



# ***Spool Converter User Guide***

***Version V6R1M0  
2010***

# **Table of Contents**

Introduction .....	10
What CoolSpools can do for you .....	10
Automated distribution of reports and documents .....	10
Information sharing .....	11
Report Enhancement .....	11
Archiving and offline storage .....	11
Document Formats .....	12
Adobe® PDF (Portable Document Format).....	12
XML.....	12
HTML (Hypertext Mark-up Language).....	12
Excel Format .....	12
RTF (Rich Text Format) .....	13
Delimited ASCII Text.....	13
Plain ASCII Text.....	13
TIFF format .....	13
Archive format.....	13
Upgrade Notes .....	14
License Keys .....	14
Warning/Disclaimer.....	15
Minimum OS/400 Release Level .....	15
Product Library .....	16
Memo to Users .....	16
Changes between V4 and V5 .....	16
Changes between V3 and V4 .....	17
Changes between V2 and V3 .....	17
What's new in CoolSpools Version 6? .....	20
System Requirements .....	24
Installation .....	25
Maintenance.....	26
Getting Started with CoolSpools Spool Converter.....	27
Using styles.....	29
Using conditional formatting .....	30
Using encrypted passwords .....	31
Using parameter sets .....	31
Where Did My Output Go? .....	32
The TOSTMF parameter .....	32
Understanding IFS path names .....	32
Choosing where to store your output .....	35
Root File System.....	36
QDLS File System.....	37
QNTC File System .....	37
Typical Solutions .....	38
CoolSpools Commands.....	41
CoolSpools Variables .....	43
CoolSpools Functions .....	48
Command Parameters .....	57
Basic Parameters .....	57
FROMFILE – From spooled file name.....	57
TOSTMF - To stream file name or *FTP .....	58

JOB - Job name .....	62
SPLNBR - Spooled file number .....	64
STMFOPT – Stream file option .....	65
Additional Parameters .....	67
APYSTYLES – Apply styles .....	67
AUT - Public data authority .....	70
BLANKS - Include blank lines? .....	71
BMARKKEY- PDF bookmark string key .....	72
BMARKPOS- PDF bookmark string position .....	73
BOOKMARK - PDF bookmarks .....	74
CNDFMTGRP – Conditional formatting groups .....	76
CNDFMTRULE – Conditional formatting rules .....	78
COLOR - Colors .....	87
COLWIDTHS – Column widths .....	92
CSV - CSV Options (CVTSPLCSV command) .....	93
CUSTOMPAGE – Custom page size .....	98
CVTFONTID - Convert font ids .....	99
CVTFNTRSC – Convert font resources .....	103
DBCS - DBCS conversion options .....	106
DELIMITERS - Delimited file options .....	108
DFNSTYLES – Define styles .....	111
DTACPR – Data compression .....	128
EMAIL – Email the output? .....	129
EMAILFILE – Email address file .....	131
EMAILFROM - Email sender information .....	134
EMAILMSG – Email message .....	136
EMAILOPT – Email options .....	138
EMAILSQL – Email address SQL .....	143
EMAILTO - Email recipient(s) .....	145
EXCEL – Excel Options .....	149
EXCLLINKEY - Exclude lines by key .....	161
EXCLLINNBR – Exclude Line Numbers .....	163
EXCLPAGKEY – Exclude pages by key string .....	165
EXCLPAGNBR – Exclude pages by page number .....	166
EXITPGM – Exit Programs .....	168
EXITPGMPRM - Exit Program Parameters .....	171
EXITPGMKEY - Exit program parameters string key .....	173
EXITPGMPOS - Exit program parameters string position .....	175
FONT – Font options .....	177
FTP – FTP parameters .....	181
HTML – HTML options .....	185
INCLFILE – Include image files .....	187
MARGINS - PDF margins and alignment .....	193
OPTIONS – Miscellaneous command options .....	197
OUTPTY – Output priority .....	200
OUTQ – Output queue .....	201
OWNER – New spooled file owner .....	202
PAGEOPTION – Page options .....	203
PAGESIZE - Page Size .....	205
PASSWORD – PDF Security .....	208
PDF – PDF options .....	212
PRTDEV – Printer device .....	218

RPLXLSSHT– Replace Excel worksheet names .....	220
RSCDIR – Resource directory .....	221
RTF – RTF options.....	222
RTVPRMSET – Retrieve Parameter Set.....	224
SIGNATURE - Digital signing options .....	225
SPLFCCSID – Spooled File CCSID .....	229
SPLIT - Split spooled file .....	230
SPLITKEY – Split by key options .....	234
SPLITPOS - Split by position options.....	237
SPLITPAGE – Split file every n pages .....	239
STMFCODPAG – Stream File Code Page.....	240
TEXT – Text options .....	243
TITLE - Title for HTML or PDF .....	248
TOFILE - To spooled file name .....	249
COLUMNOPT – Column creation options.....	250
COLUMNPOS –Column positions.....	252
LINTYPES –Line types .....	254
XLSADJUST – Adjust pages to.....	257
XLSFIT – Fit pages to .....	258
XLSCOLUMNS – Excel columns .....	259
XLSPRINT – Excel print setup .....	260
XLSPROTECT – Excel worksheet protection .....	263
XLSPRPRTY – Document properties.....	265
RSTSPLF Command .....	267
FROMSTMF – From stream File.....	267
NEWOWN – New owner .....	267
OUTPTY – Output priority .....	268
MRGPDF .....	269
FROMPDF - PDF files to merge.....	269
TOPDF - Merged PDF file.....	270
REPLACE - Replace existing PDF .....	270
PASSWORD - Merged PDF file security .....	270
NOTFOUND - File not found action .....	272
AUT - Public data authority .....	272
ADDPDFSGN .....	274
PDFFILE - PDF file .....	274
PDFPWD – Owner password.....	274
CTFFILE - Certificate file.....	274
CTFPWD – Certificate password .....	274
REASON – Reason for signing .....	275
LOCATION - Location .....	275
CONTACT - Signing contact information .....	275
VISIBLE - Visible signature? .....	275
PAGNBR - Show on page number.....	276
IMAGE - Display image file .....	276
XCOORD - X coordinate .....	276
YCOORD - Y coordinate .....	276
WIDTH - Width .....	276
HEIGHT - Height.....	277
Report Definitions and Report Maps.....	278
Commands related to Report Definitions .....	279
CRTRPTDFN – Create Report Definition .....	279

REPORTNAME – Report definition name .....	279
LPI – Lines per inch .....	279
CPI – Characters per inch .....	280
TEXT ‘description’ .....	281
CCSID - Spooled file CCSID .....	281
DATFMT - Spooled file date format .....	281
DATSEP - Spooled file date separator .....	281
CURSYM - Spooled file currency symbol .....	282
DECPOINT - Spooled file decimal point .....	282
THOUSANDS - Spooled file 1000s separator .....	282
CHGRPTDFN – Change Report Definition .....	283
CPYRPTDFN – Copy Report Definition .....	283
FROMREPORT – From report definition name .....	283
TOREPORT – To report definition name .....	283
DLTRPTDFN – Delete Report Definition .....	283
REPORTNAME – Report definition name .....	283
CHKDEPMAP – Check dependent maps .....	283
DSRPRTDFN – Display Report Definition .....	283
RNMRPTDFN – Rename Report Definition .....	284
REPORTNAME – Report definition name .....	284
NEWREPORT – New report definition name .....	284
RTVRPTDFN – Retrieve Report Definition .....	284
REPORTNAME – Report definition name .....	284
SRCFILE – Source file .....	284
SRCMBR – Source member .....	284
MBROPT – Source member .....	284
SAVRPTDFN – Save Report Definition .....	284
REPORTNAME – Report definition name .....	285
TOSTMF – To stream file .....	285
REPLACE - Replace existing stream file .....	285
AUT- Public data authority .....	285
RSTRPTDFN – Restore Report Definition .....	285
REPORTNAME – Report definition name .....	286
FROMSTMF – From stream file .....	286
REPLACE - Replace existing report .....	286
WRKRPTDFN – Work with Report Definition .....	286
DSNRPTDFN – Design Report Definition .....	286
REPORTNAME – Report definition name .....	286
SPLFNAME - Based on spooled file .....	286
JOB-Spooled file job .....	287
SPLNBR-Spooled file number .....	287
ADDRPTLIN – Add Report Line .....	287
REPORTNAME – Report definition name .....	288
LINENAME – Report line name .....	288
LINETYPE – Line type .....	288
PAGERANGE – Can occur on page numbers .....	288
LINERANGE – Can occur on line numbers .....	290
RULETYPE – Rule type .....	290
RULEPTY - Rule evaluation priority .....	292
SECTION – Section name .....	292
TEXT – Text ‘description’ .....	293
REPEAT –Depth of repeat group .....	293

LINERULES – Line rules.....	293
CHGRPTLIN – Change Report Line .....	296
CPYRPTLIN – Copy Report Line.....	296
RMVRPTLIN – Remove Report Line .....	296
DSRPTLIN – Display Report Line .....	297
RNMRPTLIN – Rename Report Line.....	297
WRKRPTLIN – Work with Report Lines.....	297
REPORTNAME – Report definition name.....	297
SECTION – Report section name .....	297
ADDRPTITM – Add Report Item.....	297
REPORTNAME – Report definition name.....	297
ITEMNAME – Report item name.....	297
LINENAME – Report line name.....	298
SECTION – Report section name .....	298
CHARPOS – Character position .....	298
CHARLEN – Character length .....	299
ITEMTYPE – Item type .....	299
TEXT – Text ‘description’ .....	299
DATATYPE – Data type.....	299
NULLDTAOPT – Blank data option.....	299
DATFMT – Date format.....	300
DATSEP - Spooled file date separator.....	300
CHGRPTITM – Change Report Item .....	300
CPYRPTITM – Copy Report Item .....	301
RMVRPTITM – Remove Report Item .....	301
DSRPTITM – Display Report Item.....	301
RNMRPTITM – Rename Report Item .....	301
WRKRPTITM – Work with Report Items.....	301
REPORTNAME – Report definition name.....	301
LINENAME – Report line name.....	301
ADDRPTSCT – Add Report Section.....	301
REPORTNAME – Report definition name.....	302
SECTION – Report section name .....	303
PARENT– Parent section name.....	303
STARTLINE - Section start line name.....	303
STARTRULES – Section start rules.....	303
ENDLINE - Section end line name .....	305
ENDRULES – Section end rules .....	306
LINENAMES – Lines included in section .....	306
CHGRPTSCT – Change Report Section .....	306
CPYRPTSCT – Copy Report Section .....	306
RMVRPTSCT – Remove Report Section .....	306
DSRPTSCT – Display Report Section.....	306
RNMRPTSCT – Rename Report Section .....	306
WRKRPTSCT – Work with Report Sections.....	307
REPORTNAME – Report definition name.....	307
PARENT– Parent section name.....	307
Commands related to Report Maps.....	308
CRTRPTXL – Create Report-to-Excel Map .....	308
MAPNAME – Report-to-Excel map name .....	308
REPORTNAME – Report definition name.....	308
DFTUSEAUT - Default use authority.....	308

DFTCHGAUT - Default change authority .....	308
TEXT – Text ‘description’ .....	309
GRPSEQOPT - Row group sequence option.....	309
CHGRPTXL – Change Report-to-Excel map.....	309
CPYRPTXL – Copy Report-to-Excel map.....	309
FROMMAP – From Report-Excel map name .....	310
TOMAP – To Report-Excel map.....	310
DLTRPTXL – Delete Report-to-Excel map .....	310
DSRPTXL – Display Report-to-Excel map .....	310
RNMRPTXL – Rename Report-to-Excel map.....	310
MAPNAME – Report Report-to-Excel map .....	310
NEWMAP – New Report-to-Excel map .....	310
RTVRPTXL – Retrieve Report-to-Excel map.....	310
MAPNAME – Report-to-Excel map .....	310
SRCFILE – Source file .....	310
SRCMBR – Source member .....	311
MBROPT – Source member .....	311
WRKRPTXL – Work with Report-to-Excel maps .....	311
ADDRPTXLR – Add Report-to-Excel Map Row Group.....	311
MAPNAME – Report-to-Excel map name .....	312
ROWGRPNAME – Row group name .....	312
PARENT– Parent row group name .....	312
TEXT – Text ‘description’ .....	314
NEWGRPOPT - New row group option.....	314
SECTION – Report section name .....	314
LINENAME – Report line name.....	315
CHGRPTXLR – Change Report-to-Excel Map Row Group .....	315
CPYRPTXLR – Copy Report-to-Excel Map Row Group .....	315
RMVRPTXLR – Remove Report-to-Excel Map Row Group .....	315
DSRPTXLR – Display Report-to-Excel Map Row Group.....	315
RNMRPTXLR – Rename Report-to-Excel Map Row Group .....	315
WRKRPTXLR – Work with Report-to-Excel Map Row Groups.....	316
MAPNAME – Report-to-Excel map name .....	316
PARENT– Parent row group name .....	316
ADDRPTXLC – Add Report-to-Excel Map Cell.....	316
MAPNAME – Report-to-Excel map name .....	316
ROWGRPNAME – Row group name .....	316
ROWNBR–Row number .....	317
COLUMN –Column letter .....	317
CONTENT – Cell content.....	317
CELLITEM –Report item .....	317
CELLTEXT – Cell text.....	318
MRCCELLS – Merge to cell.....	318
CHGRPTXLC – Change Report-to-Excel Map Cell.....	319
CPYRPTXLC – Copy Report-to-Excel Map Cell .....	319
RMVRPTXLC – Remove Report-to-Excel Map Cell .....	319
DSRPTXLC – Display Report-to-Excel Map Cell.....	319
RNMRPTXLC – Rename Report-to-Excel Map Cell.....	319
WRKRPTXLC – Work with Report-to-Excel Map Cells.....	319
MAPNAME – Report-to-Excel map name .....	319
ROWGRPNAME – Row group name .....	319
ADDRPTXML (Add Report-to-XML Map Element) command.....	320

MAPNAME –Report-to-XML Map Name .....	320
ELEMENT–Element name .....	320
PARENT– Parent element name .....	320
SEQNBR – Sequence number.....	320
ITEMNAME- Report item .....	320
TEXT – Text ‘description’ .....	321
NEWELMOPT - New element option .....	321
SECTION – Report section name .....	322
LINENAME – Report line name.....	322
CHGRPTXMLE – Change Report-to-Excel Map Element .....	322
CPYRPTXMLE – Copy Report-to-Excel Map Element .....	323
RMVRPTXMLE – Remove Report-to-Excel Map Element.....	323
DSRPTXMLE – Display Report-to-Excel Map Element.....	323
RNMRPTXMLE – Rename Report-to-Excel Map Element .....	323
WRKRPTXMLE – Work with Report-to-Excel Map Elements .....	323
MAPNAME – Report-to-Excel map name .....	323
PARENT– Parent element name .....	323
ADDRPTXMLA (Add Report-to-XML Map Attribute) command.....	324
MAPNAME –Report-to-XML Map Name .....	324
ELEMENT–Element name .....	324
ATTRIBUTE–Attribute name .....	324
SEQNBR – Sequence number.....	324
ITEMNAME- Report item .....	324
TEXT – Text ‘description’ .....	325
CHGRPTXMLA – Change Report-to-Excel Map Attribute .....	325
CPYRPTXMLA – Copy Report-to-Excel Map Attribute .....	325
RMVRPTXMLA – Remove Report-to-Excel Map Attribute .....	325
DSRPTXMLA – Display Report-to-Excel Map Attribute.....	325
RNMRPTXMLA – Rename Report-to-Excel Map Attribute .....	326
WRKRPTXMLA – Work with Report-to-Excel Map Attributes.....	326
MAPNAME – Report-to-Excel map name .....	326
ELEMENT–Element name .....	326
CRTRPTXML – Create Report-to-XML Map.....	327
MAPNAME – Report-to- XML map name.....	327
REPORTNAME – Report definition name.....	327
DFTUSEAUT - Default use authority.....	327
DFTCHGAUT - Default change authority .....	327
TEXT – Text ‘description’ .....	328
ELMSEQOPT - Element sequence option .....	328
CHGRPTXML – Change Report-to-XML map .....	328
CPYRPTXML – Copy Report-to-XML map .....	328
FROMMAP – From Report-XML map name .....	328
TOMAP – To Report-XML map .....	329
DLTRPTXML – Delete Report-to-XML map.....	329
DSRPTXML – Display Report-to-XML map.....	329
RNMRPTXML – Rename Report-to-XML map .....	329
MAPNAME – Report Report-to-XML map.....	329
NEWMAP – New Report-to-XML map .....	329
RTVRPTXML – Retrieve Report-to-XML map .....	329
MAPNAME – Report-to-XML map.....	329
SRCFILE – Source file .....	329
SRCMBR – Source member .....	329



MBROPT – Source member .....	330
WRKRPTXML – Work with Report-to-XML maps .....	330
Worked Example: Using report definitions and maps .....	331
SUPER SUN SEEDS – Customer Order Report .....	331
1. Create the Report Definition .....	331
2. Design the Report Definition .....	331
3. Define the different line types .....	332
4. Define the report items for each line .....	343
5. Define the report sections .....	346
6. Create a Report-to-XML Map .....	351
7. Add attributes to Report-to-XML Map Elements .....	354
8. Use a Report-to-XML Map to generate an XML document .....	358
9. Create a Report-to-Excel Map .....	358
10. Add row groups to a Report-to-Excel Map .....	359
11. Add cells Report-to-Excel Row Groups .....	362
12. Use a Report-to-Excel Map to generate an Excel file .....	363
Digital Signatures .....	364
Commands related to Parameter Sets .....	367
CRTPRMSET – Create Parameter Set .....	367
PRMSETNAME – Parameter set name .....	367
CMDUSER – User running command .....	367
SPLF – Spooled file name .....	367
SPLUSER – Spooled user profile .....	368
SPLNBR – Spooled file number in job .....	368
OUTQ – Spooled file output queue .....	369
FORMTYPE – Spooled file form type .....	369
USRDTA – Spooled file user data .....	370
JOB – Spooled file job name .....	370
PGM – Spooled file program name .....	370
PRIORITY – Evaluation priority .....	371
CMD – Command string .....	371
DFTUSEAUT – Default use authority .....	372
DFTCHGAUT – Default change authority .....	372
TEXT – Text ‘description’ .....	373
Examples for CRTPRMSET .....	373
CHGPRMSET – Change Parameter Set .....	374
CPYPRMSET – Copy Parameter Set .....	374
FROMPRMSET – From parameter set name .....	374
TOPRMSET – To parameter set name .....	374
DLTPRMSET – Delete Parameter Set .....	374
DSPPRMSET – Display Parameter Set .....	374
RNMPRMSET – Rename Parameter Set .....	374
PRMSETNAME – Parameter set name .....	374
NEWPRMSET – New parameter set name .....	375
WRKPRMSET – Work with Parameter Set .....	375
CFGPRMSET – Configure Parameter Set .....	375
SAMPLESPLF – Sample spooled file .....	375
PRMSETNAME – Parameter set name .....	376

# **Introduction**

This introduction tells you just a little about what CoolSpools can do for you and will give you a few ideas how you might like to put it to use in your company or organization.

