifacts created via SMP/E APPLY
- Wheel file distributed in :

  /usr/lpp/sdsf/python/dist/
- Documentation distributed in :

  /usr/lpp/sdsf/python/doc

IBM | Rocket software

pysdsf – Example Code

```
from pysdsf import Connection
from pysdsf.isfexception import ISFException

connection = Connection()
# Get all the jobs with prefix SDSF*
try:
    # Any owner and job name prefix of SDSF*
    status_result = connection.status_of_jobs(prefix='SDSF*',
                                              owner='*')
    # Any row returned?
    if len(status_result):
        print('Job_Name Job_ID     Queue')
        for row in status_result:
            print(
                f'{row["JNAME"].ljust(8)} {row["JOBID"].ljust(8)} '
                f'  {str(row["QUEUE"]).ljust(10)} ')
    else:
        print('Jobs with prefix SDSF* not found')
except ISFException as e:
    print(f'Error occurred response is {e}')
finally:
    connection.close()
```

Following slides will breakdown the statements to explain the general flow and look-and-feel of pysdsf

IBM | Rocket software

pysdsf – Breakdown Of Statements

```
from pysdsf import Connection
from pysdsf.isfexception import ISFException
```

Import the pysdsf package and its allied custom exception

```
connection = Connection()
```

pysdsf uses the "connection" object to facilitate communication with the SDSF server

```
status_result = connection.status_of_jobs(prefix='SDSF*',
                                          owner='*')
```

Named SDSF functions can be passed parameters explicitly or set as a dictionary :

```
options = {
    'prefix':'SDSF*',
    'owner':'*'
}

 status_result = connection.status_of_jobs(**options)
```


IBM | Rocket software

pysdsf – Breakdown Of Statements

```
if len(status_result):
    print('Job_Name Job_ID     Queue')
    for row in status_result:
        print(
            f'{row["JNAME"].ljust(8)} {row["JOBID"].ljust(8)} '
            f'  {str(row["QUEUE"]).ljust(10)} ')
else:
    print('Jobs with prefix SDSF* not found')
```

Process each row of the results and access column data using the internal column name

Internal column names match SDSF REXX

```
except ISFException as e:
    print(f'Error occurred response is {e}')
```

Handle any exceptions from SDSF

```
connection.close()
```

Close the connection with SDSF

IBM | Rocket software

pysdsf – Browse SYSOUT For Active Job

```
from pysdsf import Connection
from pysdsf.isfexception import ISFException

connection = Connection()
# Get all the jobs with prefix SDSFAUX*
options = {
    'prefix': 'SDSFAUX*',
    'owner': '*'
}
try:
    da_result = connection.active_users(**options)
    # Any row returned?
    if len(da_result):
        for da_row in da_result:
            try:
                print(f'Browse result for {da_row["JNAME"]}({da_row["JOBID"]})')
                try:
                    browse_result = da_result.browse(da_row, **options)
                    if browse_result:
                        for line in browse_result:
                            print(line)
                    else:
                        print('Nothing to browse')
                except ISFException as e:
                    print(e)
            except ISFException as e:
                print(e)
    else:
        print('Jobs with prefix SDSFAUX* not found')
except ISFException as e:
    print(f'Error occurred response is {e}')
finally:
    connection.close()
```

connection.active_users = “DA”

Simple interface to browse SYSOUT for job.

pysdsf – Issue Actions Against Jobname(s)

```python
from pysdsf import Connection
from pysdsf.isfexception import ISFException

connection = Connection()

# Define jobs to process
options = {
    'prefix': 'MYJOB',
    'owner': 'MYOWNER',
    'columns': ['JNAME', 'JOBID'],
}
# Get list of jobs to process
try:
    status_result = connection.status_of_jobs(**options)
    if len(status_result):
        for st_row in status_result:
            # Display the job
            print(f'Job {st_row["JNAME"]}({st_row["JOBID"]}) is being processed
            try:
                action_result = status_result.action('D', st_row)
                print("Command responses follow ...")
                for response in action_result:
                    print(response)
            except ISFException as e:
                print(e)
    else:
        print("No jobs to process")
except ISFException as e:
    print(e)
finally:
    connection.close()
```

Simple interface to issue action character against a row

Will result in z/OS operator command being issued and we can examine the response

1
2

pysdsf – Scanning the SYSLOG/OPERLOG

```
import datetime
from pysdsf import Connection
from pysdsf.isfexception import ISFException

# Connection for SDSF. Note: Does not invoke SDSF
connection = Connection()
# Get datetime for the time and date
today: datetime = datetime.datetime.now()
an_hour_ago: datetime = today - datetime.timedelta(hours=1)
# Set the start and stop times to limit records
# Set response line limit to limit number of line to return in response
try:
    sys_log_result = connection.read_syslog(start_date=an_hour_ago.strftime("%m/%d/%Y"),
                                            start_time=an_hour_ago.strftime("%H:%M:%S"),
                                            stop_date=today.strftime("%m/%d/%Y"),
                                            stop_time=today.strftime("%H:%M:%S"),
                                            response_limit=100)
    # List any messages related to the read request
    for row in sys_log_result:
        print(row)
except ISFException as e:
    print(e)
finally:
    connection.close()
```

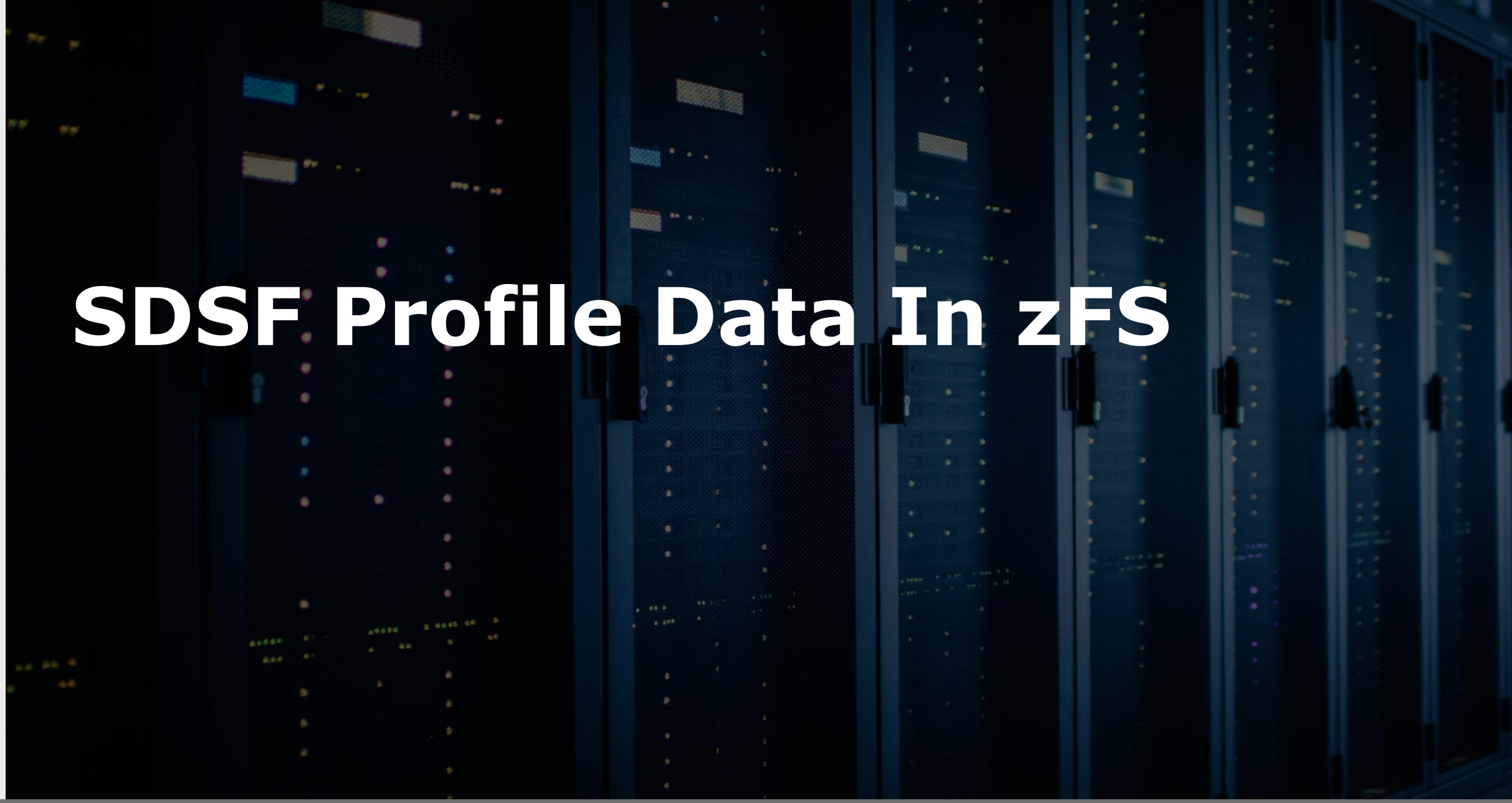
Simple interface to browse and process SYSLOG rows

IBM | Rocket software

pysdsf – Initial Function List

• *active_users*	DA panel
• *address_space_memory*	AS panel
• *browse*	SYSOUT browse
• *held_output_queue*	H panel
• *input_queue*	I panel
• *issue_command*	"/" facility
• *output_queue*	O panel
• *read_syslog/read_operlog*	LOG panel (SYSLOG/OPERLOG)
• *status_of_jobs*	ST panel

IBM | Rocket software



SDSF Profile Data In zFS

SDSF Profile Data in ISPF - Restrictions

- SDSF for z/OS 3.1 and prior releases use ISPF profile to store user settings, preferences including sort, arrange and filter criteria
- ISPF has a fixed limit of 64K for all data within a single xxxxPROF member of the ISPPROF data set
- SDSF users with large amounts of sort, arrange or filter criteria can exceed 64K in size causing errors when trying to VREPLACE/VPUT data into ISFPROF
- Using ISPF for profile data also means that other interfaces cannot retrieve stored user settings

SDSF Profile Data in zFS

- SDSF for z/OS 3.2 introduces the ability to store user profile data in the z/OS UNIX file system
  - Removes the 64K data limit constraint
  - Makes profile data available to non-ISPF interfaces
- Profile data stored as JSON documents
- Positions SDSF to provide further enhancements in later releases based on the JSON document profile store

SDSF Profile Data in zFS Implementation

- New SDSF group keyword “PROFILE”
  - ISPF Use legacy ISPF profile (default)
  - FILESYS Use z/OS UNIX file system
- Profile data stored in the “.isf” directory in user home directory
- New SDSF group keyword “PROFILEPATH”
  - Specifies any additional directory path appended after “.isf”
  - Default is null
- Invocation screen name appended to directory name if not equal to “SDSF”
  - Allows users to have different profile stores for fastpath SDSF invocations from places like the ISPF command table

SDSF Profile Data in zFS Migration

- SDSF profile path will be created if does not exist and any existing ISPF profile settings will be automatically migrated to new JSON format
  - Users can re-migrate by deleting the ".isf" directory and re-invoking SDSF
- Once the SDSF profile path is created, any existing or updated ISPF profile settings will be ignored
  - Data will not be saved to ISPF profile members after migration
- Important to note if sharing SDSF ISFPROF members with downlevel systems
  - SDSF for z/OS 3.1 and below will have no knowledge of profile data in the profile path filesystem
  - Some customers might postpone using FILESYS profiles until all members in sysplex are at least z/OS 3.2

IBM | Rocket software

SDSF Profile Data Considerations

- SDSF users should not edit the JSON documents in the ".isf" profile directory
- Any malformed or invalid JSON data encountered during initialization will cause SDSF to discard all profile path information and fall back to using default settings

RACF Log

RACF Log Panel

- New RLOG panel
- Provides a summary of recent logged RACF activity
  - Includes failures and logged access for traditional resources as well as certain z/OS UNIX information
- Uses the SDSF event log architecture to provide the data
  - SMF record type 80 contents used to populate panel entries
- Drill down to the OPERLOG from row on RLOG panel using “L” action
  - Saves hunting OPERLOG for ICH408I messages
- Direct access to profile information and access list using “LP” and “LA” actions
- RLOG panel can be filtered using FEATENT statements in similar fashion to ELOG
  - New categories of “SAF”, “SAF_CMD” and “SAF_UNIX”