If you are upgrading from a previous version of CoolSpools, please read the “Upgrade Notes” section of this document before switching live applications over to running this version of the software.

If you’re already familiar with earlier versions of CoolSpools, you may like to go straight to the “What’s new in this release?” section of this document.

Many commercial companies and other organizations have a long and successful association with the IBM system i (originally known as the AS/400 and also previously known as the iSeries). System i users typically create a lot of printer output or “spooled files”, whether in terms of reports that are distributed internally or of other documents that are sent to customers or trading partners.

Historically, pretty much the only thing you could do with a spooled file was print it on paper and send it to your users or customers through the mail. Today, in many environments, paper-based distribution of information is no longer acceptable, either because of time, or cost, or simply because your users and customers expect their information to be provided in a more usable format.

This is where CoolSpools comes in. You don’t need to throw away your traditional system i applications which output information in the form of spooled files. You don’t even need to modify those applications in any way, other than to add a simple call to the appropriate CoolSpools commands. CoolSpools will take the spooled files your systems already produce and convert them to a format in which they can be distributed, archived, accessed and published electronically, for example in the form of Adobe PDF files or Excel spreadsheets,

Let's have a look in a bit more detail at CoolSpools can do for you.

## **What CoolSpools can do for you**

### **Automated distribution of reports and documents**

Are you still sending reports out on paper?

Maybe you take orders over the Web but still have to send invoices by snail-mail because the billing system is an old AS/400 package?

Maybe your customers can order a catalogue or buy an insurance policy online, but they still have to wait a day or so for the paperwork to arrive. Wouldn’t it be great to be able to email the catalogue or the policy schedule to your customer within a few minutes of the order having been placed?

PDF and other files created by CoolSpools can be e-mailed to users, colleagues and customers, rather than sent out on paper. Not only does this save you money and streamline your business processes, it also makes a statement about your organization’s commitment to the era of e-commerce.

If security or confidentiality is a concern, PDF files created with CoolSpools can be secured so that they cannot be modified or printed, or if you want them to be really secure, you can password-protect them.

### ***Information sharing***

However many copies of a report you print, it's never enough. There's always someone else who'd like to see it but is in an office on the other side of the country and can't get access to a copy.

Files created by CoolSpools can be stored on a central corporate server, such as your system i or a Windows or UNIX server, and shared amongst your users as a corporate information resource.

Alternatively, you could publish them on your Web site for customers worldwide to see, or on a secure Intranet or Extranet.

### ***Report Enhancement***

CoolSpools doesn't just convert your reports, it enhances them too.

For example, when converting to PDF format, you can add bookmarks that index the pages of your report so your users can find the information they want quickly and easily. When you create a PDF or HTML document from a spooled file, you can add color to improve the presentation as well. Maybe you couldn't afford a system i color printer: now that's no longer an issue since you can simply convert your spooled file to PDF and print your document on an inexpensive PC color printer!

### ***Archiving and offline storage***

The system i provides no built in means of saving and restoring spooled files. Yet for many companies their system i reports are a critical part of their business process, and may not be easy to re-create. Maybe you have to waste large amounts of expensive system i disk just keeping copies of old reports online. Doing this can also impact your system's performance, since jobs which created spooled files remain in the system even when they have ended if they created spooled files which still exist on an output queue,

Files created by CoolSpools can be stored on inexpensive storage media such as PC disk or CD-ROM for easy retrieval. Once CoolSpools has converted your report to a stream file, you can move this file to a PC server, or save it to tape or to CD-ROM, then delete the original spooled file, freeing up precious system i resources.

CoolSpools offers several options for archiving spooled files. You can convert the spooled file to a PDF file, in which case you would use Adobe Acrobat reader to re-print the report; you can convert it to an RTF (Rich Text Format) file, in which case you would use a Word Processor such as MS Word; or you can use the CoolSpools SAVSPLF command to create a stream file in CoolSpools' own stream file archive format, in which case the RSTSPLF command can be used to restore the original spooled file from the stream file archive at a later date.

Both PDF and archive format use data compression to minimize the size of the archived spooled file. PDFs can be created in PDF/A format, which is ISO 19005-1:2005, an international standard for document archiving.

## **Document Formats**

CoolSpools converts system i spooled files to a stream file in one of several different formats. Where the stream file is created, where you will store it permanently, and how you will access it, will depend on a number of factors. Some typical approaches are discussed below in the section "Where did my output go?"

You can choose several different formats for your files, depending on your particular requirements.

### ***Adobe® PDF (Portable Document Format)***

Adobe® Portable Document Format (PDF) is the de facto standard for electronic document distribution. PDF is a universal file format that preserves the fonts and formatting of the source document. PDF files are compact and can be shared, viewed, navigated, and printed exactly as intended by anyone with a copy of the free Adobe Acrobat Reader, which can be downloaded from many places on the Internet, including the Adobe site at <http://www.adobe.com/products/acrobat/readstep.html>.

Adobe PDF is the ideal format for electronic document distribution because it overcomes the problems commonly encountered in electronic file sharing. If you create a document in PDF format and e-mail it to a customer, so long as the recipient has a copy of the Acrobat reader, you can be confident that they will be able to read and print the document and that it will appear to them just as it did to you when you created it.

PDF files can be published and distributed anywhere. You can attach them to e-mail, make them available on a corporate server, an Intranet or Extranet, post them on Web sites or circulate them on CD-ROM.

If you want to use CoolSpools to distribute and archive your system i reports, PDF is the format to choose.

### ***XML***

Starting with Version 6, CoolSpools can convert your system i spooled files to XML. XML is used by a variety of applications, such as EDI.

Conversion to XML requires the use of a [report definition](#) which defines the semantic structure and content of the spooled file.

### ***HTML (Hypertext Mark-up Language)***

HTML is the language in which web pages are written. If you would like to view your reports in a browser, such as Netscape® Navigator or Microsoft® Internet Explorer, CoolSpools can convert your spooled files to HTML format so you can do so.

### ***Excel Format***

If your users would prefer to have their data in the form of a spreadsheet rather than just columns of number on a sheet of paper, CoolSpools can create a native Excel (™) spreadsheet (.xls file) from your spooled file.

Options allow you to exclude unwanted lines from your report as it is converted to Excel format (for example, report headings not required in the spreadsheet).

## ***RTF (Rich Text Format)***

If you want to access your spooled file data in a word processing application, RTF is the format to choose.

RTF (Rich Text Format) is a format understood and handled by most if not all modern WP programs, including MS Word, Lotus WordPro, MS WordPad etc.

## ***Delimited ASCII Text***

CoolSpools can also convert your report to a delimited ASCII text file, such as a CSV (comma-separated variable file) or TSV (tab-separated variable file). This format is ideal for loading reports containing columns of numbers into a spreadsheet, Business Intelligence tool or other application for further manipulation.

CoolSpools can use any field delimiter you like (by default a comma, but also tabs, semicolons, blanks etc.) and any string separator you specify (by default a double quote “), allowing you to generate files in the precise format required by your PC application.

## ***Plain ASCII Text***

CoolSpools can also simply convert your system i spooled file to a basic ASCII text file. This file can then be loaded into virtually any PC application, such as a word processor or spreadsheet. ASCII text versions of your spooled files may also be useful for indexing purposes as part of a document management solution.

## ***TIFF format***

CoolSpools can also create a TIFF (Tagged Image Format File) image from your spooled file. This is suitable for viewing in an imaging application such as Windows Image and Fax Viewer.

## ***Archive format***

You can also save spooled files as stream files in a highly compressed spooled file archive format using the CoolSpools SAVSPLF command. Spooled files saved in this way can be restored from the stream using the CoolSpools RSTSPLF command.

Please note that (depending on the IFS file system selected) the stream files created using this option may still reside on your system i disks, albeit in the IFS rather than as a spooled file. You should use the OS/400 SAV command to back these stream files up to tape, or copy them to CD-ROM or to a PC server, before considering that your spooled files are truly secure. Please note also that these archive files are not viewable in any PC application.

# **Upgrade Notes**

Please read the following notes carefully before upgrading to CoolSpools Spool Converter V6R1M0 from an earlier version of CoolSpools or CoolSpools PLUS.

To determine which version of CoolSpools you are running, check the name of the library in which the CVTSPLSTMF command object you are using resides, e.g.:

**DSPOBJD OBJ(CVTSPLSTMF) OBJTYPE(\*CMD)**

The library name corresponds to the version of CoolSpools as shown in the table below:

<b>Product Library Name</b>	<b>Licensed Program Id</b>	<b>Version</b>
CVTSPLV2R1	2CVTSPL	CoolSpools Version 2
CVTSPLV3R1	3CVTSPL	CoolSpools Version 3
CVTSPLV4R1	4CVTSPL	CoolSpools Version 4
CVTSPLV5R1	5CVTSPL	CoolSpools Version 5
COOLSPV5R1	5COOLSP	CoolSpools PLUS V5R1M0

If your command is in a library other than those shown, you are either running an unsupported, unlicensed version or have moved or copied the command object from its original location. Contact [support@ariadnesoftware.co.uk](mailto:support@ariadnesoftware.co.uk) for assistance.

CoolSpools Spool Converter was previously available as both a standalone module and also as part of the complete CoolSpools PLUS suite. Now, CoolSpools Spool Converter is packaged as a product option (option 1) of CoolSpools V6R1M0.

## **License Keys**

You are entitled to upgrade to V6R1M0 of CoolSpools Spool Converter free of charge if:

- the machine on which you wish to run CoolSpools Spool Converter V6R1M0 has a valid license for an earlier version of CoolSpools or CoolSpools PLUS

**and**

- you are either in your first 12 months' maintenance period after purchase or have paid your latest annual maintenance invoice.

Please note that if your system has multiple logical partitions (LPARs), you must purchase a license for each partition on which you wish to run the software.

If you wish to upgrade, you can simply download the software from [www.ariadnesoftware.co.uk](http://www.ariadnesoftware.co.uk) and install it according to the instructions contained in the "Installation" section of this User Guide. However, if you licensed an earlier version of CoolSpools Spool Converter, you will need to request a license key for the new version. Simply e-mail [support@ariadnesoftware.co.uk](mailto:support@ariadnesoftware.co.uk) and ask for your key for V6R1M0 of CoolSpools Spool Converter. Please quote your system serial number(s) and processor group code(s) in your e-mail. These are shown at the top of the WRKLICINF screen.

Without a license key, CoolSpools Spool Converter V6R1M0 will allow you a 30-day grace period and will then no longer run.

If you have not paid your annual maintenance invoice, and if you need longer than 30 days to test CoolSpools V6R1M0, we will, on request, send you a temporary license key to extend the trial period.

If you require additional temporary license keys to assist with testing CoolSpools V6R1M0, or if you run into any problems during your testing, please do not hesitate to contact us at [support@ariadnesoftware.co.uk](mailto:support@ariadnesoftware.co.uk).

### **Warning/Disclaimer**

**We recommend strongly that all production applications are re-tested thoroughly using the new version in your development environment before you switch over to running the new version in your production environment.**

All CoolSpools versions are packaged as separate licensed programs and install into different libraries. This means that all versions of CoolSpools can coexist and run alongside one another on the same machine. You can switch an application from using one version to using another simply by changing the library list of the job to include the appropriate version library or by specifying a different library name when you run the command. Hence it is quite a simple matter to test your applications using the new version while continuing to run the older version in production.

Please note that while ariadne makes every effort to ensure that CoolSpools functions in the same way with the same parameters from one version to the next, it is not possible to guarantee this. This is why you should re-test your applications against a new version before going live with it as it is possible that in some cases different parameter settings will be necessary to obtain the same results as before.

ariadne software accepts no responsibility for any damage, expense or loss of income incurred as a result of unforeseen and unwanted effects resulting from installing new versions of its software or applying PTFs.

### **Minimum OS/400 Release Level**

The minimum OS/400 release level required to run V6R1M0 of CoolSpools Spool Converter is OS/400 **V5R3M0**.

If you are running V5R2M0 or an earlier version of OS400, you will not be able to install V6R1M0 of CoolSpools Spool Converter.



## **Product Library**

All product options of CoolSpools V6R1M0 install into the single product library **COOLSPV6R1**. This means that you no longer have to manage multiple product libraries for the separate modules that made up CoolSpools PLUS V5R1M0 (Slipstream, Communiqué, CoolSpools etc.)

You will probably need to change library lists in job descriptions and other system objects in order to pick up the new version of the code rather than the old.

This change of library name has the advantage that it allows you to run both V6R1M0 and the earlier versions on the same machine. You are therefore able to test V6R1M0 before swapping your production applications over to the new version, as we strongly advise you to do.

## **Memo to Users**

Please refer to the [Memo to Users](#) for important information about changes you need to take account of before migrating to CoolSpools Spool Converter from an earlier version of CoolSpools or CoolSpools PLUS.

## **Changes between V4 and V5**

This section lists changes affecting users upgrading from Version 4 of CoolSpools or earlier.

Please note that V5 was a major modification from V4 and this list is not exhaustive. You should re-test your applications before going live with a later version and should not rely on checking this list alone.

- **CVTSPLxxxx commands**

In previous releases, the only way to convert a spooled file using CoolSpools was to run the CVTSPLSTMF (Convert Spooled File to Stream File) command.

V5 introduced two alternatives: the Conversion API and the format-specific commands (CVTSPLPDF for PDF, CVTSPLXLS for Excel, CVTSPLRTF for RTF etc.)

While CVTSPLSTMF remained backwards compatible with previous releases, and existing code that runs CVTSPLSTMF would normally produce the same results as before, you cannot simply replace a call to CVTSPLSTMF with a call to the equivalent format-specific command and assume that you will obtain equivalent output.

One of the main reasons for introducing the format-specific commands was to allow parameter formats and defaults to be modified compared with those in CVTSPLSTMF. Whilst this enabled us to enhance the functionality of those parameters, rationalize their behavior or simplify their use, it did of course also mean that running one of the format-specific commands would not necessarily give the same results as CVTSPLSTMF with the same parameters. In some instances you need to modify parameters in order to obtain the same results.



- **Color values**

In Version 4 and earlier, the colors generated in PDF when the various predefined color names were used (e.g. \*GRAY, \*BROWN etc.) were proprietary, i.e. they were set to values selected by ariadne. The RGB color values associated with the new set of color names in V5 was consistent with the industry standard colors adopted for HTML. This means that the same color name in V5 might generate a slightly different color in PDF from that generated in V4. However, the previous color could be generated by means of the new V5 feature which allows the specification of user-defined colors.

### **Changes between V3 and V4**

- **TODIR parameter**

Under V3, a second element of the TODIR parameter allowed you to specify the directory in which work files were created when generating a PDF file. This element was removed in V4, which does not use work files in the same way as V3.

- **PMTADLPARM parameter**

The Prompt Additional Parameters parameter was introduced to control the displaying of many less frequently used options.

### **Changes between V2 and V3**

If you are currently running CoolSpools Version 2 in production, you must read the notes below before upgrading to a higher version.

V3 fixed a number of problems in V2. However, having fixed these problems, we do not and cannot guarantee that the results you obtain with a later version of CoolSpools will be the same as those you obtained with V2, using the same parameters. You may therefore need to modify your parameters to obtain the same results.

These comments apply in particular if you are using any of these features:

- Bookmarks
- Spooled file splitting
- Exit program parameters
- Additional page or overlay margins
- Page size changes

- **Bookmarks, Split keys and Exit Program Parameters**

In the creation of V3, a lot of effort was put into improving the accuracy of CoolSpools text selection features, i.e. the identification of text strings in the spooled file (e.g. for SPLIT(\*KEY) processing) and extraction of text strings from the spooled file (e.g. for bookmark purposes or as exit program parameters). These features worked very well in V2 in relation to traditional row-and-column based spooled files, but perhaps not quite so well when dealing with complex AFP spooled files, especially those using proportional fonts.

With a proportional font, the positioning of a piece of text on the page is dependent not only on the font itself but also on the text content. For example, the string “WWWWWWWWWWW” takes up much more space on the page than the string “iiiiiiiiii”, even though each string contains 10 characters in the same font. Previously CoolSpools did not take account of the text content and estimated the position of text on the page based on the number of characters and the average width of a character in the font being used. In V3 CoolSpools handles proportional fonts much better, and the accuracy with which text string identification and extraction works is much improved.

However, this does have implications for existing applications. If you have production applications which use text functions (bookmarks, splitting or exit program parameters), it is possible that the results you obtain with V3 and later versions will not be identical to those you obtained previously with V2. Although we believe that V3 and later versions will deliver more accurate results, it may be that you are already running CoolSpools live with parameters which produce the results you desire (arrived at perhaps by trial and error). Running V3 and later versions with the same parameters may not give the same results, which could have undesirable effects on live applications.

We strongly recommend therefore that any applications using text string features (bookmarks, splitting or exit program parameters) are re-tested and, if necessary, the parameters readjusted, before you go live with V3.

- **EXITPGMPOS and EXITPGMKEY parameters**

The EXITPGMPOS and EXITPGMKEY parameters were modified in V3 to allow multiple exit program parameters to be defined so that more than one string extracted from the spooled file could be passed as parameters to an exit program.

If you have applications which run V2 of CoolSpools and specify an EXITPGMPOS or EXITPGMKEY parameter, please note that it is likely these will need to be modified to take account of this change before V3 or any later version will run properly. Specifically, an extra set of parentheses is necessary around the elements of the parameter.

For example, if you currently have something like:

**CVTSPLSTMF...EXITPGMPOS(1 2 3 4 \*INCH)**

this needs to be changed to:

**CVTSPLSTMF... EXITPGMPOS((1 2 3 4 \*INCH))**

otherwise your application will report an error when running CVTSPLSTMF.

- **PAGESIZE parameter**

Please note that the operation of the PAGESIZE parameter was changed slightly between V2 and V3 in response to a number of improvements we made in the way CoolSpools handles rotated pages, overlays, images and text.

The third element of this parameter previously allowed you to control whether text rotation was implemented in PDF or not. Text rotation in the spooled file is now always reflected in PDF output. The third element of the PAGESIZE parameter now

controls whether or not rotated pages are viewed in rotated mode, or displayed without rotation for easier viewing.

The fourth element of this parameter previously controlled whether CoolSpools took any account of page rotation. CoolSpools now always implements page rotation where it exists. Now this parameter element allows you to instruct CoolSpools to operate as if the spooled file were being directed to a printer which caused an automatic page rotation to occur.

- **MARGINS parameter**

A new element was added to the MARGINS parameter in V3 which allows you to instruct CoolSpools to shift text outside of an overlay by a distance on the page that you specify. This is necessary because some printers automatically shift text which would otherwise encroach on their non-print borders. This often happens when page rotation is in effect. The result of this automatic shifting is that text is printed on the page perhaps ¼ inch below where it would be expected to print based on the content of the spooled file alone. Since CoolSpools cannot anticipate whether this kind of shift occurs on your particular printer or not, you will need to instruct it to apply the shift using this new parameter element, in order to achieve proper alignment of text on the page.

Previously you may have handled this kind of text misalignment by applying an additional overlay margin. You may obtain better results now using the text shift option.

## **What's new in CoolSpools Version 6?**

Highlights include:

- **Report Definitions and Report Maps**

- Define the structure of reports to be converted by creating *Report Definitions*
- Define the structure of Excel files and XML documents to be generated by creating *Report-to-Excel maps* or *Report-to-XML maps*
- By using a Report-to-Excel map with the new CVTSPLXL (Convert Spooled File to Excel) and CVTSPLDLM (Convert Spooled File to Delimited Text) commands, it is possible to achieve much greater control over the structure and formatting of Excel and delimited files such as CSVs than was previously possible.
- By using a Report-to-XML map with the new CVTSPLXML command, complex XML documents can be generated from a spooled file.

- **XML**

- CoolSpools Spool Converter can now generate XML by means of the new CVTSPLXML command
- A simple schema (or XSD or DTD) can be automatically generated or the XML can be linked to an existing schema
- A simple stylesheet (XSLT or CSS) can be automatically generated or the XML can be linked to an existing stylesheet

- **Excel 2007 .xlsx (Office Open XML) format**

- CVTSPLXLS and the new CVTSPLXL command can now optionally generate Excel 2007 Office Open XML format files (.xlsx)
- BIFF 8 (Excel 97+) .xls files is still the default Excel format
- Support for BIFF 5 (Excel 95) files is now withdrawn.

- **Styles and formatting options**

- When converting to Excel using CVTSPLXL and when converting to XML with CVTSPLXML, named styles can be defined on the new DFNSTYLES parameter.
- Style definitions can also be defined and stored permanently using the WRKSTLDFN, CRTSTLDFN, etc, commands and referred to subsequently on commands such as CVTSPLXLS, CVTSPLXL and CVTSPLXML.
- The styles can be associated with particular types of data (details, headings, titles etc.) or, using the APYSTYLES parameter, with individual fields to control the visual formatting of the output such as font, text color, background color, numeric formatting etc.

- **Conditional formatting**

- When converting to Excel using CVTSPLXLS or CVTSPLXL, you can apply conditional formatting to selected columns or entire rows using the CNDFMTGRP and CNDFMTRULE parameters.
  - For example, you can set the color of rows based on the value of a specified field: red for high values, green for low values etc. or make certain fields bold if rules you specify evaluate to true.
- **Digital Signatures**
    - Authenticate PDFs you produce by using the new option to apply a digital signature as you create them using the new SIGNATURE parameter of the CVTSPLPDF command.
    - Add digital signatures to existing PDFs by means of the new ADDPDFSGN (Add PDF Signature) command.
- **New ways of supplying email addresses to which documents should be sent**
    - When emailing spooled files, it is necessary to tell CoolSpools the email addresses of the people the spooled file should be sent to. This information might vary from one spooled file to the next, or from one section of a spooled file to the next, depending on its content. For example, you might want to split a spooled file that contains a batch of invoices into multiple PDFs, one per invoice, and email each invoice to the appropriate customer. Until now, this has often required the use of a user-written exit program to achieve.
    - Three new methods of supplying email addresses to use are now available in this release
    - The new EMAILTO(\*EMAILFILE) option and related EMAILFILE parameter allow you to tell CoolSpools to look up the emails to be used in a specified file. [CoolSpools variables](#) can be used to extract data from the spooled file at run time to be used as keys to read the file. For example, you might take the customer number from the spooled file and use it to read a customer file to obtain the email address(es) for a particular invoice.
    - The new EMAILTO(\*EMAILSQL) option and related EMAILSQL parameter allow you to tell CoolSpools to look up the email address(es) to be used by running an SQL statement. [CoolSpools variables](#) can be used to extract data from the spooled file at run time and replace parameter markers in the SQL statement. For example, you might take the invoice number from the spooled file and run a piece of SQL to join the invoice file to the customer file to obtain the email address(es) for a particular invoice.
    - The new EMAILTO(\*USRDFNDDTA) option lets you tell CoolSpools that the USRDFNDDTA attribute of the spooled file contains one or more email addresses to be used.
    - See the EMAILTO parameter below for further information.
- **User-defined names for CoolSpools variables**

- Support for [CoolSpools variables](#) was added by PTF to Version 5. Use of the <:EXITPGMPOSn:> and <:EXITPGMKEYn:> CoolSpools variables let you extract items of data from the spooled file at run time and reference them on text parameters that support CoolSpools variables.
- In this release, it is now possible to assign your own names to these variables. For example, the item of data referred to by the first element of the EXITPGMPOS parameter can be referenced as <:EXITPGMPOS1:>. However, if you use the new option to assign it your own name to that item of data, perhaps "Customer\_number", you can also refer to it as <:Customer\_number:>.

- **Parameter sets**

- You can now maintain and use named **parameter sets**.
- A parameter set provides a means of specifying and storing a set of command parameters for CVTSPLPDF and the other CVTSPLxxxx commands so that those command parameters can be retrieved quickly and simply at a later time by using the new RTVPRMSET (Retrieve Parameter Set) command parameter. When you specify a parameter set name on the RTVPRMSET parameter, the command parameter values stored with the parameter set are retrieved and override the default values for the command.
- You can also specify a number of spooled file attributes with each parameter set. When you specify RTVPRMSET(\*SPLF) on the CVTSPLPDF command or another CVTSPLxxxx command, the system will search for a parameter set where the attributes match those of the spooled file being converted. This provides a convenient but powerful and flexible means of defining default conversion parameters for different types of spooled files. For example, you can now configure things using this facility so that the system will use one set of default parameters for one type of spooled file and another set of default parameters for another, saving you the trouble of inputting specific parameter settings every time to convert spooled files that need special parameters.

- **Secure FTP (FTPs)**

- Support for FTP over SSL when using TOSTMF(\*FTP) to send the output to a remote system using FTP

- **Encrypted passwords**

- Where a password can be specified on a command parameter (e.g. FTP connection, zip file), the password can be supplied as an encrypted hex string to avoid the need to hold passwords in plain text form in source code
- DSPENCPWD (Display Encrypted Password) command generates the encrypted form of a password to be used

- **Split to new worksheet**

- When converting to Excel format with CVTSPLXL, you can use the SPLIT() options to split the input spooled file into multiple workbooks and/or multiple worksheets within each workbook. For example, create a new workbook for each region in your report and a new worksheet in the region's workbook for each store in the region.
- New exit points \*SHEETSTR and \*SHEETEND related to this functionality.
- **Conversion to HTML now supports most images and graphics**
  - When converting to HTML format, most types of images from overlays and page segments can now be automatically converted to JPEGs and referenced from within the HTML.
  - Line and box graphics are also now reproduced in HTML.
  - Generated HTML documents can be used as the text of an email message, complete with embedded images and graphics
- **New exit program parameter type \*TYPE4**
  - The new \*TYPE4 option streamlines exit program parameter. All information is passed as a single program parameter in the form of a data structure
  - All future enhancements to exit program parameters will be applied to this format

## **System Requirements**

- A system i or running IBM i (OS/400) V5R3M0 or above.
- 100 Mb of system i disk space.
- **No** PC is required.



## **Installation**

Refer to the [Installation Guide](#) for instructions.

## **Maintenance**

Refer to the [Maintenance Guide](#) for instructions.

# Getting Started with CoolSpools Spool Converter

If you are upgrading from a previous version of CoolSpools, please read the “Upgrade Notes” section of this document before switching live applications over to running this version of the software.

Refer to the “Installation” section for instructions for installing CoolSpools on your system.

The simplest way to get started with CoolSpools is to display the CoolSpools menu by entering:

## **GO COOLSPV6R1/SPOOLCONV**

The menu displays the various conversions that are available with CoolSpools.

When you select an option, you will be prompted to enter the parameters required.

```
-----
SPOOLCONV              CoolSpools - Spool Converter Menu

Select one of the following:

    Convert a spooled file to:

        1. PDF
        2. Excel
        3. HTML
        4. RTF
        5. CSV
        6. Text
        7. TIFF
        8. Spooled file(s)
        9. XML
       10. Excel using a map
       11. Database

       21. Save a spooled file
       22. Restore a spooled file
       23. Merge PDF files
       24. Parameter sets

       31. Report definitions
       32. Report-to-XML maps
       33. Report-to-Excel maps
       34. Report-to-Database maps

Selection or command

==>
```

```
-----
```

The commands that are run by these options are as follows:

### 1. PDF

Prompts the **CVTSPLPDF** (Convert Spooled File to PDF) command.

Converts a system i spooled file to a stream file in PDF (Portable Document Format) format, suitable for viewing with Adobe's free Acrobat Viewer application.

### 2. Excel

Prompts the **CVTSPLXLS** (Convert Spooled File to Excel command.

Converts a system i spooled file to a stream file in native Excel format (.xls, BIFF5 or BIFF 8), suitable for opening in Microsoft Excel or another spreadsheet program that supports Excel files.

### 3. HTML

Prompts the **CVTSPLHTML** (Convert Spooled File to HTML) command.

Converts a system i spooled file to a stream file in HTML format, suitable for viewing in a browser such as Microsoft Internet Explorer, Opera or Netscape Navigator.

### 4. RTF (Rich Text Format)

Prompts the **CVTSPLRTF** (Convert Spooled File to RTF) command.

Converts a system i spooled file to a stream file in RTF (Rich Text Format) format, suitable for opening in a word processor application such as Microsoft Word.

### 5. CSV (Comma Separated Variable)

Prompts the **CVTSPLCSV** (Convert Spooled File to CSV) command.

Converts a system i spooled file to a CSV (Comma Separated Variable) or similar delimited ASCII text file. The delimiter does not have to be a comma; you can use any other character you wish, for example a tab, semicolon or pipe (|).

### 6. Text

Prompts the **CVTSPLTXT** (Convert Spooled File to Text) command.

Converts a system i spooled file to a flat ASCII text file.

### 7. TIFF

Prompts the **CVTSPLTIFF** (Convert Spooled File to TIFF) command.

Converts a system i spooled file to a TIFF image file.

### 21. Save Spooled File

Prompts the **SAVSPLF** (Save Spooled File) command.

Saves a spooled file as a stream file in ariadne's highly compressed proprietary spooled file archive format. The spooled file can be restored from this stream file subsequently using option 12 or the **RSTSPLF** or **CVTSTMSPLF** commands.

### 22. Restore Spooled File

Prompts the **RSTSPLF** (Restore Spooled File) command.

Restores a spooled file previously saved using the **SAVSPLF** or **CVTSPLSAV** commands or **CVTSPLSTMF** with the **TOFMT(\*SAV)** option,

### **23. Merge PDF files**

Prompts the MRGPDF (Merge PDF) command.

Merges (combines) two or more PDF files.

### **24. Parameter sets**

Lets you work with parameter sets. A parameter set is a predefined group of command parameters that can be retrieved by name to save you having to re-type the same parameters every time you convert a particular spooled file.

### **31. Report definitions**

Lets you work with report definitions. See [Report definitions and Report Maps](#) below.

### **33. Report-to-Excel maps**

Lets you work with Report-to-Excel maps definitions. See [Report definitions and Report Maps](#) below.

### **32. Report-to-XML maps**

Lets you work with Report-to-XML maps definitions. See [Report definitions and Report Maps](#) below.

### **34. Report-to-Database maps**

Lets you work with report-to-Database maps definitions. See [Report definitions and Report Maps](#) below.

## ***Using styles***

You can define styles that will be applied to your output when you are converting to Excel or XML formats. These styles control the appearance of data on screen when the spreadsheet is opened (in MS Excel or another spreadsheet application) or when the XML document is opened (in your browser).

There are two ways to define styles:

- Permanently, by means of the WRKSTLDFN (Work with Style Definitions) and CRTSTLDFN (Create Style Definition) commands. Styles defined in this way are stored for future reference by name on APYSTYLES parameter of CVTSPLXLS, CVTSPLXL and CVTSPLXML, as well as the CNDFMTRULE parameter of CVTSPLXLS and CVTSPLXL.
- Temporarily, using the DFNSTYLES parameter of the CVTSPLXL and CVTSPLXML commands. Styles defined in this way exist only for the duration of this running of the command and become undefined after the command completes. If the name of a style defined on the DFNSTYLES parameter is the same as that of an existing style definition created using CRTSTLDFN, the attributes defined on the DFNSTYLES parameter override those of the permanent style definition for the duration of the current running of the command. Note that this option is not supported with CVTSPLXLS which does not have a DFNSTYLES parameter.

There is a single predefined style named \*NORMAL, which corresponds to the Normal style in Excel and defines the default styling for cells in your spreadsheet. The default styling for cells in your Excel spreadsheet or XML stylesheet is thus determined using the following hierarchy:

- If a style named \*NORMAL is specified on the DFNSTYLES parameter, the attributes specified there will apply.
- If it is not defined on DFNSTYLES, but a style named \*NORMAL has been created using WRKSTLDFN or CRTSTLDFN, the attributes specified there will apply.
- Otherwise the system-supplied defaults from the table shown under the DFNSTYLES parameter below apply.