IBM | Rocket software

RACF Log Panel

RS21

File Edit Font Transfer Macro Options Window Help

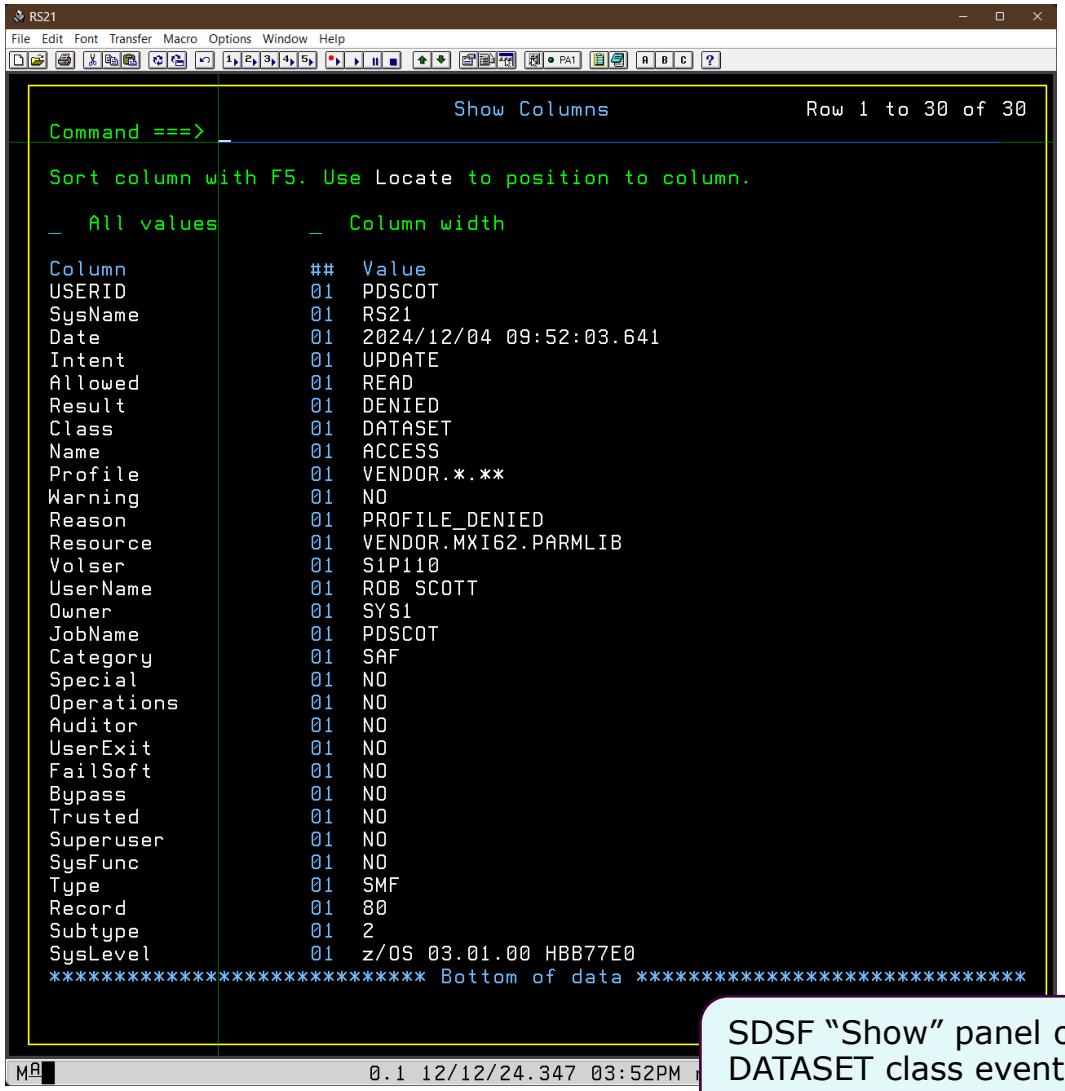
Display Filter View Print Options Search Help

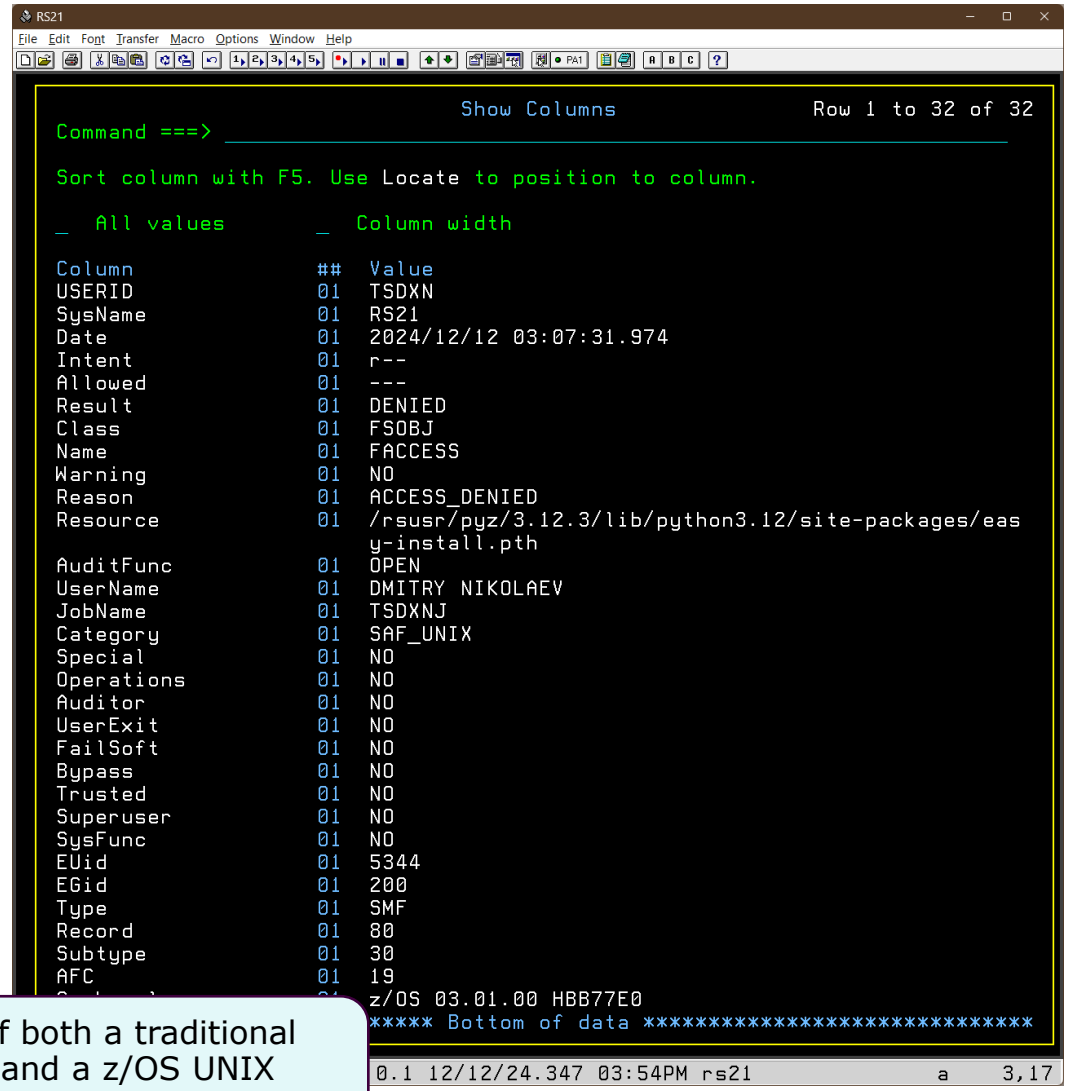
```
SDSF RACF LOG   RS21     (ALL)     ACTIVE    ALL       LINE 105-125 (896)
COMMAND INPUT ===>                                     SCROLL ===> CSR
PREFIX=*  DEST=(ALL)  OWNER=PDSCOT  SYSNAME=*  FILTERS=1
NP    USERID   SysName  Date                    Intent   Allowed  Result   Class    Name                    Profile
      PDZTEAM  RS21     2024/11/19 17:01:02.46  -w-      r--      DENIED   FSOBJ    FACCESS
      TSDXN    RS21     2024/11/19 15:38:19.01  r--      ---      DENIED   DIRACC   DACCESS
      TSDXN    RS21     2024/11/19 15:38:19.01  r--      ---      DENIED   DIRACC   DACCESS
      TSDXN    RS21     2024/11/19 15:38:19.01  r--      ---      DENIED   DIRACC   DACC
      TSDXN    RS21     2024/11/19 15:38:19.01  r--      ---      DENIED   DIRACC   DACC
      PDKOVI   RS21     2024/11/19 14:49:57.18  -w-      r--      DENIED   FSOBJ    FACC
      PDKOVI   RS21     2024/11/19 14:49:57.18  -w-      r--      DENIED   FSOBJ    FACCESS
      PDKOVI   RS21     2024/11/19 14:49:57.18  -w-      r--      DENIED   FSOBJ    FACCESS
      PDPWRES  RS21     2024/11/19 13:46:41.64                   SUCCESS           ALTUSER
      TS5485   RS21     2024/11/19 12:13:59.72  READ     NONE     DENIED   XFACILIT ACCESS                  CKG.**
      TS5485   RS21     2024/11/19 12:13:38.45  READ     NONE     DENIED   XFACILIT ACCESS                  CKG.**
      TS8839   RS21     2024/11/19 11:34:01.50  READ     READ     SUCCESS  DATASET  ACCESS                  RSRTE.SLD*.**
      TS8839   RS21     2024/11/19 11:33:57.40  READ     READ     SUCCESS  DATASET  ACCESS                  RSRTE.SLD*.**
      TS8839   RS21     2024/11/19 11:33:52.97  READ     READ     SUCCESS  DATASET  ACCESS                  RSRTE.SLD*.**
      TS8839   RS21     2024/11/19 11:33:49.71  READ     READ     SUCCESS  DATASET  ACCESS                  RSRTE.SLD*.**
      TS8839   RS21     2024/11/19 11:33:46.28  READ     READ     SUCCESS  DATASET  ACCESS                  RSRTE.SLD*.**
      TS8839   RS21     2024/11/19 11:33:43.42  READ     READ     SUCCESS  DATASET  ACCESS                  RSRTE.SLD*.**
      TS5485   RS21     2024/11/19 11:12:22.46  READ     NONE     DENIED   XFACILIT ACCESS                  CKG.**
      PDPWRES  RS21     2024/11/19 11:08:00.32                   SUCCESS           ALTUSER
      PDPWRES  RS21     2024/11/19 11:08:00.29                   SUCCESS           ALTUSER
      PDPWRES  RS21     2024/11/19 11:08:00.26                   SUCCESS           ALTUSER
```

Both traditional and z/OS UNIX access events are displayed

MA 0.1 11/21/24.326 12:32PM rs21 a 4,22

RACF Log Panel - Detail





SDSF "Show" panel of both a traditional DATASET class event and a z/OS UNIX event.