There are two ways in which to associate a style with a piece of data in your output file (e.g. a cell in an Excel spreadsheet or an element of an XML document):

### 1. Implicitly

By defining the style name the same as the name of a row group in your Database-to-Excel map or an element in your Database-to-XML map, you implicitly apply that style to the data in question. For example, a style called REPORT\_HEADING will be implicitly and automatically applied to an Excel row group called REPORT\_HEADING.

***Note that style names are case-sensitive because XML element names need to be case-sensitive and for this association of names to work, the names must match exactly in terms of case.***

### 2. Explicitly

Alternatively, use the APYSTYLES parameter to define the styles you wish to apply to different parts of the file you create.

**Example:**

**CVTSPLXL**

...

```
DFNSTYLES((HIGHLIGHT *YES *NO *GENERAL *NONE *BOTTOM *NO *NO
*AUTOFIT *ARIAL 12 *YES *NO *NO *YELLOW *BLUE *AUTO *NONE *THIN))
APYSTYLES((TOTALS *ANY *ANY HIGHLIGHT))
```

This code defines a new style called HIGHLIGHT that uses Arial bold 12-point yellow on blue and applies that style to the row group called TOTALS.

### **Using conditional formatting**

Styles are also used when you want to apply conditional formatting rules to Excel spreadsheets that CoolSpools Spool Converter generates. Conditional formatting lets you modify the appearance of cells in the spreadsheet depending on whether certain rules you define are met or not. For example, if your spreadsheet contains data from customer accounts, you might color those rows that relate to accounts with a negative balance red to highlight them, while those with a credit balance over \$1,000 might be colored green.

Use the CNDFMTGRP (Conditional Formatting Groups) parameter to define the range of columns to which a group of related rules should be applied.

Use the CNDFMTRULE (Conditional Formatting Rules) parameter to define the rules to be applied and the style that will be used to format cells where those rules evaluate to true.

## Using encrypted passwords

In the past, if you specified a password on a command such as CVTSPLSTMF and embedded that command in your CL source code, you would need to store that password in plain text form. This was clearly a security exposure.

Now, CoolSpools Spool Converter gives you the opportunity to use encrypted passwords on all command parameters that accept a password string. An encrypted password is a scrambled version of your password which is returned to you when you supply the actual password to the DSPENCPWD (Display Encrypted Password) command. You can then code the scrambled password in your source code and specify \*YES for the associated “**Encrypted password supplied**” element to indicate to CoolSpools Spool Converter that it needs to decrypt the password before use.

For example, if you supply the password “test” to DSPENCPWD, thus:

**DSPENCPWD PWD('test')**

it send you the completion message:

**Encrypted password is X'178D2D35E0EBFF508A63252433D6C4E0'.**

You can then use this encrypted password on commands that require a password, e.g.:

**ZIPDTA ... PWD(X'178D2D35E0EBFF508A63252433D6C4E0' \*YES)**

The password of the zipped file(s) will be “test”.

## Using parameter sets

A parameter set is a set of command parameters which you can manage and reference using a parameter set name.

For example, when you convert a particular spooled file to PDF, you might want to include one or more images using the INCLFILE parameter of the CVTSPLPDF command so that the PDF includes a forms overlay replacing the preprinted stationery on which the spooled file used to be printed.

However, typing the INCLFILE parameter every time can be laborious and error-prone. You can avoid the need to do that by using a parameter set. With a parameter set, you define a particular command string just once - for example, the INCLFILE parameter and any other special parameters needed to convert a given spooled file or files - and assign a name to that set of parameters. You can then retrieve and use that named parameter set just by specifying it on the RTVPRMSET (Retrieve Parameter Set) parameter of the CVTSPLPDF command and other commands that support this function.

Moreover, when you define a parameter set, you can specify the command to which it applies and other criteria such as the user profile of the user running the command and a number of spooled file attributes. If you specify RTVPRMSET(\*SPLF) on the CVTSPLPDF command, the system will look for a match against these criteria and use the first parameter set where the criteria correspond. This provides a powerful but convenient means of defining default conversion parameters for different types of spooled file.

See the [CRTPRMSET](#) (Create Parameter Set) command and the RTVPRMSET (Retrieve Parameter Set) parameter below for further details and examples.

# **Where Did My Output Go?**

Each of CoolSpools' CVTSPLxxxx commands converts a system i spooled file to a stream file in a format such as PDF, Excel or RTF. Where the output is created depends on what you specify on the TOSTMF parameter of the CVTSPLxxxx command that you ran. You have a number of options which we will discuss shortly.

Normally you will want to access these stream files from a PC application such as Adobe Acrobat Viewer, Microsoft Excel or Microsoft Word. How you access CoolSpools output from your PC depends on a number of factors which we will also consider now.

## **The TOSTMF parameter**

When you run one of the CVTSPLxxxx commands, you specify where you want the output to go and what you want it to be called on the **TOSTMF (To Stream File)** parameter.

There are 3 basic options:

- **IFS path name**

You can define an absolute or relative IFS path specifying the name of the file to be created and the directory in which it will be placed.

The IFS is a collection of file systems provided by your system i. Depending on which file system you select, your output may be stored locally on your system i disks or remotely on another system on your network, which could be a PC, another system i a UNIX server etc.

Use of the IFS is explained more fully below.

The special value **\*FROMFILE** (the parameter default value) tells CoolSpools to create a file name from the name of the spooled file and an appropriate extension based on the format of the file being created (e.g. .pdf for a PDF file, .xls for an Excel file etc.) and place it in the current directory of the job.

- **\*FTP**

This tells CoolSpools to send the output using FTP (File Transfer Protocol) to another system running an FTP server process. This could be another system i, a PC server, a UNIX machine etc.

- **\*EXITPGM**

This indicates that you will specify the location at a later stage in an exit program that will be called while CoolSpools is running.

## **Understanding IFS path names**

The IFS (Integrated File System) is a collection of file systems that your system i can use to store and retrieve information. Depending on which file system you choose to use, the data may be stored locally (on your system i' own disks) or remotely (on another system in your network).



When you enter a path name on the TOSTMF parameter, you are telling CoolSpools the name of the file you wish to create. You will also be telling it, explicitly or implicitly, in which file system and directory to save that file.

The path consists of four elements:

- **The Extension**

If you type a name that ends with a period (.) and then a sequence of characters, you have specified an **extension**.

For example: **.pdf, .xls, .rtf**

Windows and other operating systems may use this extension to determine what type of file you have created. For example, if you double-click in Windows on a file name ending in **.pdf**, it is likely that Windows will start or switch to Adobe Acrobat Reader and open the file.

This makes it very important that you should choose an extension which is appropriate to the type of file you are creating.

For example, if you are using CVTSPLPDF to create a PDF file, specify a file name ending in **.pdf** so Windows recognizes that the file should be opened with Adobe Acrobat Reader, but if you are using CVTSPLXLS to create an Excel file, choose a file name ending in **.xls** to ensure that Windows recognizes the file as an Excel spreadsheet.

- **The File Name**

The part of the path name that precedes the extension is the name of the file itself. CoolSpools does not impose any restrictions other than the limit of 1,024 bytes for the entire path name.

Please note, however, that the syntax and rules that apply to the name will be dependent on the file system you choose. For example, the QDLS file system ("shared folders") does not allow the file name to be longer than 8 characters with an optional extension of 1-3 characters (old DOS-style 8.3 naming). Also note file names in some file systems are case-insensitive (e.g. root file system) while file names in other file systems are case-sensitive (e.g. QOpenSys).

- **The Directory Path**

You can optionally specify a directory or list of sub-directories in which the file is to be saved.

For example, if you have a directory called sales with subdirectories for each region, and then subdirectories for each year and month, you may need to specify a path such as:

**sales/north/2010/nov**

to indicate that the directory in which you wish to save your file is the November subdirectory within the 2005 subdirectory of the north region's subdirectory within **sales**.

- **The File System**

You can optionally specify a file system name at the beginning of the path to indicate to which file system the path refers.

Here is a list of commonly used file system names that can be used at the beginning of a path name. Note that each begins with a / (forward slash) and that the root file system is indicated by a single forward slash alone:

<b>/</b>	The “root” file system. This is the “default” system i hierarchical file system.
<b>/QDLS</b>	Document Library Services (“shared folders”)
<b>/QNTC</b>	Windows file system. This file system provides access to data and objects that are stored on a server running Windows NT 4.0 or higher.  Although this includes access to the data on a Windows NT Server that is running on an IXA (Integrated xSeries Adapter, previously known as the Integrated Netfinity Server, Integrated PC Server or FSIO), it is <b>NOT</b> restricted to the IXA.  <b>This file system can be used to directly read data from and write data to a separate Windows server on your network.</b>
<b>/QOpenSys</b>	A hierarchical file system compatible with UNIX and POSIX. Uses case-sensitive names.
<b>/QSYS.LIB</b>	The system i database. Although it is possible to save CoolSpools output in a database file member, this is not recommended as the data is unlikely to be easily accessed there.

You should also understand the difference between an **absolute** path name and a **relative** path name.

An absolute path name is one which explicitly defines the full location at which a file is to be saved.

For example, the path name

**/sales/north/2010/nov/new\_business.pdf**

is an absolute path name which specifies the full location of a file to be created and breaks down as follows:

<b>/</b>	The initial / indicates the root file system
<b>sales</b>	The name of the directory in the root file system
<b>north</b>	The name of a subdirectory within <b>/sales</b>
<b>2010</b>	The name of a subdirectory within <b>/sales/north</b>
<b>nov</b>	The name of a subdirectory within <b>/sales/north/2010</b>
<b>new_business</b>	The name of the file to be created
<b>.pdf</b>	The file extension, indicating an Adobe Acrobat file.

However, if you do not enter a forward slash (/) at the beginning of a path name, your system i will interpret this as a **relative** path name. Relative path names are interpreted relative to the current directory of the job (similar to the current directory in Windows or DOS).

For example, if your current directory is already set to **/sales**, the path

**north/2010/nov/new\_business.pdf**

(note there is no leading /) would be interpreted relative to **/sales** and would refer to exactly the same location as the absolute path

**/sales/north/2010/nov/new\_business.pdf**

The current directory of your job can be set with the **CHGCURDIR** or **CD** commands. Often, the current directory will be set automatically for you when you sign on to the system i based upon the HOMEDIR (home directory) attribute of your user profile.

Assume that your user profile has HOMEDIR = **/home/john**, indicating that when you sign on the current directory should be set to the **john** subdirectory within the **home** directory of the root file system. Unless you have changed this with CHGCURDIR or CD, if you specify a relative path name, the path will be interpreted relative to your current directory **/home/john**.

For example, the relative path

**reports/sales.pdf**

would be interpreted as referring to a file called **sales.pdf** in a subdirectory called reports within **/home/john**.

You will need to enclose path names in single quotes (') on the TOSTMF parameter if they contain forward slashes or other special characters.

For example:

**TOSTMF(new\_business.pdf)**

is acceptable to OS/400 without single quotes, but your system i will insist that:

**TOSTMF('/sales/north/2010/nov/new\_business.pdf')**

is entered with single quotes around the path name. When prompting the command with F4, the system i will enclose the path name in quotes for you if you have not already done it.

Further information on the IFS can be found at:

[publib.boulder.ibm.com/iserics/v5r1/ic2924/info/ifs/rzaaxmst.pdf](http://publib.boulder.ibm.com/iserics/v5r1/ic2924/info/ifs/rzaaxmst.pdf)

## **Choosing where to store your output**

When it comes to deciding where to save your CoolSpools output, a number of factors need to be considered, for example:

- **Simplicity**

How easy is it to save files to and retrieve files from a particular IFS file system? Are the naming rules for the file system complex or restrictive?

- **Performance**

How well does that file system perform? Is saving and retrieving data from that file system quick and efficient or slow and laborious?

- **Reliability**

Will the file system always be available or is there a chance that it might be unavailable for some reason at the time when you try to save data to it or retrieve data from it?

- **Access**

What choices do you have with regards to accessing the data? How easy is it to retrieve data from the file system you choose to use using an appropriate application? For example, how easy is it to open a PDF file in Acrobat from a PC?

- **Management**

How easy is it to perform management functions on the files in the file system, such as backup, archiving and purging of old documents?

- **Security**

Can you ensure that only the right people have access to the documents?

- **Scalability**

Will problems occur when volumes increase?

We will now consider the various IFS file systems you are most likely to want to use according to these criteria.

### **Root File System**

The “root” file system is in many ways the “default” IFS file system and is probably where most CoolSpools users choose to store their output.

You save a CoolSpools file in the root file system if you enter a path name on the TOSTMF parameter which does not explicitly and implicitly refer to any other file system.

Users can access files created on your system i in the “root” file system using network drives, just as they would a share on a Windows server. For example, if your users have their I: drive assigned to the system i root file system, they could open a file called sales\_report.pdf saved in a directory called sales by opening i:/sales/sales\_report.pdf in Adobe Acrobat.

Simplicity	Excellent. The simplest and easiest to use. Long file names are supported. Not case-sensitive.
Performance	Good. Writing data locally will keep down the time taken to create the files. Speed of retrieval from a PC will depend on your network and other factors such as the power and loading of your system i.
Reliability	Excellent. Writing data locally means that file creation is not dependent on the availability of the network or another system.
Access	Good. Easy to access from Windows using network drives.
Management	Good. Can be backed up with the system i. Can be managed from the system i command line or from Windows using a network drive.
Security	Excellent. System i security applies.
Scalability	Moderate. High cost of system i disks a possible issue.
Comments	Recommended unless other factors dictate otherwise

## QDLS File System

The QDLS or “shared folders” file system implements a DOS-style method of saving PC files and other documents on the system i own disks. It is really a legacy file system providing backwards compatibility for older applications written for the S/38 or versions of OS/400 that pre-date the availability of the IFS (OS/400 V3R1M0).

You save a CoolSpools file in the QDLS file system if you enter a path name on the TOSTMF parameter which starts /QDLS or if you use a relative path name and your current directory path starts /QDLS.

Users can access files created on your system i in the QNTC file system using network drives. For example, if your users have their I: drive assigned to the system i root file system, they could open a file called REPORT.PDF saved in a shared folder called SALES by opening i:/QDLS/SALES/REPORT.PDF in Adobe Acrobat.

Simplicity	Good. Familiar to long-standing users of S/38 and AS/400 applications. Not case-sensitive. Naming limited to DOS-style 8.3 conventions so long file names will cause errors.
Performance	Poor compared to the “root” file system.
Reliability	Excellent. Writing data locally means that file creation is not dependent on the availability of the network or another system.
Access	Good. Easy to access from Windows using network drives.
Management	Good. Can be backed up with the system i. Can be managed from the system i command line or from Windows using a network drive.
Security	Excellent. System i security applies.
Scalability	Moderate. High cost of system i disks a possible issue.
Comments	Use the “root” file system instead.

## QNTC File System

The QNTC file system is the system i implementation of Windows network neighborhood. It allows you to write to and read from files stored on a Windows server running NT 4.0 or above. This is **not** restricted to the IXA (Integrated xSeries Adapter, previously known as the Integrated Netfinity Server, Integrated PC Server or FSIOP)

Please note that you will need OS/400 V5R2M0 or above to read and write to files stored under Windows XP.

You save a CoolSpools file in the QNTC file system if you enter a path name on the TOSTMF parameter which starts /QNTC or if you use a relative path name and your current directory path starts /QNTC. The file system name /QNTC should be followed by the name of the server, then the name of the shared resource on that server (e.g. the shared directory name) and then the path within that shared directory.

Imagine you have a Windows server which is known to the network as **server1**. On that server there is a directory called **sales** which is shared under the name **sales**. Within that shared directory there is a subdirectory called **2010**. If you have QNTC configured and your security settings allow it, you can save a file called november.pdf in that subdirectory from the system i by specifying the path name:

**/QNTC/server1/sales/2010/november.pdf**

The QNTC file system can be quite difficult to configure and manage, but once you have it running it can provide a very effective means of creating CoolSpools output directly on a Windows server in your network.

Please note in particular that the system i user profile of the job which accesses QNTC must be the same name and have the same password as a user id that Windows networking recognizes.

Further information on QNTC is at:

<http://publib.boulder.ibm.com/systemi/v5r2/ic2924/index.htm?info/ifs/rzaaxmstqntcfs.htm>

<http://www-1.ibm.com/support/docview.wss?uid=na1aea450153eebf8ff8625670f0072550f&rs=110>

<http://www.itjungle.com/fhg/fhg031704-story04.html>

<http://www.itjungle.com/mgo/mgo111903-story02.html>

Once you have saved your files on a Windows server in your network, users can then access files created with CoolSpools on that Windows server using Windows networking. For example, if they have their F: drive assigned to a directory called sales on that server, they could access a file called sales\_report.pdf in that directory simply by opening file F:/sales\_report.pdf.