RACF Log Panel -Columns

- Columns include
  - Class, resource and profile
  - Jobname, User name, application name and owner
  - Access intent and allowed
  - Warning, special, operations, auditor, failsoft, bypass, trusted attributes in effect
  - Any z/OS UNIX EUID and EGID
  - Any log string data and RECVR
  - Any SECLABEL
  - Names for any attempted RACF commands

RACF Log Panel - Actions

- L
  - Invoke LOG (OPERLOG) panel and position to the date-stamp of the logged event
- LA
  - Show the RACF access list for the profile covering the resource in shown in the row
- LP
  - Browse the RACF profile covering the resource in shown in the row

Sysname Lists

SDSF Sysname – Problem Statement

- Cross-system data gathering with a sysplex governed by the SYSNAME setting for the current user
  - If not null and not equal to local sysname, data-gathering request is broadcast to all systems matching the SYSNAME pattern using XCF signalling services
- Sometimes groups of z/OS systems in a sysplex do not follow naming standards that can be easily masked using “*” and “%”
- Users frequently run with SYSNAME set to “*” and then use filtering on the SYSNAME to discard unwanted rows
  - Increased overhead for data gathering on unnecessary systems
  - Increased chance of “missing data” situation for test systems or systems under stress

IBM | Rocket software

Sysname Lists - Implementation

- New ISFPRMxx statements LIST and LISTENT
- LIST statement declares the name and type of the defined list
- LISTENT statement declares the list items for the current list
- Assuming three systems “PROD”, “DEV” and “TEST” in a sysplex

```
LIST NAME(XTEST) TYPE(SYSNAME)
    LISTENT NAME(PROD)
    LISTENT NAME(DEV)
```

- User can issue “SYSNAME XTEST” and data gathering requests will be sent ONLY to “PROD” and “DEV”
- SET DISPLAY ON will show “/L” appended to SYSNAME setting to indicate that a sysname list is being used
- Can also be used in the RSYS command to filter WTORs shown on LOG panel
  - SR panel can also take advantage with custom property : *Panel.SR.EnableRsysFilter*

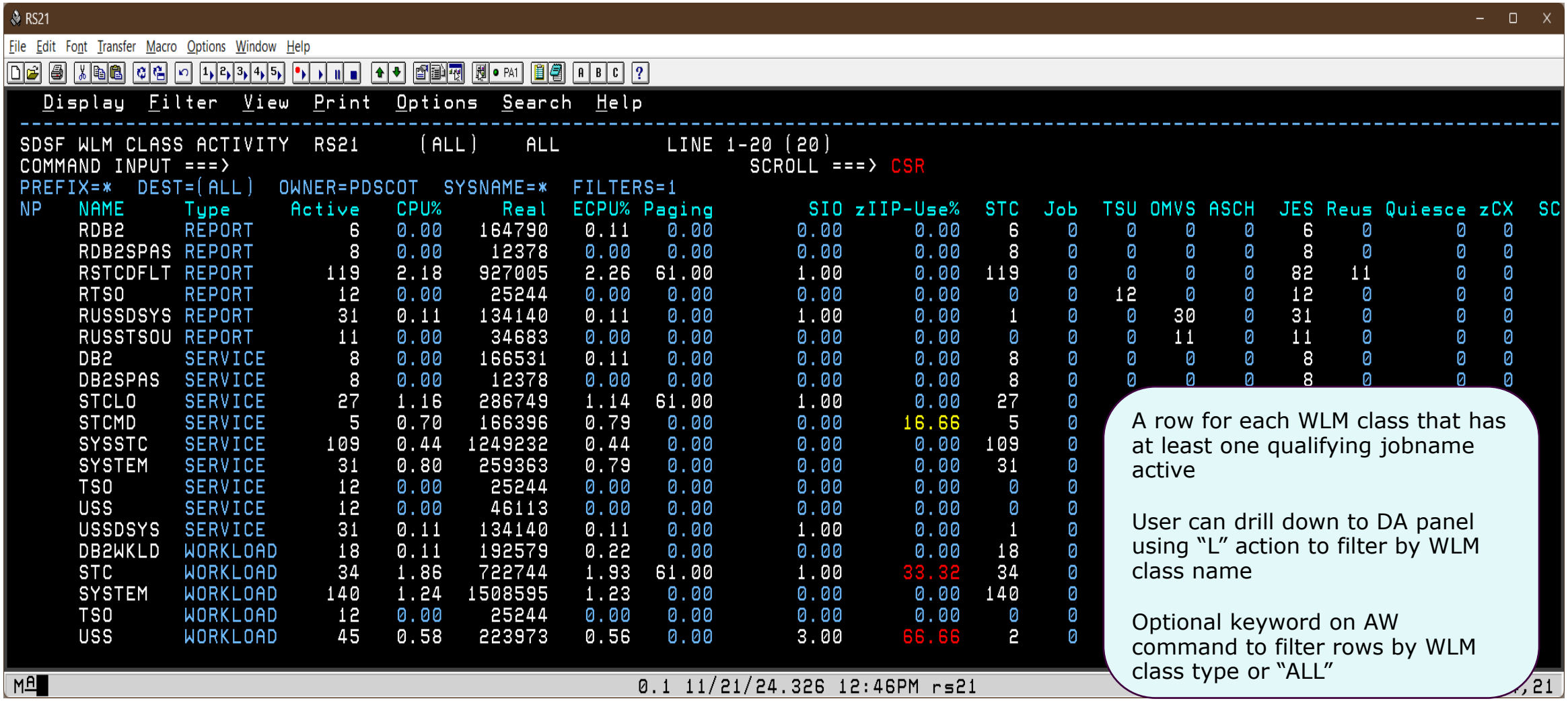
Other New SDSF Panels

New SDSF Panels

AW	Active Job Summary By WLM Class
CAT	Master And User Catalogs
CMO	64-bit Common Memory Objects
DEVS	Device Space
FXE	z/OS Function Registry
JRU	JES2 Resource Usage by Userid
MFP	z/OS UNIX Path Module Fetch
NAP	TCP/IP Activity Summary By Port
PST	z/OS UNIX Threads
RACD	RACF Data Sets
RACF	General RACF Information
RACR	RACF RRSF Nodes
SMFL	SMF Logstreams
SMFR	SMF Real Time Resources
VTOC	DASD Volume Table Of Contents

© Rocket Software, Inc. or its affiliates 1990 – 2023. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates
© Copyright IBM Corporation 2023. IBM, the IBM logo, ibm.com, and Watson are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

AW Panel – Address Space Summary By WLM Class

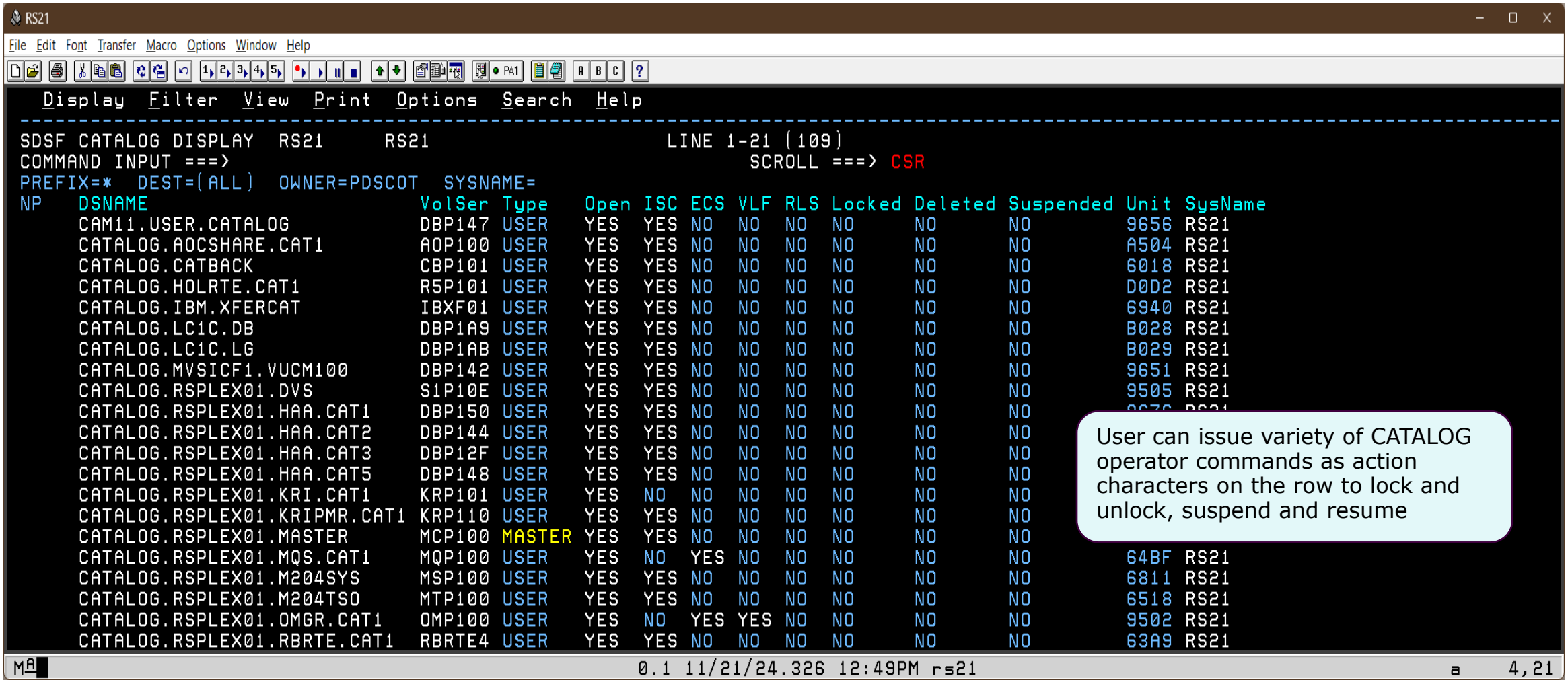


A row for each WLM class that has at least one qualifying jobname active

User can drill down to DA panel using "L" action to filter by WLM class name

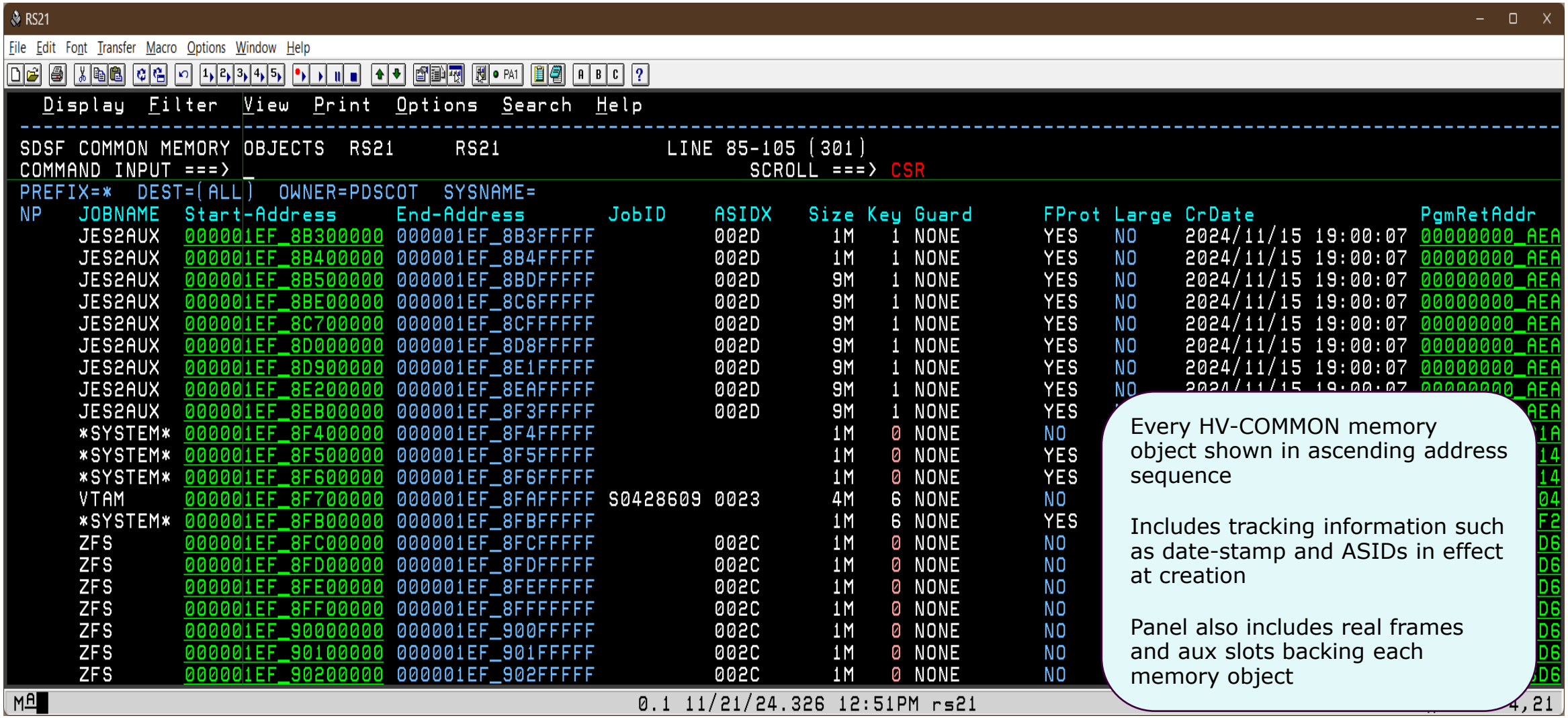
Optional keyword on AW command to filter rows by WLM class type or "ALL"

CAT Panel – Master and User Catalogs



3
3

CMO Panel – 64-Bit Common Memory Objects

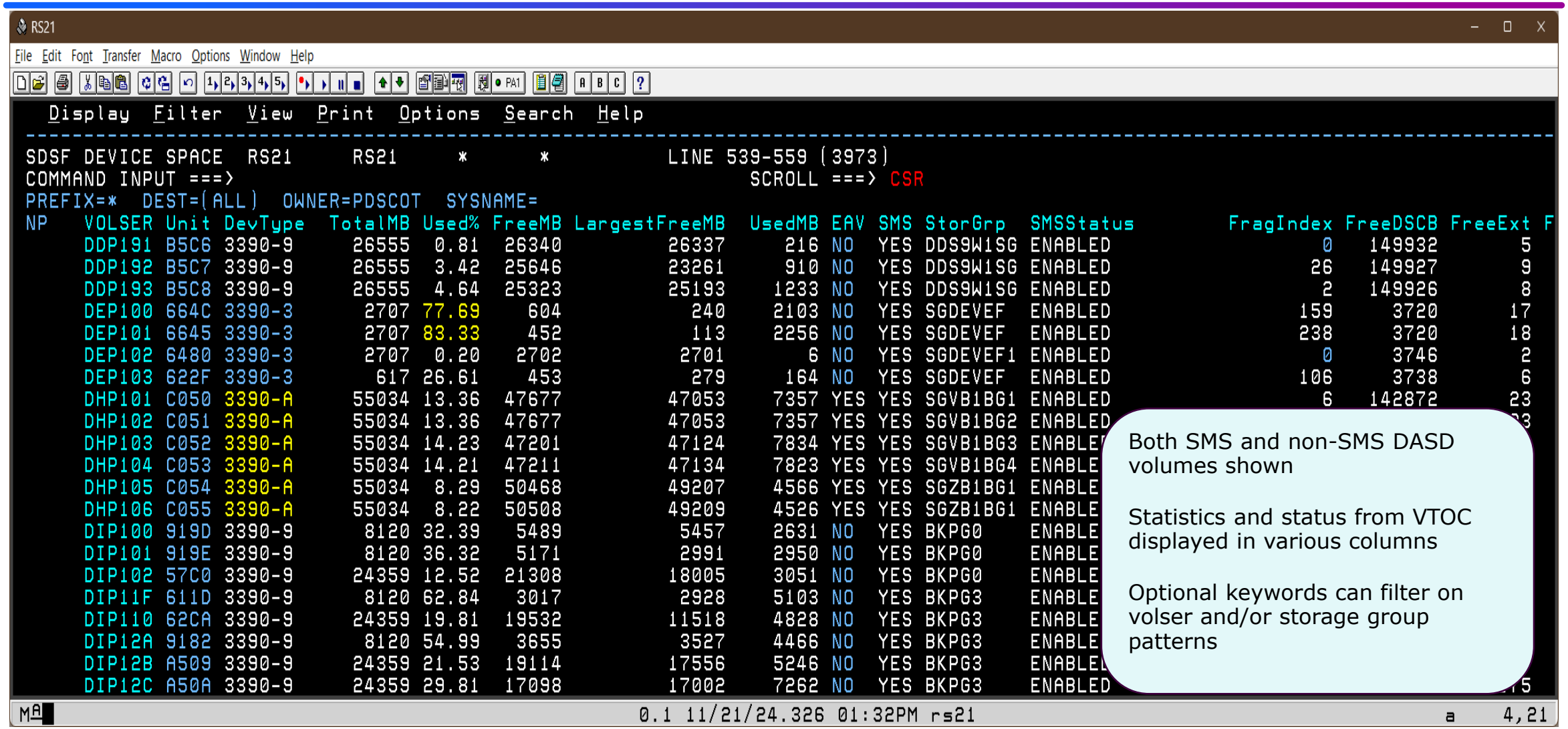


Every HV-COMMON memory object shown in ascending address sequence

Includes tracking information such as date-stamp and ASIDs in effect at creation

Panel also includes real frames and aux slots backing each memory object

DEVS Panel – Device Space

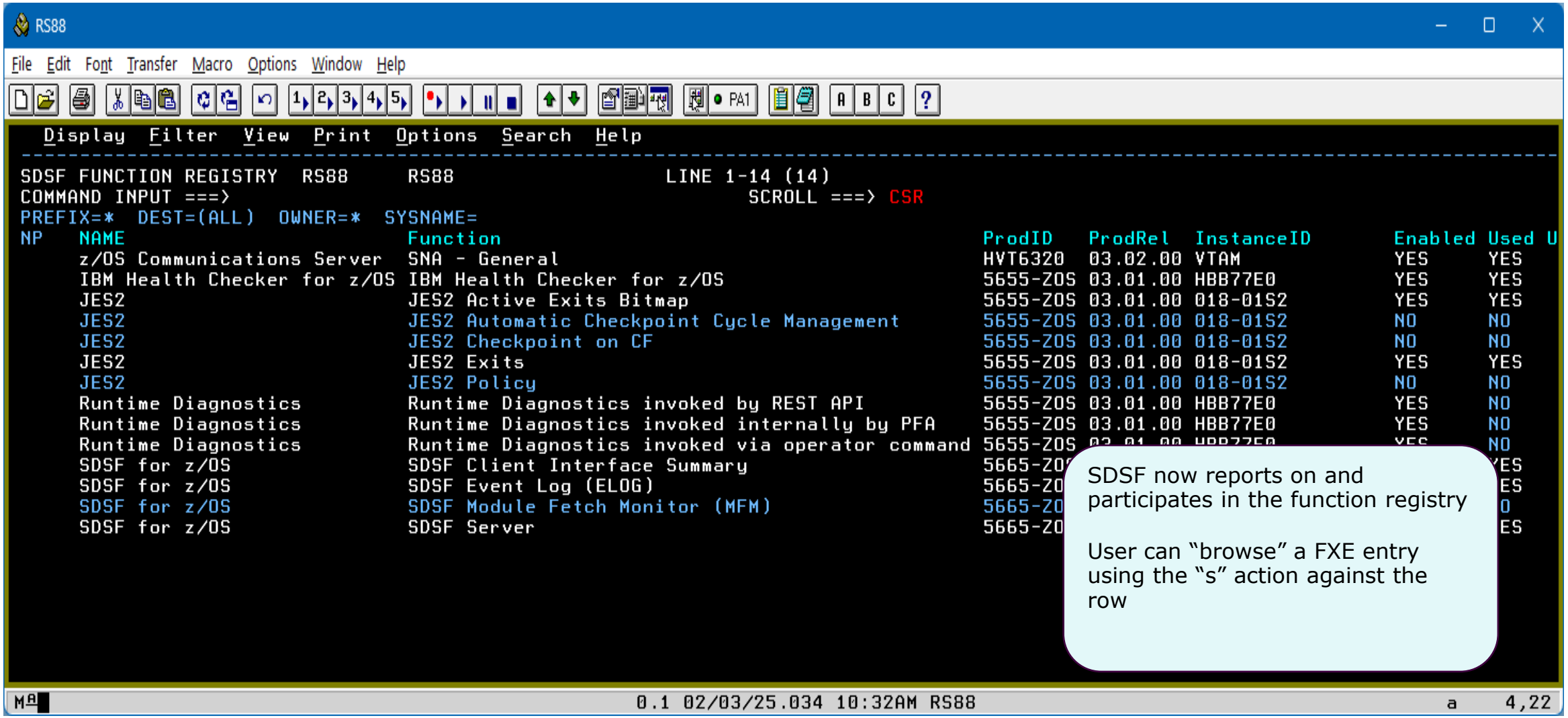


Both SMS and non-SMS DASD volumes shown

Statistics and status from VTOC displayed in various columns

Optional keywords can filter on volser and/or storage group patterns

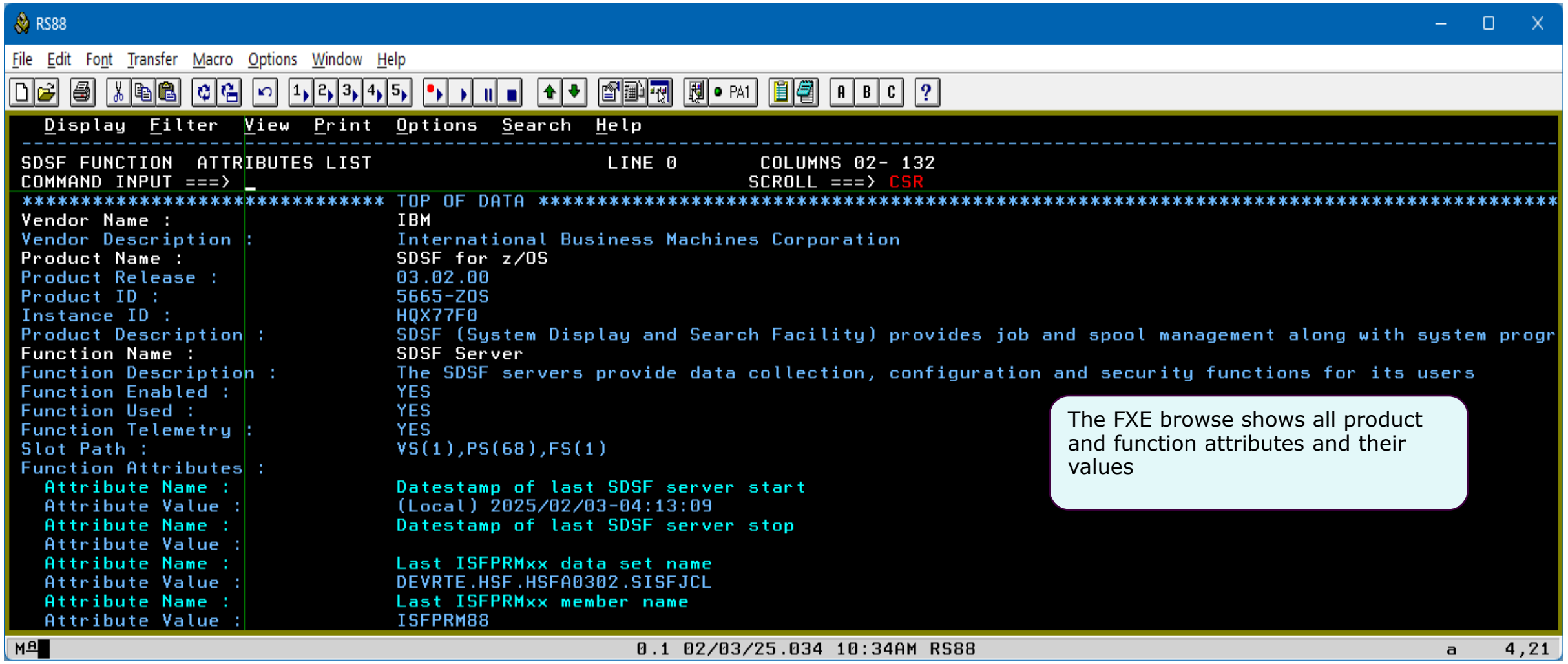
FXE Panel – z/OS Function Registry



SDSF now reports on and participates in the function registry