Simplicity	Can be difficult to set up and manage. Once files are saved on the Windows server, access should be very simple.
Performance	Creating files across the network on the PC server may be slow. Retrieval of files once created should be very fast but will depend on the server and network loading.
Reliability	Creating files across the network on the PC server requires both the server and the network to be available at the time.
Access	Easy to access from Windows using Windows networking.
Management	Good. Will need to be backed up with your Windows server.
Security	Good. Windows security applies.
Scalability	Excellent. Low-cost PC disks can be used.
Comments	If you prefer to store your files on a Windows PC server rather than on the system i, this is an ideal solution if the initial setup issues can be overcome and you can ensure that the PC server will be available to the system i when it needs to create the files.

### **Typical Solutions**

When implementing CoolSpools, it is important to make the right choices about where you will save the files you create and how you will access them.

Here are a few typical approaches that users have successfully implemented in the past.

- **Save the files in the system i “root”**

This is a really simple, easy and reliable method.



To save a file in the “root” file system, you just specify a path name starting with a forward slash / and without any special file system identifier (i.e. not /QDLS, /QNTC etc.).

You can open files saved in the root file system from your PC applications (Acrobat, Excel, Word etc.) by using network drives to open the file just as you would a file saved locally on your PC or on a Windows or UNIX server.

The only real downside of this approach is that the files occupy space on your system i disks, which can be expensive compared to PC disks.

See the section on NetServer below for details of how to make files stored on your system i available for access from a PC using a network drive.

- **Save the files directly to a Windows server using QNTC**

As explained above, the QNTC file system allows you to write directly to a Windows server from your system i.

Once QNTC is configured, you can use CoolSpools to create your files on a suitable Windows server by specifying a path name starting /QNTC on the TOSTMF parameter of the CoolSpools command you are running.

Once your files are saved on your Windows server, they can be accessed by any authorized user who can connect to that server.

- **Save the files directly to a Windows or UNIX server using FTP**

As an alternative to using the QNTC file system, if your Windows server is running the FTP service, you can use the CoolSpools TOSTMF(\*FTP) option to send the output to that server via FTP.

Once your files are saved on your Windows server, they can be accessed by any authorized user who can connect to that server.

This method can also be used to send the output to a UNIX server.

- **Email**

In the past you may have produced a large number of system i spooled files which were printed then distributed on paper through your internal or external mail.

This process can be transformed into an automated, low-cost electronic service by creating PDFs, RTFs or Excel files from your spooled files rather than printing them on paper.

If you have installed CoolSpools Email (CoolSpools product option 2), or if you have some other method of sending email from your system i, you can then distribute them electronically by email. The stream files could then be deleted once they had been emailed if they were no longer required.

## **NetServer**

In order to access files stored on the system i from a PC network drive, you must have NetServer running on your system i and you must have created an appropriate NetServer file share.

NetServer can be managed from a PC using System i Navigator (part of System i Access). However, System i Navigator can be slow and heavy on resource usage, so many customers find our FREE NetServer Toolkit (CoolSpools Product Option 5) a simpler and more convenient way to administer NetServer.

A NetServer file share is very similar to a shared directory on a Windows server, in that it makes a system i IFS directory available to access on the network. Users can assign a network drive under Windows by specifying a directory path such as:

```
\\systemi_name\share_name
```

where "systemi\_name" is the name of the system i as known to NetServer (usually the system name prefixed by a Q, but modifiable using Ops Nav or the NetServer Toolkit CHGNETSVRA (Change NetServer Attributes) command;

or

```
\\systemi_IP_address\share_name
```

where "systemi\_IP\_address" is the IP address of the system i

"share\_name" in both cases is the name of the share you created

If using NetServer Toolkit, you can create a file share with the CRTFILSHR command.



# CoolSpools Commands

- **CVTSPLPDF** (Convert Spooled File to PDF)  
Converts spooled files to Portable Document Format files that can be viewed in Adobe Acrobat Reader.
- **CVTSPLXML** (Convert Spooled File to XML)  
Converts spooled files to XML documents. Requires the use of a Report-to-XML map to specify the structure of the input spooled file and the output required.
- **CVTSPLXL** (Convert Spooled File to Excel)  
Converts spooled files to Excel spreadsheets in .xls (BIFF8) or .xlsx (Excel 2007) format that can be opened in Microsoft Excel or another spreadsheet application that supports Excel files. Requires the use of a Report-to-Excel map to specify the structure of the input spooled file and the output required.
- **CVTSPLDLM** (Convert Spooled File to Delimited Text)  
Converts spooled files to CSV (Comma Separated Variable) or a similar delimited text file format. Requires the use of a Report-to-Excel map to specify the structure of the input spooled file and the output required.
- **CVTSPLXLS** (Convert Spooled File to Excel)  
Converts spooled files to Excel spreadsheets in .xls (BIFF8) or .xlsx (Excel 2007) format that can be opened in Microsoft Excel or another spreadsheet application that supports Excel files.
- **CVTSPLCSV** (Convert Spooled File to CSV)  
Converts spooled files to CSV (Comma Separated Variable) or a similar delimited text file format.
- **CVTSPLRTF** (Convert Spooled File to RTF)  
Converts spooled files to Rich Text Format (RTF) files that can be opened in Microsoft Word or another word processing application that supports RTF.
- **CVTSPLHTML** (Convert Spooled File to HTML)  
Converts spooled files to HTML files that can be viewed in a browser such as Microsoft Internet Explorer.
- **CVTSPLTXT** (Convert Spooled File to Text)  
Converts spooled files to flat text files (normally ASCII).
- **CVTSPLTIFF** (Convert Spooled File to TIFF)  
Converts spooled files to a TIFF image file.
- **CVTSPLSPLF** (Convert Spooled File to Spooled File)  
Converts spooled files to one or more other spooled files. Pages may be excluded from the original spooled file or the spooled file may be split into multiple spooled files.
- **MRGPDF** (Merge PDF)

Lets you merge two or more PDF files into a single PDF file.

- **RTVSPLDTA** (Retrieve Spooled File Data)

Saves the raw spooled file data stream as a stream file. This function may be useful in conjunction with applications which can process a printer data stream, for example IBM AFP Viewer.

- **RTVPCLRSC** (Retrieve PCL Resources)

Allows resources such as soft fonts and macros held in \*USERASCII spooled files containing PCL data to be retrieved and saved for later use. When other PCL spooled files are later converted, if they refer to the resources in question, the resources saved earlier can be retrieved and included in the conversion process.

- **SAVSPLF** (Save Spooled File)

Saves a spooled file as a compressed stream file from which it can be restored using RSTSPLF.

- **RSTSPLF** (Restore Spooled File)

Restores spooled files from stream files which were created by SAVSPLF.

## CoolSpools Variables

Certain parameters listed below support the use of CoolSpools variables.

CoolSpools variables consist of a pre-defined variable name from the list below enclosed in<: ... :> (start of variable marker = left-hand angle bracket followed by a colon, end of variable marker = colon followed by a right-hand angle bracket).

You can define a different pair of markers from <: and :> by adding/changing the environment variables **CS\_VAR\_LEFT\_MARKER** and **CS\_VAR\_RIGHT\_MARKER**. For example, if you have CS\_VAR\_LEFT\_MARKER set to \$% and CS\_VAR\_RIGHT\_MARKER set to %\$, you would use \$%PAGSETNBR%\$ rather than <:PAGSETNBR:> etc.

These variable names, including the markers, are replaced at run time by the corresponding data value. Variable names are not case-sensitive.

CoolSpools Version 6 introduces support for assigning user-defined names to the data items returned by the <:EXITPGMPOSn:> and <:EXITPGMKEYn:> CoolSpools variables, allowing you to refer to those data items using more meaningful and memorable names. In this release, it is now possible to assign your own names to these variables. For example, the item of data referred to by the first element of the EXITPGMPOS parameter can be referenced as <:EXITPGMPOS1:>. However, if you use the new option to assign your own name to that item of data, perhaps "Customer\_number", you can also refer to it as <:Customer\_number:>.

Variable name	Description
<:PAGSETNBR:>	Page Set Number. This is a sequential number identifying the page set. A page set is a set of pages which will be output to a separate stream file when splitting is occurring.
<:STRPAGNBR:>	Starting page number. The first page number from the spooled file in the stream file being created.
<:ENDPAGNBR:>	Ending page number. The last page number from the spooled file in the stream file being created.
<:EXITPGMPOSn:> where n is a number from 1 to 99	<p>The value of the exit program user-defined parameter selected by the nth element of the EXITPGMPOS command parameter.</p> <p>If you wish to select text items from the spooled file and use them as variables, but do not wish to call any exit programs, specify EXITPGM(*VAR).</p> <p>You can assign as name of your own choosing to this item of data by means of the "Variable name" element of the EXITPGMPOS parameter. For example, if you were to specify the variable name "Invoice_number" on the first element of EXITPGMPOS, the CoolSpools variables &lt;:EXITPGMPOS1:&gt; and &lt;:Invoice_number:&gt; could then be used interchangeably.</p>
<:EXITPGMKEYn:>	The value of the exit program user-defined parameter selected by the nth element of the EXITPGMKEY command

where n is a number from 1 to 99	<p>parameter.</p> <p>If you wish to select text items from the spooled file and use them as variables, but do not wish to call any exit programs, specify EXITPGM(*VAR).</p> <p>You can assign as name of your own choosing to this item of data by means of the “Variable name” element of the EXITPGMKEY parameter. For example, if you were to specify the variable name “Customer_number” on the first element of EXITPGMKEY, the CoolSpools variables &lt;:EXITPGMKEY1:&gt; and &lt;:Customer_number:&gt; could then be used interchangeably.</p>
<:CURJOB:>	Current job name
<:CURUSER:>	Current user id
<:CURJOBNNBR:>	Current job number
<:CURDATE:>	The current date in the format of the current job (DATFMT attribute).
<:CURDATE*xxx:>	<p>The current date in the format indicated by *xxx, where *xxx is any one of:</p> <p>*YMD, *MDY, *DMY, *YYMD, *MDYY, *DMYY, *CYMD, *CMDY, *CDMY, *ISO, *EUR, *JIS, *JUL, *LONGJUL, *JOB or *SYSVAL.</p>
<:CURDAY:>	The current day of the month as a number 01-31.
<:CURMONTH:>	The current month as a number 01-12.
<:CURYEAR:>	The current year as a number 0001-9999
<:CURYEAR4:>	The current year as a number 0001-9999
<:CURYEAR3:>	The current year as a number c01-c99 where c is 0 for the 20th century and 1 for the 21st.
<:CURYEAR2:>	The current year as a number 01-99.
<:CURTIME:>	The current time in hhmmss format.
<:FROMFILE:>	Spooled file name
<:SPLNNBR:>	Spooled file number
<:SPLJOB:>	Spooled file job name
<:SPLUSER:>	Spooled file user name
<:SPLJOBNNBR:>	Spooled file job number
<:STMFEXT:>	File extension corresponding to the format being output (e.g. ‘.PDF’ when PDF being generated or ‘.XLS’ when an Excel file is being created).
<:TOFMT:>	To-format. The format of the data being generated (corresponding to the TOFMT parameter of the CVTSPLSTMF command), .e.g. ‘*PDF’, ‘*XLS’
<:SPLDATE:>	The date the spooled file was created (opened) in the format of the current job (DATFMT attribute).

<:SPLDATE*xxx:>	The date the spooled file was created (opened) in the format indicated by *xxx, where *xxx is any one of: *YMD, *MDY, *DMY, *YYMD, *MDYY, *DMYY, *CYMD, *CMDY, *CDMY, *ISO, *EUR, *JIS, *JUL, *LONGJUL, *JOB or *SYSVAL.
<:SPLDAY:>	The day the spooled file was created (opened) as a number 01-31.
<:SPLMONTH:>	The month the spooled file was created (opened) as a number 01-12.
<:SPLYEAR:>	The year the spooled file was created (opened) as a number 0001-9999
<:SPLYEAR4:>	The year the spooled file was created (opened) as a number 0001-9999
<:SPLYEAR3:>	The year the spooled file was created (opened) as a number c01-c99 where c is 0 for the 20th century and 1 for the 21st.
<:SPLYEAR2:>	The year the spooled file was created (opened) as a number 01-99.
<:SPLTIME:>	The time the spooled file was created (opened) in hhmmss format.
<:SPLUSRDTA:>	The user data attribute of the spooled file.
<:SPLUSRDFNDTA:>	The user-defined data attribute of the spooled file.
<:OWNUSER:>	The user profile that owns the spooled file.
<:SPLUSEREMAIL:>	The email address of the spooled file user (user part of spooled file job details). The email address is the SMTP email address of the user from the system directory.
<:SPLUSERNAME:>	The name of the spooled file user (user part of spooled file job details). The name is derived from the information held for the user in the system directory.
<:SPLUSERHOME:>	The home directory of the spooled file user (user part of spooled file job details). The home directory is taken from the HOMEDIR attribute of the user profile.
<:SPLOUTQ:>	Spooled file output queue
<:SPLOUTQLIB:>	Spooled file output queue library
<:CURUSEREMAIL:>	The email address of current user. The email address is the SMTP email address of the user from the system directory.
<:CURUSERNAME:>	The name of the current user. The name is derived from the information held for the user in the system directory.
<:CURUSERHOME:>	The home directory of the current user. The home directory is taken from the HOMEDIR attribute of the user profile.
<:OWNUSEREMAIL:>	The email address of the user profile that owns the spooled file. The email address is the SMTP email address of the user from the system directory.

<:OWNUSERNAME:>	The name of the user profile that owns the spooled file. The name is derived from the information held for the user in the system directory.
<:OWNUSERHOME:>	The home directory of the user profile that owns the spooled file. The home directory is taken from the HOMEDIR attribute of the user profile.

**Example:**

```
CVTSPLPDF
FROMFILE(QSYSPRT)
TOSTMF('<:fromfile:>_<:spljob:>_<:spluser:>_<:spljobnbr:>_<:splnbr:>.pdf')
```

Here the CVTSPLPDF command is being applied to a spooled file called **QSYSPRT**. The name of the stream file to be generated will be derived from various spooled file attributes to give a unique name such as:

QSYSPRT\_INVOICES\_QSYSOPR\_123456\_2.pdf

**Example:**

```
CVTSPLPDF
FROMFILE(QSYSPRT)
EXITPGM(*VAR)
EXITPGMPRM(*POS)
EXITPGMPOS((1 7 10 40))
EMAIL(*YES)
EMAILTO(('<:exitpgmpos1:>'))
```

Here the CVTSPLPDF command is being applied to another spooled file called **QSYSPRT**. No exit programs are to be called, but exit program parameters are defined for the purposes of using them as variables (EXITPGM(\*VAR)). The text item on page 1, line 7, column 10 for 40 characters is extracted and used as the email address to which the spooled file should be sent.

**Example:**

```
CVTSPLPDF
FROMFILE(QSYSPRT)
EXITPGM(*VAR)
EXITPGMPRM(*POS)
EXITPGMPOS((1 7 10 40 email_address))
EMAIL(*YES)
EMAILTO(('<:email_address:>'))
```

This example is exactly the same as the previous one, except that the item of data extracted from the spooled file using the EXITPGMPOS parameter is given the name "email\_address" and can then be referenced using this name on the EMAILTO parameter.

**Example:**

```
CVTSPLPDF  
FROMFILE(QSYSPRT)  
EMAIL(*YES)  
EMAILTO(*EMAILSQL)  
EMAILSQL('select email from INVMST, CSTMST where INVMST.CUSTNO =  
CSTMST.CUSTNO and INVMST.INVNO = ?'('<:invoice_number:>'))  
SPLIT(*POS)  
SPLITPOS((5 9 7))  
EXITPGM(*VAR)  
EXITPGMPRM(*POS)  
EXITPGMPOS((1 5 9 7 INVOICE_NUMBER))
```

Here, a spooled file called QSYSPRT (which contains a batch of invoices) is being split into multiple PDFs every time the value on line 5 position 9 for 7 characters (the invoice number) changes. The EXITPGM(\*VAR) EXITPGMPRM(\*POS) and EXITPGMPOS parameters are being used to extract that same item of data from page 1 of each split file and assigned the name "INVOICE\_NUMBER". That value is then supplied as a variable to the SQL statement defined on the EMAILSQL parameter and used to select the email address or addresses to which the PDF should be sent.

## CoolSpools Functions

When using CoolSpools variables, you can also use a number of CoolSpools functions to adjust the data substituted at run time for each variable. These functions can often be helpful in converting the data returned by a variable to a consistent, standard format. For example, you might want to use CoolSpools variables to build the names of the PDF files you're creating from data held inside the spooled file. CoolSpools functions can help with this, for example by allowing you to:

- remove any leading or trailing spaces
- pad numeric value to a constant fixed length with leading zeros
- translate certain characters which would be invalid in a file name (such as / ) to an alternative acceptable character (such as -)

By default, CoolSpools functions consist of a pre-defined function name from the list below preceded by the marker **\$\$** but you can define a different marker from **\$\$** by adding/changing the environment variable **CS\_FCN\_MARKER**. For example, if you have CS\_FCN\_MARKER set to **%%**, you would use **%%TRIM**, **%%PADL** etc. rather than **\$\$TRIM**, **\$\$PADL** etc.

Function parameters are enclosed in parentheses ( ) and separated by commas. Character values used as parameters are case-sensitive and can be either enclosed in single quotes ' ' (doubled up where required by OS/400), double quotes " ", or not enclosed by anything.

Function names are not case-sensitive.

Function name		\$\$TRIM
Description		Trim characters from the left and right sides of the data. Similar to the ILE RPG %trim() builtin function.
Parameters		
1	Data to trim (typically a CoolSpools variable).	
2	Characters to remove (optional, default = blank).	
Examples		
\$\$TRIM(<:EXITPGMPOS1:>)		Trims blanks from the start of the value returned by CoolSpools variable <:EXITPGMPOS1:>. For example, the value “ 000123.45- “ becomes “ 000123.45-“
\$\$TRIM(<:EXITPGMPOS1:>,'0')		Trims zeros from the start of the value returned by CoolSpools variable <:EXITPGMPOS1:>. For example, the value “000123.4500 “ becomes “123.45 “.



<b>Function name</b>		<b>\$\$TRIML</b>
<b>Description</b>		Trim characters from the left (start) of the data. Similar to the ILE RPG %triml() builtin function.
<b>Parameters</b>		
<b>1</b>	Data to trim (typically a CoolSpools variable).	
<b>2</b>	Characters to remove (optional, default = blank).	
<b>Examples</b>		
\$\$TRIML(<:EXITPGMPOS1:>)		Trims blanks from the start of the value returned by CoolSpools variable <:EXITPGMPOS1:>.  For example, the value “ 000123.45- “ becomes “000123.45- “
\$\$TRIML(<:EXITPGMPOS1:>,'0')		Trims zeros from the start of the value returned by CoolSpools variable <:EXITPGMPOS1:>.  For example, the value “000123.4500 “ becomes “000123.45 “.

<b>Function name</b>		<b>\$\$TRIMR</b>
<b>Description</b>		Trim characters from the right (end) of the data. Similar to the ILE RPG %trimr() builtin function.
<b>Parameters</b>		
<b>1</b>	Data to trim (typically a CoolSpools variable).	
<b>2</b>	Characters to remove (optional, default = blank).	
<b>Examples</b>		
\$\$TRIMR(<:EXITPGMPOS1:>)		Trims blanks from the end of the value returned by CoolSpools variable <:EXITPGMPOS1:>.  For example, the value “ 000123.45- “ becomes “ 000123.45-“
\$\$TRIMR(<:EXITPGMPOS1:>,'0')		Trims zeros from the end of the value returned by CoolSpools variable <:EXITPGMPOS1:>.  For example, the value “000123.4500 “

	becomes "000123.45 ".
--	-----------------------

<b>Function name</b>		<b>\$\$PADL</b>
<b>Description</b>		Pad a string to a given length by adding a specified character at the start.
<b>Parameters</b>		
<b>1</b>	Data to pad (typically a CoolSpools variable).	
<b>2</b>	Length to pad to	
<b>3</b>	Characters to pad with (optional, default = blank).	
<b>Examples</b>		
\$\$PADL(<:EXITPGMPOS1:>,10)		Pads the value returned by CoolSpools variable <:EXITPGMPOS1:> to a length of 10 characters by adding blanks at the start.  For example, the value “123.45- “ becomes “123.45-“
\$\$PADL(<:EXITPGMPOS1:>,10,'0')		Pads the value returned by CoolSpools variable <:EXITPGMPOS1:> to a length of 10 characters by adding zeros at the start.  For example, the value “123.45- “ becomes “0000123.45-“

<b>Function name</b>		<b>\$\$PADR</b>
<b>Description</b>		Pad a string to a given length by adding a specified character at the end.
<b>Parameters</b>		
<b>1</b>	Data to pad (typically a CoolSpools variable).	
<b>2</b>	Length to pad to	
<b>3</b>	Characters to pad with (optional, default = blank).	
<b>Examples</b>		
\$\$PADL(<:EXITPGMPOS1:>,10)		Pads the value returned by CoolSpools variable <:EXITPGMPOS1:> to a length of 10 characters by adding blanks at the end.  For example, the value “123.45-“ becomes

	"123.45- "
\$\$PADL(<:EXITPGMPOS1:>,10,'0')	Pads the value returned by CoolSpools variable <:EXITPGMPOS1:> to a length of 10 characters by adding zeros at the end.  For example, the value "123.45" becomes "123.450000 "

Function name	\$\$REPLACE		
Description	Replaces each occurrence of a string in the data with a replacement string. Similar to a combination of the ILE RPG %scan and %replace functions.		
Parameters			
1	String to replace		
2	String to replace the string specified in the previous parameter with. May be an empty string if you wish to delete the string specified in the previous parameter.		
3	Data to translate (typically a CoolSpools variable).		
Examples			
\$\$REPLACE("/", "",<:EXITPGMPOS1:>)		Replaces each occurrence of the character / with a null string (i.e. removes all / characters) e.g. 01/05/2011 becomes 01052011.	
\$\$REPLACE("Facture","Invoice",<:EXITPGMPOS1:>)		Replaces each occurrence of the string "Facture" with "Invoice".	

<b>Function name</b>		<b>\$\$SUBST</b>
<b>Description</b>		Returns a substring. Similar to ILE RPG's %subst.
<b>Parameters</b>		
<b>1</b>	Data to substring (typically a CoolSpools variable).	
<b>2</b>	Start position	
<b>3</b>	Length (optional, default = to end of string).	
<b>Examples</b>		

\$\$SUBST(<:EXITPGMPOS1:>,5)	Returns the substring of the value returned by CoolSpools variable <:EXITPGMPOS1:> starting at character position 5 and extending to the end of the string.  For example, the value "0000123456" becomes "123456".
\$\$SUBST(<:EXITPGMPOS1:>,5,3)	Returns the substring of the value returned by CoolSpools variable <:EXITPGMPOS1:> starting at character position 5 and extending for 3 characters.  For example, the value "0000123456" becomes "123".

<b>Function name</b>	<b>\$\$UPPER</b>
<b>Description</b>	Converts a string to upper case, assuming the CCSID of the job.
<b>Parameters</b>	
<b>1</b>	Data to convert (typically a CoolSpools variable).
<b>Examples</b>	
\$\$UPPER(<:EXITPGMPOS1:>)	Converts the value returned by CoolSpools variable <:EXITPGMPOS1:> to upper case.  For example, the value "John Smith" becomes "JOHN SMITH".

<b>Function name</b>	<b>\$\$XLATE</b>
<b>Description</b>	Translates characters in the data. Similar to the ILE RPG %xlate function.
<b>Parameters</b>	
<b>1</b>	List of characters to translate from
<b>2</b>	List of characters to translate to
<b>3</b>	Data to translate (typically a CoolSpools variable).
<b>4</b>	Start position (optional, default = first)
<b>Examples</b>	
\$\$XLATE(" ", "<:EXITPGMPOS1:>", 1)	Translates spaces in the value returned by

	<p>CoolSpools variable &lt;:EXITPGMPOS1:&gt; to underscores, starting at the first character.</p> <p>For example, the value "John Smith" becomes "John_Smith".</p>
--	--

<b>Function name</b>	<b>\$\$FIXNAME</b>
<b>Description</b>	<p>Creates a valid name usable in a path name.</p> <p>Converts characters that are invalid or unwise in a path name to acceptable characters. This can be useful, for example, if you are creating a directory or file name from some piece of information extracted from the report (e.g. a date or a name) and that data might contain characters that are invalid in a path name (such as a date separator / or an apostrophe in the name).</p> <p>Spaces are converted to underscores. Other invalid characters are converted to hyphens.</p> <p>The CCSID of the job is assumed.</p>
<b>Parameters</b>	
<b>1</b>	String to convert to a valid name
<b>Examples</b>	
\$\$FIXNAME(<:EXITPGMPOS1:>)	<p>If the &lt;:EXITPGMPOS1:&gt; variable returns the value "John O'Brien"</p> <p>\$\$FIXNAME(&lt;:EXITPGMPOS1:&gt;) converts this to "John_O-Brien".</p>

<b>Function name</b>	<b>\$\$FIXSHEET</b>
<b>Description</b>	<p>Creates a valid Excel sheet name.</p> <p>Converts characters that are invalid in an Excel sheet name to blanks. This can be useful, for example, if you are creating a sheet name from some piece of information extracted from the report (e.g. a name) and that data might contain characters that are invalid in an Excel sheet name (such as / \ [ ] * : and ?).</p> <p>The length of the name is truncated to the maximum length of 31 characters if longer than 31 characters.</p> <p>The CCSID of the job is assumed.</p>
<b>Parameters</b>	

<b>1</b>	String to convert to a valid Excel sheet name
<b>Examples</b>	
\$\$FIXSHEET(<:CUSTOMER_NAME:>)	<p>If the &lt;:CUSTOMER_NAME:&gt; variable returns the value <b>SMITH/JONES TRADING [EUROPE] CORPORATION</b></p> <p>\$\$FIXSHEET(&lt;:CUSTOMER_NAME:&gt;) converts this to <b>SMITH JONES TRADING EUROPE CO</b></p>

<b>Function name</b>	<b>\$\$XLDATE</b>
<b>Description</b>	<p>Converts a date to an Excel date (day count since 1st Jan 1900).</p> <p>This function can be useful when specifying conditional formatting rules that require a date constant to be specified, as Excel requires these to be defined in Excel date format.</p>
<b>Parameters</b>	
<b>1</b>	The date to convert, specified as date string e.g. 07/04/2011
<b>2</b>	<p>The format in which the first parameter is specified.</p> <p>If this parameter is omitted, the date format and separator implied by job attributes DATFMT and DATSEP are assumed.</p> <p>Other valid formats are similar to those supported by ILE RPG, e.g.: *ISO, *USA, *EUR, *JIS, *YMD, *MDY, *DMY.</p> <p>Optionally, a separator character may be appended to the format code, e.g. *MDY/, *DMY, *YMD0 etc.</p>
<b>Examples</b>	
\$\$XLDATE(07/04/2011)	<p>This would return the date (either 7th April or July 4th, depending on whether the DATFMT attribute is *DMY or *MDY) as a day count since 1900.</p> <p>This expression could be used as part of a conditional formatting rule, for example to highlight in red any dates equal to or after 1st January 2011:</p> <p>CNDFMTRULE((1 1 *FLDNAM date_fld *GE '\$\$XLDATE(01/01/2011)' *NONE red)</p>
\$\$XLDATE(07/04/2011,*MDY/)	Same as the above, but with the date format explicitly stated to be MDY with a separator of /

## Functions used with report line rules

Function name		<b>\$\$PATTERN</b>	
Description		Used only when defining rules for report lines (ADDRPTLIN command etc.). Specifies a pattern string against which a piece of text is tested.	
Parameters			
1	Pattern string. See below for details of how to specify a pattern string.		
<i>Pattern Symbol</i>	<i>Denotes</i>	<i>Corresponding regular expression element</i>	<i>Comments</i>
<i>.</i> (period)	<i>Any character including space</i>	<i>.</i>	
<i>X</i>	<i>Any character except space</i>	<i>[^ ]</i>	
<i>A</i>	<i>Any character except space or a digit (0-9)</i>	<i>[^ 0-9]</i>	
<i>#</i>	<i>A digit (0-9), thousands separator, currency symbol, minus sign or space</i>	<i>[0-9,\$- ]</i>	<i>The thousands separator and currency symbol are those associated with the report definition.</i>  <i>Useful for referring to areas of the page which contain edited numeric values.</i>
<i>9</i>	<i>A digit (0-9)</i>	<i>[0-9]</i>	<i>Useful for referring to areas of the page that contain unedited numbers. Use # instead of 9 if the number is edited (has zero suppression, thousands separators, minus signs or currency symbols).</i>
<i>\X</i>	<i>An X</i>	<i>[X]</i>	<i>Where a character is used as a pattern symbol, precede that character by a backslash \ to denote</i>
<i>\.</i>	<i>A period</i>	<i>[\.]</i>	
<i>\A</i>	<i>An A</i>	<i>[A]</i>	

<b>\#</b>	<i>A hash</i>	<b>[ \# ]</b>	<i>the actual character not the pattern symbol.</i>
<b>\9</b>	<i>A 9</i>	<b>[ 9 ]</b>	
<i>Any other</i>	<i>The character specified</i>	<b>[ char ]</b>	<i>Any other character just denotes that character itself.</i>
<b>Examples</b>			
<b>\$\$\$PATTERN(XXX 999)</b>		Three non-space characters followed by 2 spaces and then 3 numeric digits.	
<b>\$\$\$PATTERN(Totals for: 999999)</b>		The string "Totals for:" followed by a space and then 6 numeric digits.	
<b>\$\$\$PATTERN(999999 #9/99/99 #####\.</b> <b>#####\.</b> <b>99)</b>		Six numeric digits followed by 6 spaces then a date (allowing for zero suppression on the first digit) then 4 spaces then an edited number. Note the use of # rather than 9 to allow for zero suppression and the presence of thousands separators and currency symbols. Also note the backslash before the period to indicate that the pattern is checking for an actual period at that position.	

Function name	<b>\$\$REGEX</b>		
Description	Used only when defining rules for report lines (ADDRPTLIN command etc.). Specifies a regular expression string against which a piece of text is tested.  For a tutorial on how to use regular expressions, see <a href="http://www.regular-expressions.info">http://www.regular-expressions.info</a>		
<b>Parameters</b>			
1	Regular expression string. See below for details of how to specify a regular expression.		
<b>Examples</b>			
\$\$REGEX (19 20)\d\d[- /.](0[1-9] 1[012])[- /.](0[1-9] 1[12][0-9] 3[01])		Specifies a regular expression which tests for a date in YYYY-MM-DD format between 1900-01-01 and 2099-12-31 using any of – space / or . as the date separator character.	



## **Command Parameters**

The following pages explain the purpose and use of the various command parameters. The basic parameters are considered first, then additional parameters which are less frequently used.

In the examples, an ellipsis (...) indicates that a number of required parameters have been omitted for the sake of clarity.

### **Basic Parameters**

#### **FROMFILE – From spooled file name**

Parameter	<b>FROMFILE</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, RTVPCLRSC, RTVSPLDTA, SAVSPLF</b>
Dependent on:	<b>None</b>

Specifies the name of the spooled file to be processed.

Please note that a spooled file name is only unique in conjunction with the job details (name, user, number) of the job that created it and its spooled file number within that job. If more than one spooled file of the name specified exists in the job specified, the default value \*ONLY for the SPLNBR (Spooled File Number) parameter is no longer value and the actual spooled file number of the spooled file you wish to convert must be specified on the SPLNBR parameter (or \*LAST if you wish to convert the most recent spooled file of the name given).

The following special values are available:

#### **\*SELECT**

You will be prompted to select the spooled file to be processed from a list of spooled files.

#### **\*SLT**

This should be used only in the context of a CoolSpools command specified on the **CMD** parameter of the CoolSpools Spool Admin **WRKSPLFPDM** or **RUNSPLFCMD** command. It indicates that the spooled file to be processed is that currently selected by WRKSPLFPDM or RUNSPLFCMD when it is processing a set of selected spooled files.

#### **Example:**

**CVTSPLPDF FROMFILE(QSYSPRT)...**

Here the CVTSPLPDF command is being applied to a spooled file called **QSYSPRT**.

## ***TOSTMF - To stream file name or \*FTP***

Parameter	<b>TOSTMF</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, RTVSPLDTA, SAVSPLF</b>
Dependent on:	<b>None</b>
Variables	<b>Allows the use of CoolSpools variables</b>

The **TOSTMF** (To Stream File) parameter specifies the name of the stream file you wish to create and, optionally, the full path where the file should be saved.

Refer to “[Understanding IFS Path Names](#)” above for a discussion of how to specify the path name where the file should be saved. Further information on path names is also available at <http://publib.boulder.ibm.com/system/i/v5r2/ic2924/info/rbam6/rbam6pathnames.htm>.

All directories in the path name must exist. New directories are not created. If the stream file does not exist, it is created.

Note that the equivalent parameter on the CVTSPLSPLF command is **TOFILE**.

Special values:

### **\*FROMFILE**

CoolSpools constructs a name for you based on the name of the original spooled file (FROMFILE parameter) and an extension appropriate to the format of the file being created (see table below).

The file is saved in the current directory of the job running the command.

### **\*FTP**

CoolSpools will send the file to an FTP server. You will specify the additional information needed to connect to the FTP server and save the file on the [FTP](#) parameter, rather than here.

### **\*EXITPGM**

The name of the file to be created will be specified at run time by an exit program by adding a structure of type CS\_STM01 to the option structure list. Refer to the CoolSpools Programmers Guide for additional information.

### **\*VIEW**

(CVTSPLPDF only)

CoolSpools will invoke Adobe Reader on the PC and display the PDF file it creates, but will not save it permanently.

See below for a discussion of the prerequisites required for this option.