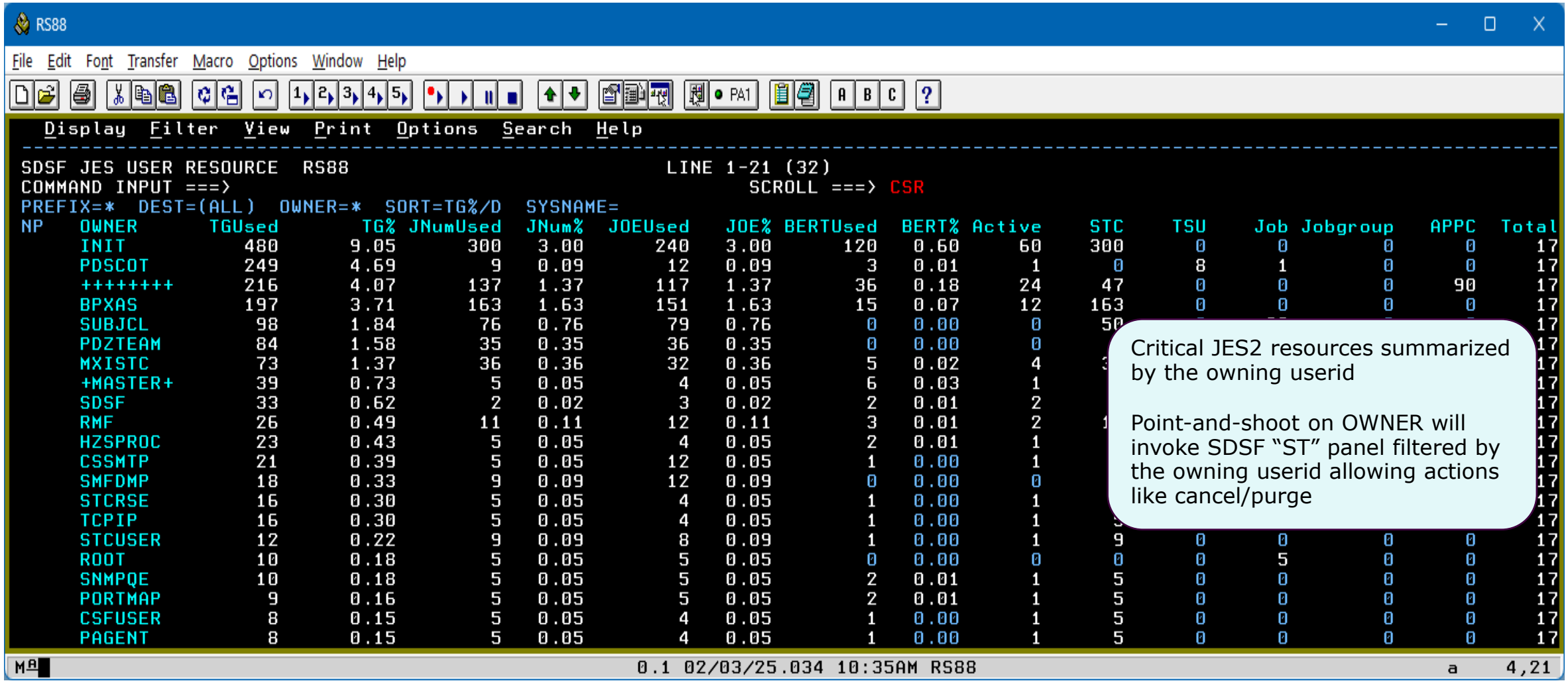
User can "browse" a FXE entry using the "s" action against the row

FXE Panel – z/OS Function Registry Browse



The FXE browse shows all product and function attributes and their values

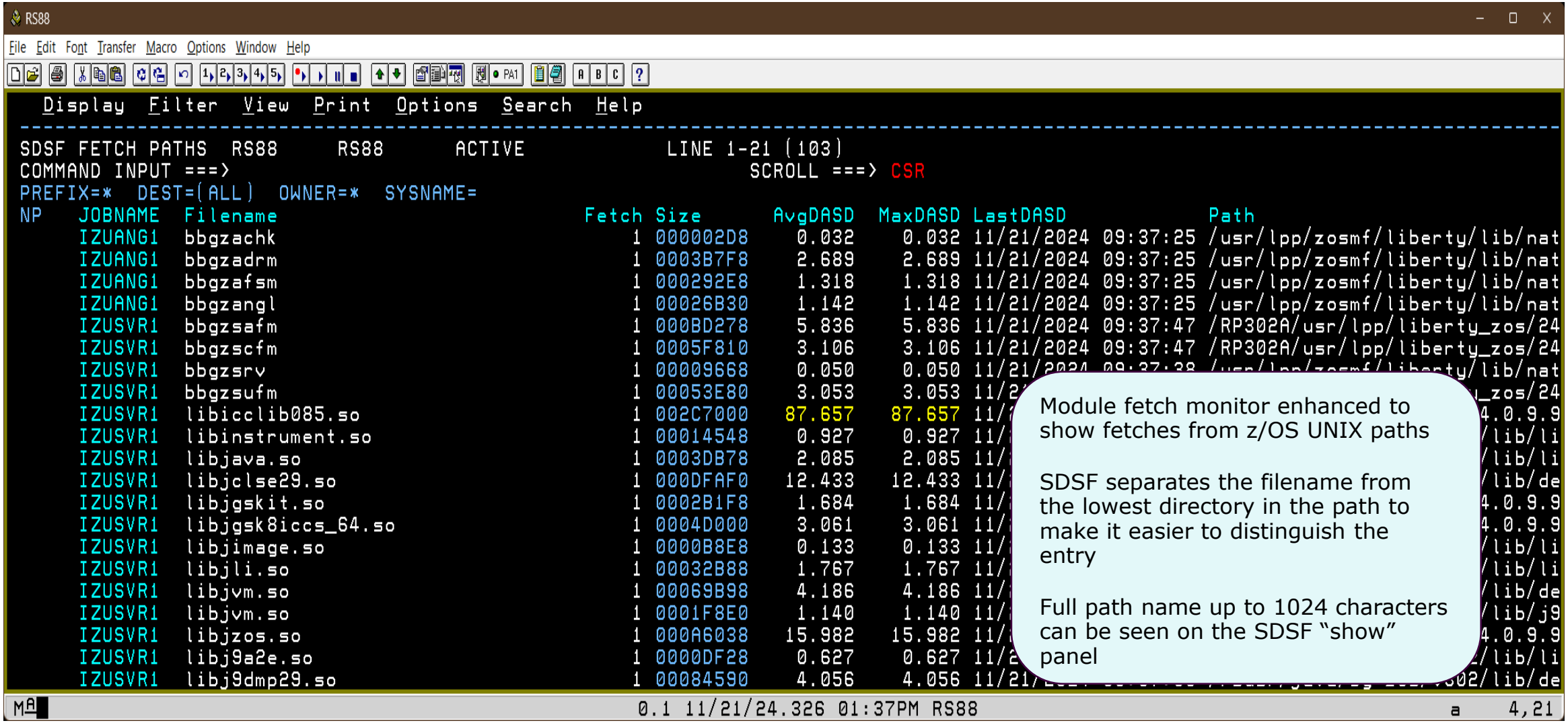
JRU Panel – JES2 Resource By Userid (Owner)



Critical JES2 resources summarized by the owning userid

Point-and-shoot on OWNER will invoke SDSF "ST" panel filtered by the owning userid allowing actions like cancel/purge

MFP Panel – Module Fetch UNIX Path

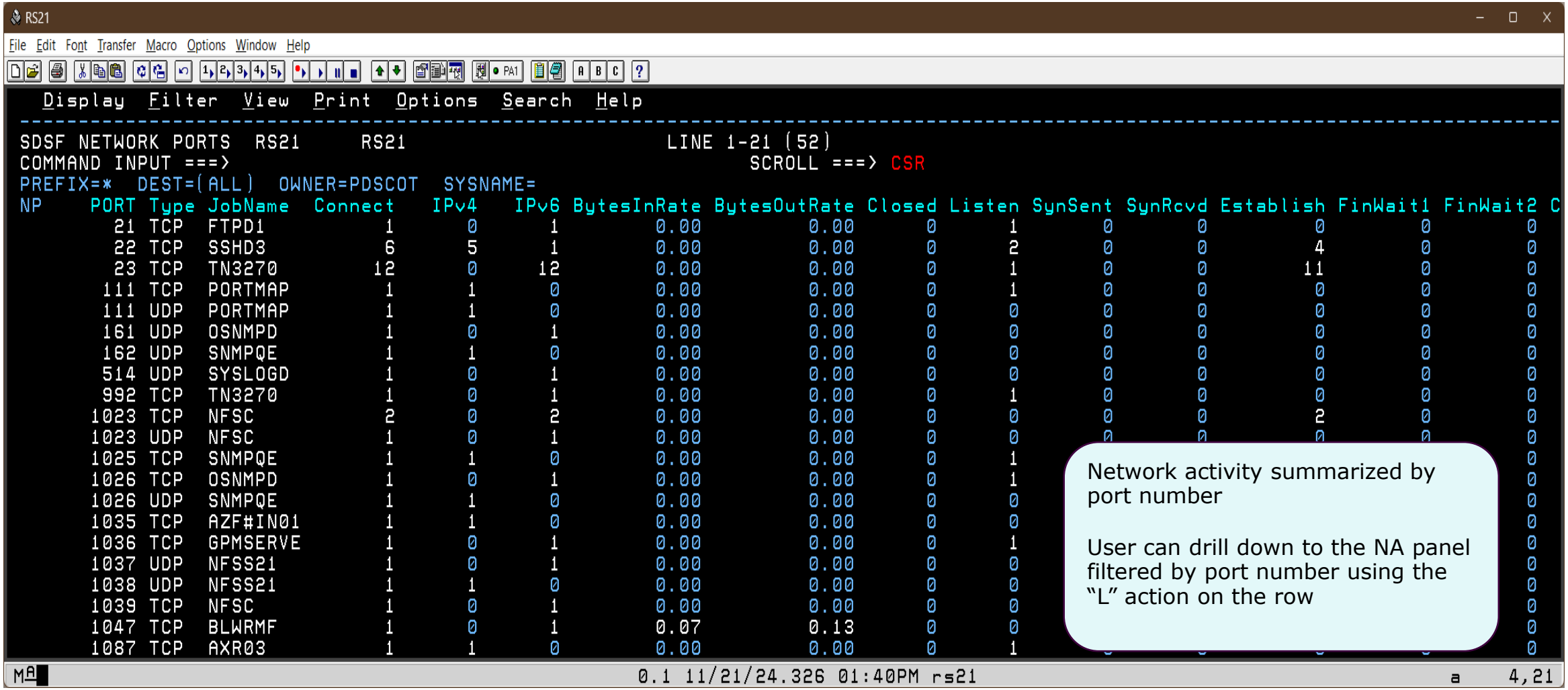


Module fetch monitor enhanced to show fetches from z/OS UNIX paths

SDSF separates the filename from the lowest directory in the path to make it easier to distinguish the entry