**\*PRINT**

(CVTSPLPDF only)

CoolSpools will invoke Adobe Reader on the PC and print the PDF file it creates, but will not save it permanently.

See below for a discussion of the prerequisites required for this option.

A path name you enter here may be up to 128 characters long if you are using CVTSPLSTMF or 1024 bytes if you are using one of the other commands. If you prompt the command using F4 and need additional space in which to type the file name, enter an ampersand (&) and OS/400 will expand the field for you.

Note that the name that you choose must be a valid name for the IFS file system into which the stream file is to be created. For example, the shared folders (QDLS) file system only supports file names in the 8.3 format, i.e. a file name up to 8 characters long followed by an optional extension of up to 3 characters. If you choose an invalid file name, an error will occur and the file will not be saved.

You should choose a file name which is suitable for the type of file being created. For example, PDF files should be given the extension **.pdf** so that they are recognized as PDF files by applications such as Adobe ® Acrobat.

Recommended extensions for use with the different commands and file formats are shown in the table below. Those values in **bold** are the ones used by CoolSpools to create a default file name when TOSTMF(\*FROMFILE) is specified.

Command	Recommended file extension
CVTSPLPDF	<b>.PDF</b> or .pdf
CVTSPLXL, CVTSPLXLS	<b>.XLS</b> or .xls if EXCEL(*XLS) <b>.XLSX</b> or .xlsx if EXCEL(*XLSX)
CVTSPLXML	<b>.XML</b> or .xml
CVTSPLHTML	<b>.HTM</b> , .HTML, .htm or .html
CVTSPLRTF	<b>.RTF</b> or .rtf
CVTSPLTIFF	<b>.TIF</b> , .TIFF, .tif or .tiff
CVTSPLTXT	<b>.TXT</b> , .DAT, .txt or .dat
CVTSPLCSV, CVTSPLDLM	<b>.CSV</b> or .csv if comma-separated .TSV or .tsv if tab-separated

SAVSPLF	<b>.SAV</b> or .sav
RTVSPLDTA	<b>.PRN</b> , .prn .AFP or .afp if AFPDS

**Example:**

```
CVTSPLPDF      FROMFILE(QSYSPRT)
                TOSTMF(/invoices/sales.pdf)
                TODIR(*TOSTMF)...
```

Here the CVTSPLPDF command is being used to convert a spooled file called **QSYSPRT**. The PDF file will be called **sales.pdf** and will be placed in the **invoices** directory of the “root” file system.

**Prerequisites for CVTSPLPDF ... TOSTMF(\*VIEW) or TOSTMF(\*PRINT) options**

These options are implemented using the System i Access STRPCO and STRPCCMD commands.

CoolSpools will generate a unique, temporary name for the stream file and this file will be automatically deleted in the event of an error or when the file has been successfully viewed or printed. The file is not permanently retained.

Viewing or printing of the file is accomplished by invoking Adobe Reader and telling it to open the temporary file. This is dependent on the following criteria being met:

- The job in which the command is run must be an interactive job running IBM System i Access 5250 emulation
- The user must be authorized to the commands STRPCO and STRPCCMD.
- The job must be able to determine the path to the Adobe Reader program on the PC. This can be accomplished in either of two ways:
  1. Setting an environment variable called CS\_ADOBE\_READER\_PATH to the required path, e.g.:

```
ADDENVVAR
  ENVVAR(CS_ADOBE_READER_PATH)
  VALUE('c:\Program Files\Adobe\Reader
  9.0\Reader\AcroRd32.exe') LEVEL(*JOB)
```

2. Using the option key \*ADOBEPATH on the Miscellaneous options parameter to specify the required path, e.g.:

```
CVTSPLPDF ..
  OPTIONS(*ADOBEPATH 'c:\Program Files\Adobe\Reader
  9.0\Reader\AcroRd32.exe')
```

- The job must be able to determine the folder path which it must pass to Adobe Reader in order to allow Adobe Reader to locate the PDF file, which is stored in the /tmp directory on the system i. Thus the path specified must be to a NetServer share which allows the PC to access the system i /tmp directory.

This can be accomplished in two ways:

1. Setting an environment variable called CS\_IFS\_TMP\_PATH to the required path, e.g.:

**ADDENVVAR**

```
ENVVAR(CS_IFS_TMP_PATH)  
VALUE("\\192.168.0.1\root\tmp")  
LEVEL(*JOB)
```

where "root" is a share for the root of the IFS "/" and 192.168.0.1 is the IP address of the system i

2. Using the option key \*TMPPATH on the Miscellaneous options parameter to specify the required path, e.g.:

**CVTSPLPDF ...**

```
OPTIONS(*TMPPATH '\\192.168.0.1\root\tmp'))
```

- In addition, for the TOSTMF(\*PRINT) option, if the PC printer you wish to print to is not the default printer, the name of the printer must be specified on the CS\_PDF\_PRINT\_PRINTERNAME environment variable.

## ***JOB - Job name***

Parameter	<b>JOB</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, RTVPCLRSC, RTVSPLDTA, SAVSPLF</b>
Dependent on:	<b>None</b>

The JOB parameter specifies the name of the job that created the spooled file you wish to convert. This parameter is required in order to identify precisely the spooled file on which you wish to operate.

You can use one of the following special values for this parameter:

<b>*</b> <b>-</b>	The job that created the spooled file was the current job, i.e. the job in which the command is running.
<b>*SBMJOB</b>	The job that created the spooled file was the job that submitted the job in which the command is running.
<b>*SLT</b>	The special value *SLT should only be used in the context of a CoolSpools command specified on the CMD parameter of the CoolSpools Spool Admin WRKSPLFPDM or RUNSPLFCMD commands. It indicates that the spooled file to be processed is that currently selected by WRKSPLFPDM or RUNSPLFCMD when it is processing a set of selected spooled files.

Alternatively, you may specify a fully qualified job name, consisting of:

<b><u>job-name</u></b>	Specify the name of the job that created the spooled file.
<b>user-name</b>	Specify the user name that identifies the user profile under which the job is run.
<b>job-number</b>	Specify the system-assigned job number.

You can determine which job created a spooled file by using the CoolSpools Spool Admin WRKSPLFPDM (Work with Spooled Files PDM-style) command or the OS/400 WRKSPLF (Work with Spooled Files) command (you will need to press F11 twice to view the job details).

**Example:**

```
CVTSPLRTF      FROMFILE(QSYSPRT)  
                TOSTMF(sales.rtf)  
                JOB(*)
```

Here the CVTSPLRTF command is being applied to a spooled file called **QSYSPRT** in order to create a stream file called **sales.rtf** in the current directory of the job. The spooled file was created by the current job.

**Example:**

```
CVTSPLXLS      FROMFILE(INVOICES)  
                TOSTMF(invoices.xls)  
                JOB(123456/QSYSOPR/INVOICES)
```

Here the CVTSPLXLS command is being applied to a spooled file called **INVOICES** in order to create a stream file called **invoices.xls**. The spooled file was created by a job called **INVOICES**, run by the System Operator **QSYSOPR**, with job number **123456**.

## ***SPLNBR - Spooled file number***

Parameter	<b>SPLNBR</b>
Applies to commands:	CVTSPLCSV, CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, RTVPCLRSC, RTVSPLDTA, SAVSPLF
Dependent on:	None

The **SPLNBR** (Spooled File Number) parameter specifies the number of the spooled file which you wish to convert.

You can use one of the special values:

<b><u>*ONLY</u></b>	The job created only one spooled file of the name specified on the FROMFILE parameter. An error will occur if there is more than one spooled file of the name specified in the job.
<b>*LAST</b>	The spooled file to be converted is the latest spooled file of that name created by the job.
<b>*SLT</b>	The special value *SLT should only be used in the context of a CoolSpools command specified on the CMD parameter of the CoolSpools Spool Admin WRKSPLFPDM or RUNSPLFCMD commands. It indicates that the spooled file to be processed is that currently selected by WRKSPLFPDM or RUNSPLFCMD when it is processing a set of selected spooled files.

Alternatively, specify the actual spooled file number of the spooled file that you wish to convert. You can determine the spooled file number by using the CoolSpools Spool Admin WRKSPLFPDM (Work with Spooled Files PDM-style) or OS/400 WRKSPLF (Work with Spooled Files) command to display the spooled file and pressing F11 twice to view spooled file number.

### ***Example:***

```
CVTSPLPDF      FROMFILE(INVOICES)  
                  TOSTMF(invoices.pdf)  
                  JOB(123456/QSYSOPR/INVOICES)  
                  SPLNBR(3)
```

Here the CVTSPLPDF command is being applied to a spooled file called **INVOICES** in order to create a stream file called **invoices.pdf**. The spooled file was created by a job called **INVOICES**, run by the System Operator **QSYSOPR**, with job number **123456**. The spooled file to be converted was the third spooled file opened by the job.



## STMFOPT – Stream file option

Parameter	STMFOPT
Applies to commands:	CVTSPLCSV, CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, RTVPCLRSC, RTVSPLDTA, SAVSPLF
Dependent on:	None

The **STMFOPT** (Stream File Option) parameter allows you to select the action the command should take if the stream file you have specified on the **TOSTMF** and **TODIR** parameters already exists or the naming convention that will be adopted in order to avoid clashes of stream file name.

The options are:

- \*NONE** If the file specified on the TOSTMF parameter already exists, the command reports an error and the existing file is not changed. For safety's sake, this is the default value.
- \*REPLACE** If the file specified on the TOSTMF parameter already exists, it is replaced.
- \*ADD** If the file specified on the TOSTMF parameter already exists, the output from the command is appended to the existing file.  
  
This option is not only supported by the following file formats: HTML, TIFF, RTF, spool archive (SAVSPLF).  
  
When used with Excel output, STMFOPT(\*ADD) adds one or more new worksheets to an existing Excel file.
- \*UNIQUE** CoolSpools avoids clashes of file names by generating a unique file name for the output file(s) by appending a numeric suffix to the body of the name specified on the TOSTMF parameter (i.e. the part of the name prior to the extension). The numeric suffix will be one higher than the highest suffix associated with any existing file of this name in the directory.  
  
You can optionally define a separator character on the third element of the **SPLIT** parameter which will be inserted between the body of the file name and the numeric suffix generated in order to keep file names unique.
- \*EXITPGM** This option indicates that the stream file option will be defined at run time by an exit program. The exit

program must generate an option structure of type CS\_STM01.

**\*RPLXLSSHT**

(CVTSPLXL command only). The new worksheet(s) written to the existing Excel file will replace one or more existing worksheets, specified on the RPLXLSSHT parameter.

**Example:**

**CVTSPLTXT      FROMFILE(INVOICES)...  
STMFOPT(\*ADD)**

The INVOICES spooled file is converted to ASCII text form and the contents of the spooled file will be appended to the end of the existing file.

**Example:**

**CVTSPLPDF      FROMFILE(SALES)  
TOSTMF('/reports/sales.pdf')  
STMFOPT(\*UNIQUE)  
SPLIT(\*POS \*BEFORE \*UNDERSCORE)**

If the reports directory already contains files called sales\_1.pdf, sales\_2.pdf and sales\_3.pdf, the next file created by CoolSpools as a result of this call will be sales\_4.pdf.

## Additional Parameters

### **APYSTYLES – Apply styles**

Parameter	<b>APYSTYLES</b>
Description	<b>Lets you override the style of individual fields, cells or sections of the output file and set field-level attributes.</b>
Applies to commands:	<b>CVTSPLXL, CVTSPLXLS, CVTSPLHTML, CVTSPLXML, CVTSPLCSV, CVTSPLTXT</b>
Dependent on:	<b>None</b>
Supports CoolSpools Spool Converter variables	<b>No</b>

The APYSTYLES parameter allows you to specify the styling to be applied to parts of the output file or to specify field-level properties such as date formatting

The single value \*NONE indicates that CoolSpools Spool Converter will take all defaults. No field-level overrides will occur.

There are two ways in which to associate a style with a piece of data in your output file (e.g. a cell in an Excel spreadsheet or an element of an XML document):

#### **1. Implicitly**

By defining the style name the same as the name of a row group in your Database-to-Excel map or an element in your Database-to-XML map, you implicitly apply that style to the data in question. For example, a style called REPORT\_HEADING will be implicitly and automatically applied to an Excel row group called REPORT\_HEADING.

***Note that style names are case-sensitive because XML element names need to be case-sensitive and for this association of names to work, the names must match exactly in terms of case.***

#### **2. Explicitly**

Alternatively, use the APYSTYLES parameter to define the styles you wish to apply to different parts of the file you create.

Note that, unlike CVTSPLXL, CVTSPLXLS does not support the definition of styles on the command string itself by means of the DFNSTYLES parameter. CVTSPLXLS can only use style definitions created with the CRTSTLDFN command and managed using WRKSTLDFN etc.

#### **Row group name (CVTSPLXL)**

#### **Line type name (CVTSPLXLS)**

#### **Element name (CVTSPLXML)**

Specifies the rows to which the styling should be applied.

Options are:

<b><u>*ANY</u></b>	This parameter defines styling that applies to all rows generated by all row groups (Excel) or elements (XML).
<b>*DETAILS</b>	(CVTSPLXLS only) Apply this styling to detail lines. A detail line is defined as any line type that does not match a line type specified on the LINTYPES parameter, which is typically used to define line types that are exceptions from the most common type of line in the report.
<b>row_group_name</b>	(CVTSPLXL only) This parameter relates only to rows generated from the specified row group.
<b>line_type_name</b>	(CVTSPLXLS only) Specifies the line types to which the styling should be applied. Must be one of the special values shown below or match the name of a line type defined on the LINTYPES parameter.
<b>element_name</b>	(CVTSPLXML only) This parameter relates only to nodes generated from the specified element.

### Row number (CVTSPLXL)

### Relative line number (CVTSPLXLS)

#### CVTSPLXL

Specifies the row number within the row group to which the styling applies.

<b><u>*ANY</u></b>	This parameter defines styling that applies to all rows generated by the row group specified above.
<b>row_number</b>	Specifies the relative row within the row group to which the styling applies, e.g. 1 = first row, 2 = second row.

#### CVTSPLXLS

Specifies the relative line number within the group to which the styling applies. Where a line type is specified on the LINTYPES parameter, it is possible to identify that line type as relating to a range of line numbers on the page. For example, column headings might appear on lines 6 to 8 of each page. If you wish to apply different styling to different lines in that group, e.g. underlining the last line of column headings perhaps, this can be done by using this element to identify the line number in the group of lines to which the styling should apply. In the case of the column heading example, the relative line number would be 3 (the third line of the group of three lines starting on line 6 and ending on line 8).

<b><u>*ANY</u></b>	This parameter defines styling that applies to any line that matches the line type specified above.
<b>line_number</b>	Specifies the relative line number within the group of lines to which the styling applies, e.g. 1 = first line, 2 = second line.

This line number must be between 1 and the total number of lines in the line group, as defined by the from- and to- lines specified for this line type on the LINTYPES parameter.

### Column reference (CVTSPLXL)

Specifies the column reference to which the styling applies.

**\*ANY**

This parameter defines styling that applies to all columns on the row.

**column\_ref**

Specifies the column on the row to which the styling applies, e.g. A = first column, B = second column.

### Style name

Specifies the name of the style to apply. The style must have been created using WRKSTLDFN or CRTSTLDFN or (except in the case of CVTSPLXLS) defined on the DFNSTYLES parameter.

Style names are case-sensitive.

## AUT - Public data authority

Parameter	<b>AUT</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, RTVPCLRSC, RTVSPLDTA, SAVSPLF</b>
Dependent on:	<b>Only shown if F10 pressed</b>

This parameter allows you to specify the public data authorities given to stream files created by CoolSpools.

In relation to CVTSPLSTMF, this parameter has two elements, but all other commands just have the first element.

### Public data authority

This option lets you to define the public data authority for the stream to be created.

The owner of the file is always granted full authority. This parameter controls the authority given to other users.

Options are:

*R	Read only
*W	Write only
*X	Execute only
*RW	Read and write
*RX	Read and execute
*WX	Write and execute
*RWX	Read, write and execute (all)
*NONE	No authority
autl_name	Alternatively, specify the name of an authorization list. This authorization list will be associated with the stream file and authorities for *PUBLIC assigned from the authorization list.

### Authority for PDF work files

This element is obsolete and provided with CVTSPLSTMF for reasons of backwards compatibility only. Any value specified is ignored.

## ***BLANKS - Include blank lines?***

Parameter	<b>BLANKS</b>
Applies to commands:	<b>CVTSPLSTMF</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and TOFMT(*TEXT), TOFMT(*HTML) or TOFMT(*HTXT)</b>

The **BLANKS** parameter is only available from the CVTSPLSTMF command and is relevant only to \*TEXT, \*HTML or \*HTXT output.

This parameter allows you to define whether blank lines in the original report should be duplicated in the output.

<b><u>*NO</u></b>	Blank lines in the original report are not reflected in the output and are compressed out.
<b>*YES</b>	Blank lines in the original report are reflected in the output. Pages are padded out with blank lines to resemble the printed page.
<b>*FF</b>	Blank lines in the original report are reflected in the output. At the end of each page, a form feed character (x'0C') is embedded in the output to force a page throw.