Full path name up to 1024 characters can be seen on the SDSF “show” panel

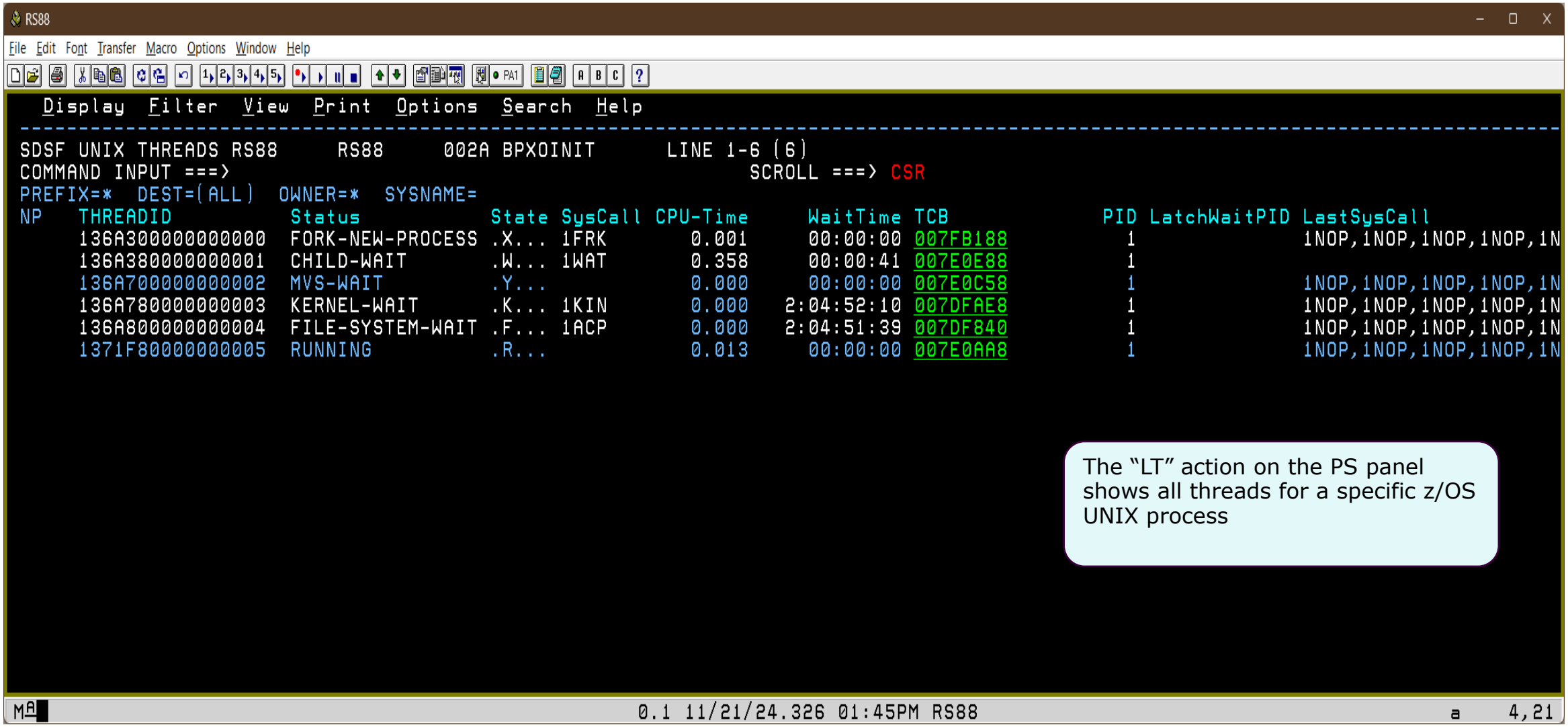
NAP Panel – Network Activity By Port Number



Network activity summarized by port number

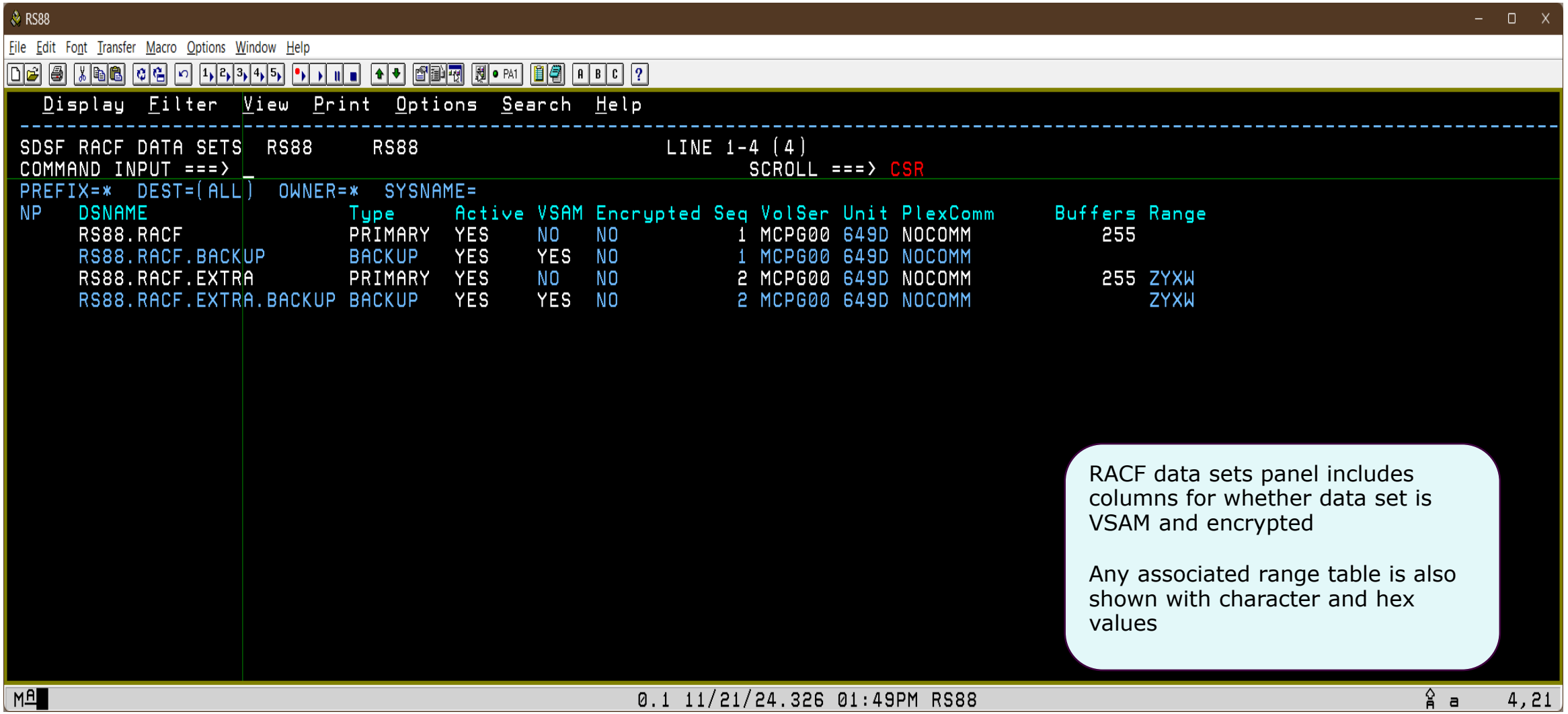
User can drill down to the NA panel filtered by port number using the "L" action on the row

PST Panel – z/OS UNIX Process Threads



The "LT" action on the PS panel shows all threads for a specific z/OS UNIX process

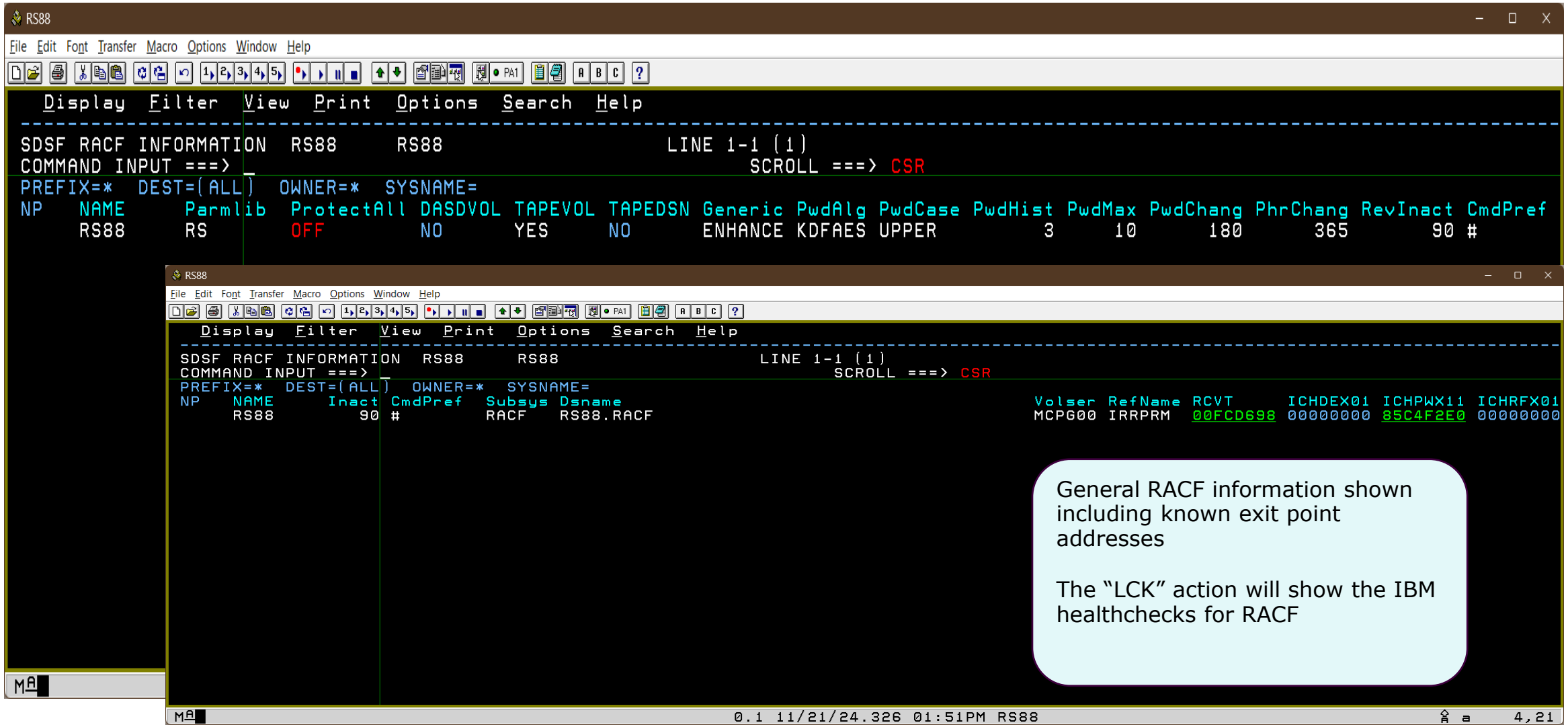
RACD Panel – RACF Data Sets



RACF data sets panel includes columns for whether data set is VSAM and encrypted

Any associated range table is also shown with character and hex values

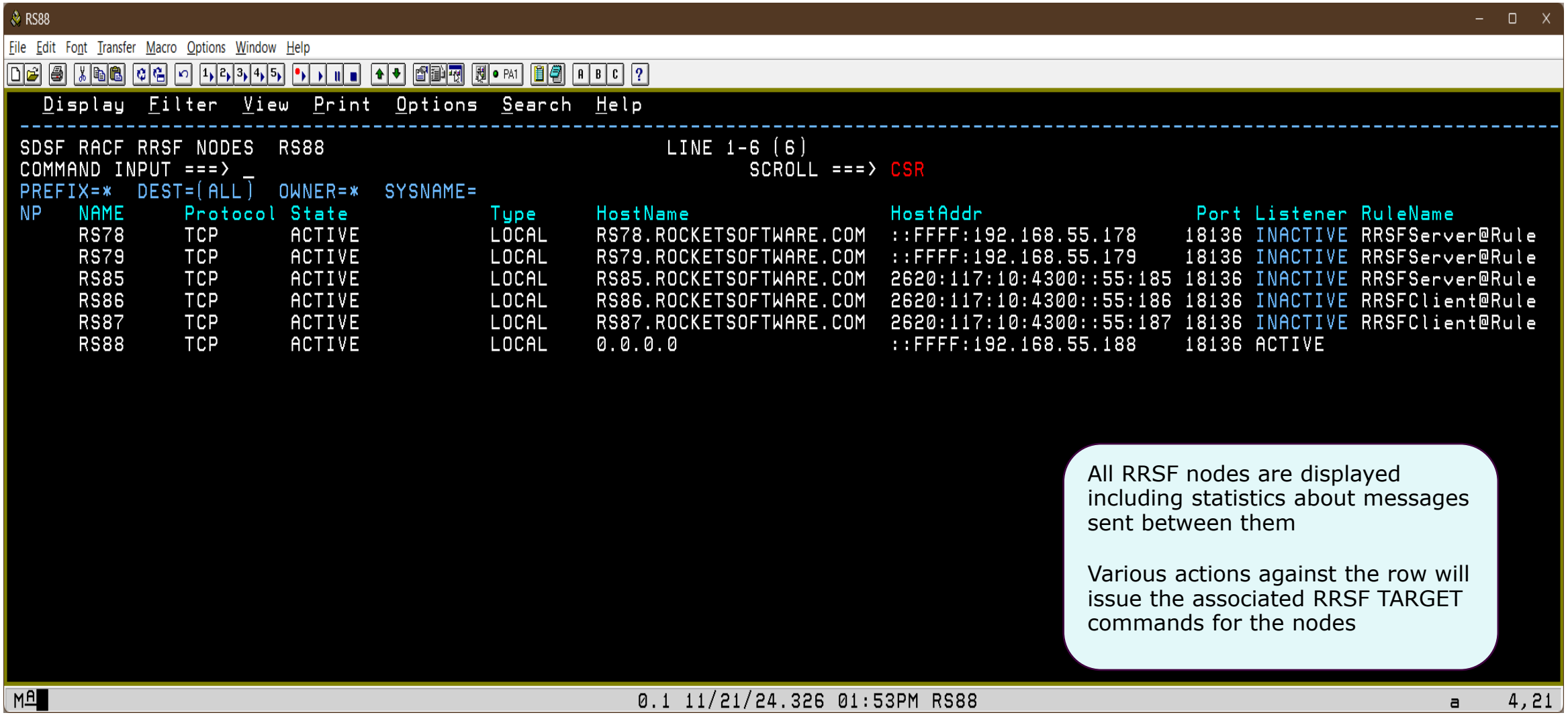
RACF Panel – General RACF Information



General RACF information shown including known exit point addresses

The "LCK" action will show the IBM healthchecks for RACF

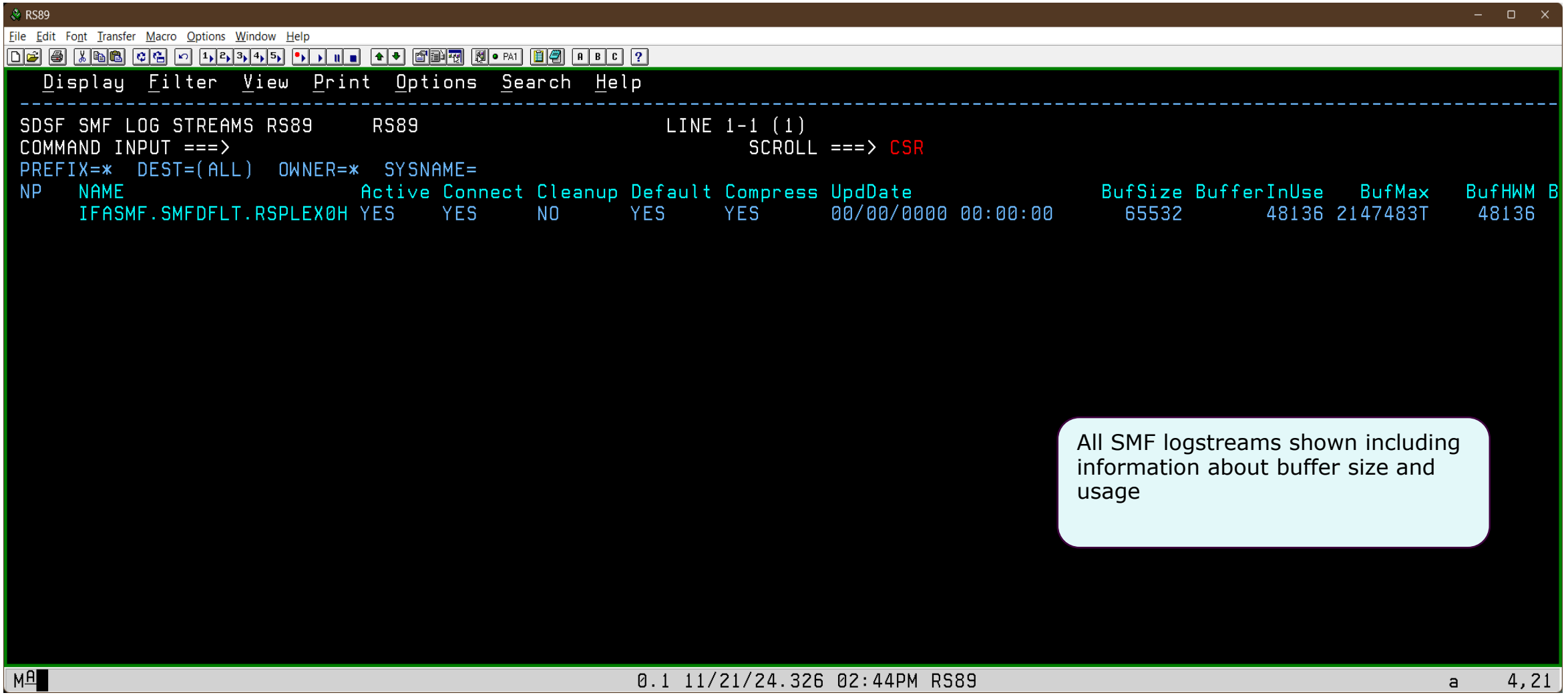
RACR Panel – RACF RRSF Nodes



All RRSF nodes are displayed including statistics about messages sent between them

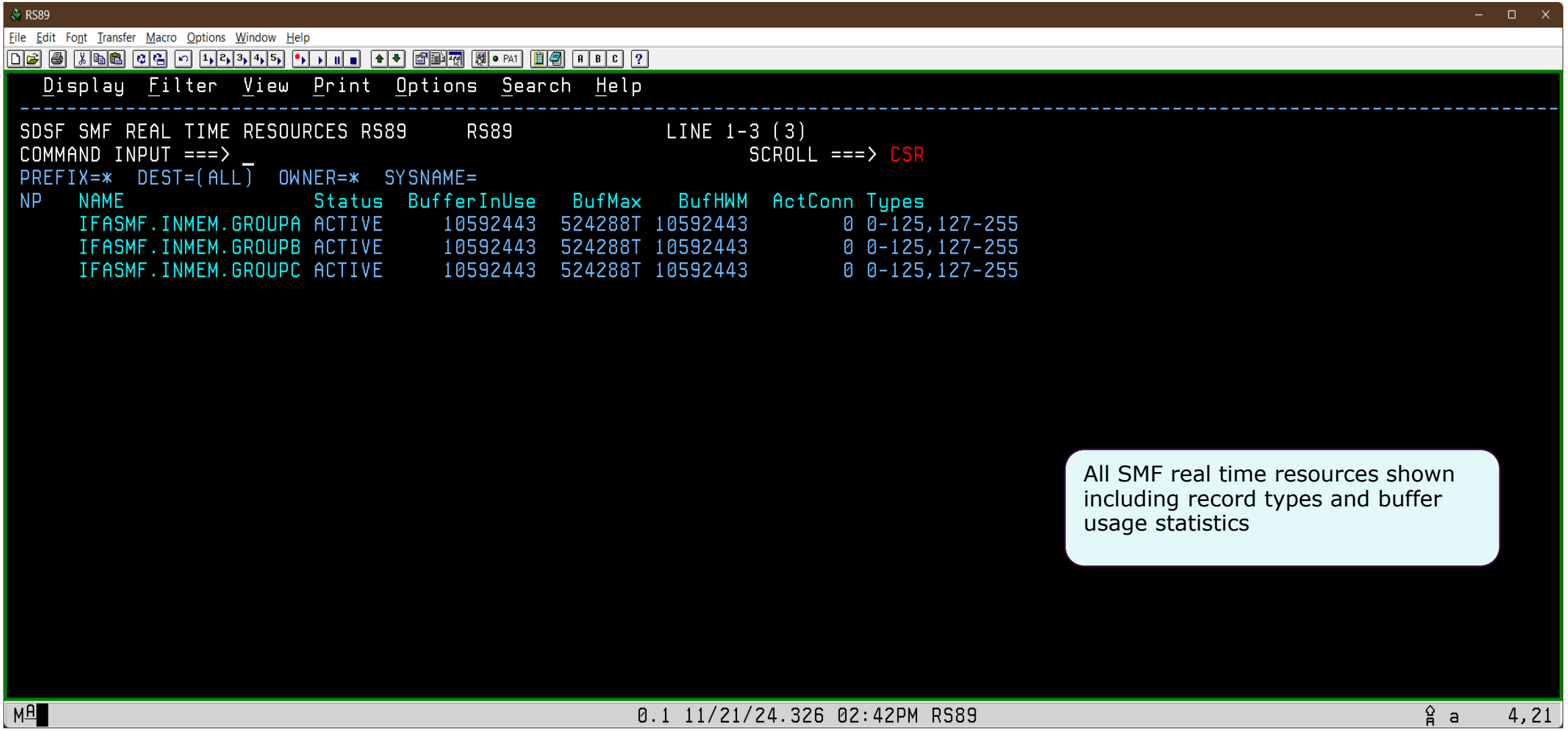
Various actions against the row will issue the associated RRSF TARGET commands for the nodes

SMFL Panel – SMF Logstreams



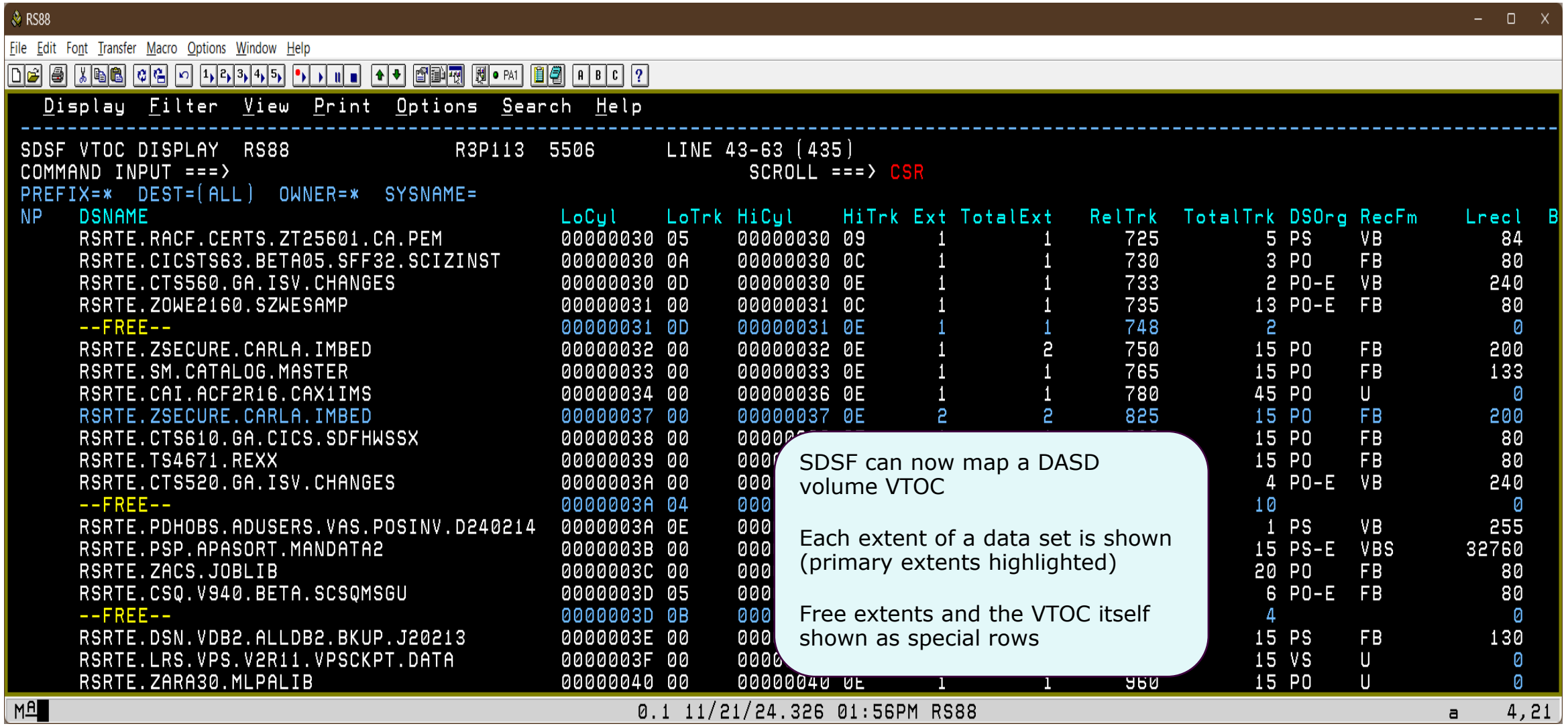
All SMF logstreams shown including information about buffer size and usage

SMFR Panel – SMF Real Time Resources



All SMF real time resources shown including record types and buffer usage statistics

VTOC Panel – DASD Volume Table Of Contents



SDSF can now map a DASD volume VTOC

Each extent of a data set is shown (primary extents highlighted)

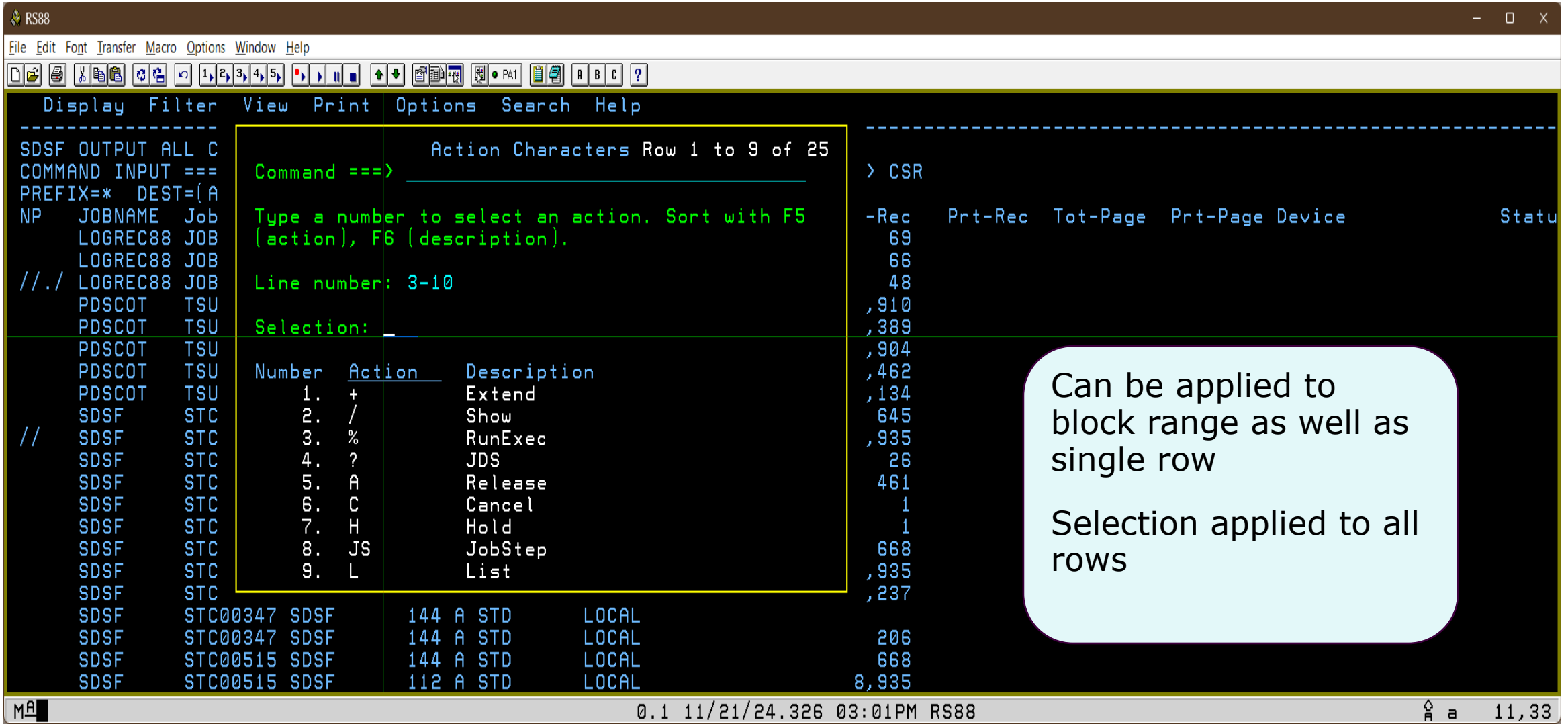
Free extents and the VTOC itself shown as special rows

Useability Enhancements

Action Prompt Panel

- New concept of SDSF “control” action char of a single dot “.”
  - Used in a couple of new enhancements
  - Common action character across all tabular panels
- Using the characters “./” in ISPF against a row will now cause a pop-up panel showing all actions defined for the panel in a scrollable list
- User can select entry to apply the action to the row(s)
- No longer need SET ACTION ON to see available actions

Action Prompt Panel

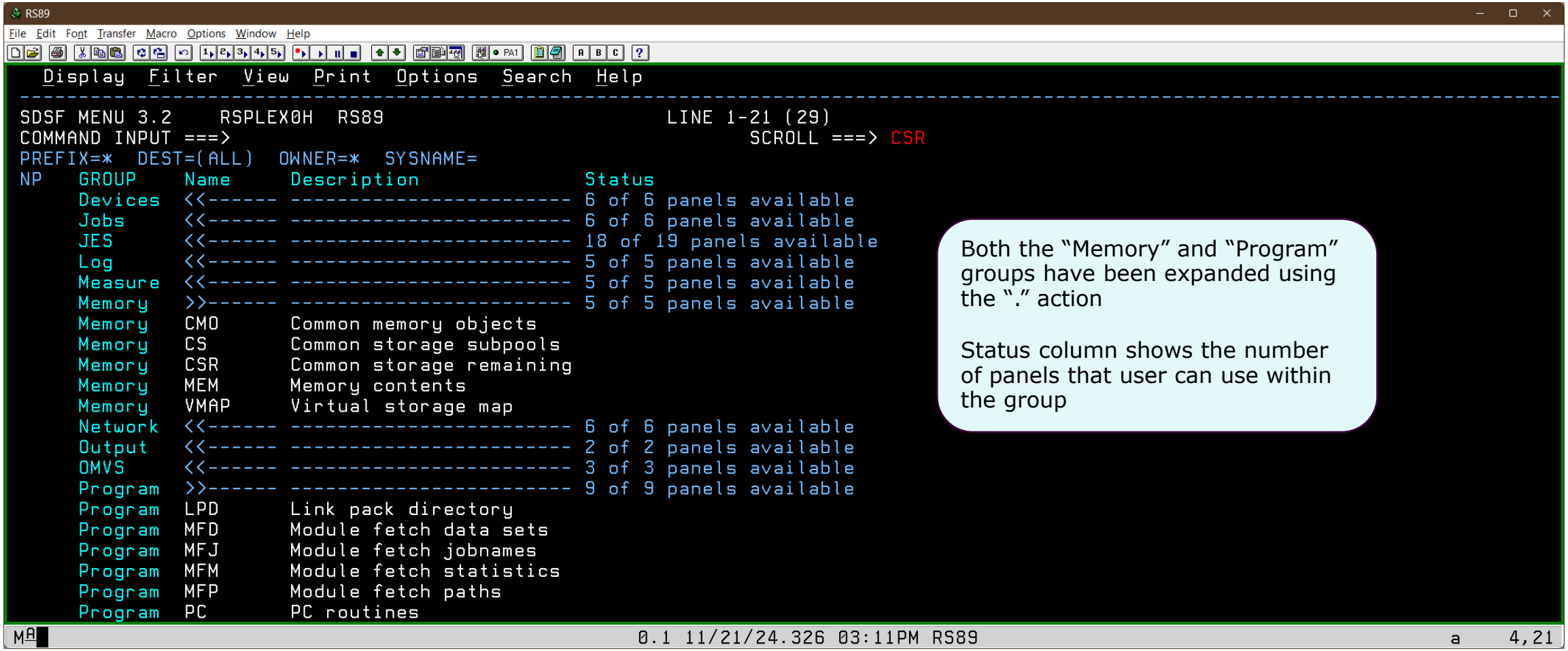


MGRP Panel - Main Menu Group

- Groups SDSF commands by their associated group name
- Allows user to expand or collapse individual groups to show associated commands
- Expand/collapse status of each group saved in user profile
  - Use “.” to expand and “..” to collapse
- MGRP command can be issued anywhere within product
- Use SET MAIN GROUP command to have this as the default SDSF main menu

IBM | Rocket software

MGRP Panel - Main Menu Group



Both the “Memory” and “Program” groups have been expanded using the “.” action

Status column shows the number of panels that user can use within the group

Miscellaneous Enhancements

Common Action Characters

- Several new common actions across multiple panels
- LCK
  - Lists health checks for the associated component for the row object
  - Panels include APF, CFC, FS, JES, LNK, RACF, VMAP
- LE
  - Lists WLM enclaves for row object
  - Panels include AD, AS, AW, DA, REPC, RGRP, SRVC and WKLD
- LVT
  - Invokes the DASD VTOC panel for the volser in the row
  - Panels include CAT, DEV, DEVS, SMSV
- LA and LP
  - List RACF profile/access for resource protecting the data in the row
  - Panels include APF, CAT, JDD, LNK, PARM, RLOG
  - Point and shoot on ULOG SECTRACE entries invokes LA action

Notable New/Updated Columns On Existing Panels

- AUTOMNT and PDevice on FS panel
- NoTypes column on SMFS
- zCX column on AD, AS and DA to indicate running container extensions
- Executable memory indicator on JM detail panel
- MemLimChg on AS to show how much 64-bit memory is actually charged towards the address space MEMLIMIT value.
- Extent and EffExtent on LLS panel
  - Enables monitoring of link list data set extents
  - Extent is number of physical extents for the data sets
  - EffExtent is the effective extents when considering that PDSE data sets count as single extent for link list regardless of physical extents

Other Enhancements

```
SDSF PROCESS   DETAILS                                    LINE 0        COLUMNS 02- 133
COMMAND INPUT ===>                                                    SCROLL ===> CSR
******************************* TOP OF DATA ***********************************************
JOBNAME:
  TS78971

JOBID:
  STC09906

OWNER:
  TS7897

PROCESS ID:
  16842889

PARENT PROCESS ID:
  65672

SYSNAME:
  RS79

SYSLEVEL:
  z/OS 03.02.00 HBB77F0

COMMAND:
  java -cp /RS79/u/ts7897/jackson/2.6.3/jackson-databind-2.6.3.jar:/RS79/u/ts789
  7/jackson/2.6.3/jackson-core-2.6.3.jar:/RS79/u/ts7897/jackson/2.6.3/jackson-an
  notations-2.6.3.jar:/RS79/u/ts7897/junit/junit5/org.junit.vintage.engine_5.7.1
  .v20210222-1948.jar:/RS79/u/ts7897/junit/junit5/org.junit_4.13.0.v20200204-150
  0.jar:/RS79/u/ts7897/junit/junit5/org.hamcrest.core_1.3.0.v20180420-1519.jar:/
  RS79/u/ts7897/sdsf/v3r2/javatest/isfjcallTest.jar:/RS79/u/ts7897/sdsf/v3r2/jav
  atest/isfjcall.jar:. -enableassertions -Djava.util.logging.config.file=logging
  .properties com.ibm.zos.sdsf.corex.test.TestISFFileSystemSampleX
```

L action on PS

Will show (via SDSF browse) the full command text for a z/OS UNIX process

5
6

Other Enhancements

- Programmer name on I, O, H and ST is no longer a delayed field
  - Will need JES2 $ACTIVATE level of Z32
  - Resolves long-standing requirement and performance improvement
- Long fields on SHOW panel
  - The SDSF show action “/” pop-up panel will display all characters for very long field data, for example z/OS UNIX paths
  - Characters will be continued on multiple lines in the scrollable area

- For more SDSF presentations :
- github.com/IBM/IBM-Z-zOS/tree/main/zOS-Education/