Please note that the equivalent to this parameter on the CVTSPLTXT command is the second element of the TEXT parameter of that command.

## **BMARKKEY- PDF bookmark string key**

Parameter	<b>BMARKKEY</b>
Applies to commands:	<b>CVTSPLPDF</b>

This parameter allows you to define the position in the report where the text to be used as a bookmark appears.

By specifying **BOOKMARK(\*KEY)** with the **BMARKKEY** parameter, you can create a set of bookmarks based on a piece of text that appears in the report on the same line as another piece of text (i.e. the key string, typically a field label) .

For example, if you know that an item of user interest - such as an order number, a customer name, or a product code - appears on the same line as a label such as 'Customer name:', 'Order No:' or 'Product Id', then by using these labels as a key string you can generate bookmarks based on the actual customer name, order number or product code.

### **Example:**

```
CVTSPLPDF ...  BOOKMARK(*KEY)
                  BMARKKEY(('Customer name:' 1 16 40))
```

### **Key string**

Specify the key string which will trigger the selection of bookmark text.

This value is case-sensitive.

### **Occurrence**

Where the key string appears more than once on each page, the number you enter on this parameter element will determine which occurrence of the key string will trigger the selection of bookmark text.

### **Offset**

Enter the offset in characters

If a positive number is entered, this is interpreted as indicating that the bookmark text is to the right of the key string, whereas a negative number indicates that the bookmark text is to the left of the key string.

### **Length**

The length of the bookmark text in characters.



## ***BMARKPOS- PDF bookmark string position***

Parameter	<b>BMARKPOS</b>
Applies to commands:	<b>CVTSPLPDF</b>

This parameter allows you to define the position in the report where the text to be used as a bookmark regularly appears.

By specifying BOOKMARK(\*POS) and using this BMARKPOS parameter, you can create a set of bookmarks based on a piece of text that appears at a particular position on each page of the report.

For example, if you know that an item of user interest - such as an order number, a customer name, or a product code - appears regularly at position 3 of line 4 on every page and is up to 20 characters long, you can generate your bookmarks by specifying:

```
CVTSPLPDF      FROMFILE(SALES)...  
                BOOKMARK(*POS)  
                BMARKPOS((4 3 20))
```

### ***Line number***

Enter either the line number on which the bookmark text appears in the spooled file.

### ***Character position***

The column number on which the bookmark text appears in the spooled file.

### ***Length***

Enter the number of characters which the bookmark occupies in the spooled file.

## BOOKMARK - PDF bookmarks

Parameter	<b>BOOKMARK</b>
Applies to commands:	<b>CVTSPLPDF</b>

This parameter controls the type of PDF bookmarks (outlines) that CoolSpools generates.

Bookmarks index a PDF file so readers can go directly to a particular section of a document. By creating bookmarks, you make it quicker and easier for users to navigate around a document in PDF format.

The CVTSPLPDF command supports the creation of bookmarks based on multiple text items extracted from the spooled file. A nested bookmark structure will be generated where multiple bookmarks triggers are defined.

The possible values are:

<b>*PAGNBR</b>	CoolSpools creates bookmarks based on the page numbers in the document. This is the default for CVTSPLSTMF.  Note that the text used to generate *PAGNBR bookmarks is held in message CVT0008 in message file CP_MSGF. By default it is set to "Page". You can modify the text of this message if you wish to. For example, if your native language is Spanish, you may wish to change it to "Página". If you do this, please remember that you will need to change the text again every time you apply a PTF or upgrade to a new version.
<b>*NONE</b>	No bookmarks are required. This is default for CVTSPLPDF.
<b>*POS</b>	Indicates that you will define bookmarks on the BMARKPOS parameter and that CoolSpools should create bookmarks based on a piece of text that appears at a particular position on each page of the report.
<b>*KEY</b>	Indicates that you will define bookmarks on the BMARKKEY parameter and that CoolSpools should create bookmarks based on an item of text associated with a key word or phrase found in the report.
<b>*POSKEY</b>	Indicates that you will define bookmark parameters on both the BMARKPOS and BMARKKEY parameters and that both positional and key bookmarks should be created. This option is not supported by CVTSPLSTMF.
<b>*PAGIDXTAG</b>	Bookmarks will be generated from page-level document index tags, for example those included

	in the spooled file with the DDS DOCIDXTAG keyword with the tag level equal to PAGE.
<b>*GRPIDXTAG</b>	Bookmarks will be generated from group-level document index tags, for example those included in the spooled file with the DDS DOCIDXTAG keyword with the tag level equal to GROUP.
<b>*IDXTAG</b>	Bookmarks will be generated from document index tags, for example those included in the spooled file with the DDS DOCIDXTAG keyword with the tag level equal to GROUP or PAGE.

***Example:***

***CVTSPLPDF      FROMFILE(SALES)...  
BOOKMARK(\*PAGNBR)***

The sales report is converted to PDF format and bookmarks are generated for each page of the report, labeled 'Page 1', 'Page 2' etc.

## CNDFMTGRP – Conditional formatting groups

Parameter	<b>CNDFMTGRP</b>
Description	<b>Defines groups of conditional formatting rules and the range of cells to which they apply</b>
Applies to commands:	<b>CVTSPLXL, CVTSPLXLS</b>
Dependent on:	

Specifies conditional formatting rule groups. A conditional formatting rule group defines a group of related rules which will be applied, in a given priority sequence, to a range of cells in order to determine the appearance of those cells. For example, you might define a rule that tests the value of a customer account balance field and makes rows where the balance is negative red and those where it is above a certain level green etc.

See the CNDFMTRULE parameter below for examples of how to define conditional formatting.

### Rule group number

Specifies an arbitrary, non-zero, positive integer which identifies the rule group. You can choose any number you like to identify the group, but it must be unique for all rule groups defined on the command.

The rule group number is used to match rules defined on the CNDFMTRULE parameter against rule groups defined on the CNDFMTGRP parameter. The CNDFMTGRP parameter defines group-level attributes such as the range of cells to which the rules should be applied, whereas the CNDFMTRULE parameter defines the individual rules in the group that will be tested, one after another, in the priority sequence you specify, against those cells.

### Rule group name

Specifies an optional, arbitrary name which identifies the rule group. You can choose any name you like to identify the group. The name has no function other than to help you document and remember the purpose of a given rule group.

### Apply to map references (CVTSPLXL)

### Apply to rows/columns (CVTSPLXLS)

Specifies the rows in the worksheet the rules should be applied to.

The default is the single value:

**\*ALL**

(Default). The rules are applied to all rows in the worksheet, including those not populated by data. If new data is entered after the last row of data, the rules will apply to those new rows too.

Alternatively specify between 1 and 50 map references (CVTSPLXL) or row/column references (CVTSPLXLS) that identify the data to which the rules should be applied.

For CVTSPLXL, use map references in one of the forms described below.

<b>row_group_name</b>	all cells generated from the row group called row_group_name e.g. DETAIL_ROW
or	
<b>row_group_name(col)</b>	all cells generated from the row group called row_group_name where the column letter is <b>col</b> e.g. DETAIL_ROW(A)
or	
<b>row_group_name(colrow)</b>	all cells generated from the row group called row_group_name where the column letter is <b>col</b> and the row number in the group is <b>row</b> , e.g. DETAIL_ROW(A2)

For CVTSPLXLS, use row/column references in one of the forms described below.

<b>line_type_name</b>	all cells on lines matched to a line type defined on the LINTYPES parameter where the name matches line_type_name e.g. HEADER_LINE
or	
<b>line_type_name(col)</b>	all cells on lines matched to a line type defined on the LINTYPES parameter where the name matches line_type_name and where the column letter is <b>col</b> e.g. HEADER_LINE(A)
<b>line_type_name(colline)</b>	all cells on lines matched to a line type defined on the LINTYPES parameter where the name matches line_type_name, where the column letter is <b>col</b> and the relative line number in the line group is <b>line</b> e.g. HEADER_LINE(A2)

## **CNDFMTRULE – Conditional formatting rules**

Parameter	<b>CNDFMTRULE</b>
Description	<b>Defines individual conditional formatting rules</b>
Applies to commands:	<b>CVTSPLXL, CVTSPLXLS</b>
Dependent on:	<b>None</b>

Specifies conditional formatting rules.

A conditional formatting rule group defines a group of related rules which will be applied, in a given priority sequence, to a range of cells in order to determine the appearance of those cells. For example, you might define a rule that tests the value of a customer account balance field and makes rows where the balance is negative red and those where it is above a certain level green etc.

The CNDFMTGRP parameter defines conditional formatting rule groups and group-level attributes such as the range of cells to which the rules in the group will be applied.

The CNDFMTRULE parameter defines the individual rules within those groups which are tested in turn.

Please note that Excel does not allow all of the attributes that can be specified for a style on the DFNSTYLE parameter or CRTSTLDFN command to be modified when using conditional formatting. In addition, the number of attributes that can be set when defining conditional formatting is even more limited when you are outputting to \*XLS format. In particular, changing of number formats, font names and font sizes through conditional formatting was not supported before Excel 2007. Therefore, a format compatible with Excel 2007 must be used if this feature is required, i.e. EXCEL(\*XLSX) or EXCEL(\*XLS07) must be specified.

Also, there are restrictions on the types of test that can be used when outputting to \*XLS format. Versions of Excel prior to Excel 2007 do not support many of the more complex test types shown below. The \*FORMULA test type is not supported when outputting to \*XLS format.

### **Rule group number**

Specifies an arbitrary, non-zero, positive integer which identifies the rule group. You can choose any number you like to identify the group, but it must be unique for all rule groups defined on the command.

The rule group number you specify here must correspond to the rule group number of a rule group defined on the CNDFMTGRP parameter. The CNDFMTGRP parameter defines group-level attributes such as the range of cells to which the rules should be applied, whereas the CNDFMTRULE parameter defines the individual rules in the group that will be tested, one after another, in the priority sequence you specify, against those cells.

### **Rule priority**

The priority of this conditional formatting rule. Where the group contains several rules, this value is used to determine which rule takes precedence and therefore

which style is applied. Lower numeric values assign a higher priority than higher numeric values, i.e. 1 is the highest priority.

### Field to test

Specifies the data item in the report which is tested in order to determine if the rule should evaluate to true or false.

Options are:

#### **\*CELLIS**

(Default). The logic test is carried out on each individual cell within the range of cells to which this rule is applied, not to any particular column.

For example, if you were using conditional formatting rules to apply different colors, and you specify a \*CELLIS rule, each separate cell in the range of cells to which the rules apply will be colored differently depending on the value of those individual cells.

#### **\*FORMULA**

You will specify a formula on the **Parameter value 1** element below. That formula will determine whether the rule evaluates to true or false and therefore what styling is applied.

Not supported when outputting to \*XLS format.

#### **Excel\_col\_ref**

Specify an Excel column letter. CoolSpools Spool Converter will generate a formula which carries out the required logic test against the value of this column in the rows to which the rule applies. For each row, the value of this column in that row will determine the formatting of cells to which this rule applies.

For example, if your file contains customer account details, and you wish to color the rows based on the value of the customer's account balance, and that balance is in column M, you would specify column M here as that is the field which determines how the rows should be formatted.

### Test to apply

Specifies the logic test which is carried out to determine if the rule evaluates to true or false and therefore what styling to apply.

Options are:

#### **\*NONE**

(Default) None. Only valid if \*FORMULA is specified or the previous element, i.e. you will specify your own formula to apply on the **Parameter value 1** element below.

The following tests compare the value of field identified by the previous parameter element, or each individual cell (if \*CELLIS was specified), against the parameter value or values specified on the **Parameter value 1** and **Parameter value 2** elements below.

<b>*EQ</b>	Equal.  A single value must be specified on <b>Parameter value 1</b> below. The rule is true if the field or cell value is equal to this value.
<b>*GT</b>	Greater than.  A single value must be specified on <b>Parameter value 1</b> below. The rule is true if the field or cell value is greater than this value.
<b>*LT</b>	Less than.  A single value must be specified on <b>Parameter value 1</b> below. The rule is true if the field or cell value is less than this value.
<b>*GE</b>	Greater than or equal to.  A single value must be specified on <b>Parameter value 1</b> below. The rule is true if the field or cell value is greater than or equal to this value.
<b>*LE</b>	Less than or equal to.  A single value must be specified on <b>Parameter value 1</b> below. The rule is true if the field or cell value is less than or equal to this value.
<b>*NE</b>	Not equal to.  A single value must be specified on <b>Parameter value 1</b> below. The rule is true if the field or cell value is not equal to this value.
<b>*BETWEEN</b>	Between.  Two values must be specified on <b>Parameter value1</b> and <b>Parameter value 2</b> below. The rule is true if the field or cell value is greater than or equal to the first value and less than or equal to the second value.
<b>*NOTBETWEEN</b>	Not between.



Two values must be specified on **Parameter value1** and **Parameter value 2** below. The rule is true if the field or cell value is less than the first value or greater than the second value.

**\*CT**

Contains.

A single value must be specified on **Parameter value 1** below and it is interpreted as a text string. The rule is true if the field or cell value contains the text string specified

Not supported when outputting to \*XLS format.

**\*CONTAINS**

Same as \*CT.

**\*NC**

Not contains.

A single value must be specified on **Parameter value 1** below and it is interpreted as a text string. The rule is true if the field or cell value does not contain the text string specified.

Not supported when outputting to \*XLS format.

**\*NOTCONTAINS**

Same as \*NC.

**\*BEGINSWITH**

Begins with.

A single value must be specified on **Parameter value 1** below and it is interpreted as a text string. The rule is true if the field or cell value begins with the text string specified.

Not supported when outputting to \*XLS format.

**\*ENDSWITH**

Ends with.

A single value must be specified on **Parameter value 1** below and it is interpreted as a text string. The rule is true if the field or cell value ends with the text string specified

Not supported when outputting to \*XLS format.

**\*BLANKS**

Contains blanks.

The value of the **Parameter value 1** below is irrelevant and is ignored. The rule is true if the field

or cell value is blank (is empty or contains only spaces).

Not supported when outputting to \*XLS format.

**\*NOTBLANKS**

Does not contain blanks.

The value of the **Parameter value 1** below is irrelevant and is ignored. The rule is true if the field or cell value is not blank (is not empty or does not contain only spaces).

Not supported when outputting to \*XLS format.

**\*TIMEPERIOD**

Time period.

The value of the **Parameter value 1** below must be one of the special time-period values listed below (\*LASTMONTH etc.). The rule is true if the field or cell value is number which Excel can interpret as a date and that date matches the time period specified.

Not supported when outputting to \*XLS format.

The following tests are only supported if \*CELLIS was specified for the **Field to test** element. Each value of cell in the range covered by the rule group is tested individually.

**\*TOPN**

Top n values.

The value of the **Parameter value 1** below must be a number indicating the value of n. The rule is true if the field or cell value in the top n values.

Not supported when outputting to \*XLS format.

**\*BOTTOMN**

Bottom n values.

The value of the **Parameter value 1** below must be a number indicating the value of n. The rule is true if the field or cell value in the bottom n values.

Not supported when outputting to \*XLS format.

**\*TOPNPC**

Top n percent.

The value of the **Parameter value 1** below must be a number indicating the value of n. The rule is true if the field or cell value in the top n percent of values.

Not supported when outputting to \*XLS format.

**\*BOTTOMNPC**

Bottom n percent.

The value of the **Parameter value 1** below must be a number indicating the value of n. The rule is true if the field or cell value in the bottom n percent of values.

Not supported when outputting to \*XLS format.

**\*DUPLICATE**

Duplicate values

The value of the **Parameter value 1** below is irrelevant and is ignored. The rule is true if the field or cell value is not unique in the range.

Not supported when outputting to \*XLS format.

**\*UNIQUE**

Duplicate values

The value of the **Parameter value 1** below is irrelevant and is ignored. The rule is true if the field or cell value is unique in the range.

Not supported when outputting to \*XLS format.

**Parameter value 1**

The first parameter value required for the test defined above.

The interpretation of the parameter element is dependent on the value of the **Test to apply** element:

Value of <b>Test to apply</b>	Interpretation of <b>Parameter value 1</b>
*EQ, *LT, *LE, *GT, *GE, *NE	A value representing a number or string, e.g. 1000 New York
*BETWEEN, *NOTBETWEEN	The first of a pair of values representing numbers or strings. The second value in the pair must be specified on Parameter value 2. 1000 A
*CT, *NC, *CONTAINS,	A value representing a text string, e.g. New York

*NOTCONTAINS, *BEGINSWITH, *ENDSWITH	
*TIMEPERIOD	Must be one of the special time-period values specified below, e.g. *LASTMONTH.
*TOPN, *BOTTOMN	The ranking value, e.g. 10 = "Top 10"
*TOPNPC, *BOTTOMNPC	The percentage value, e.g. 10 = "Top 10%"

When **Test to apply** is \*TIMEPERIOD, the value must be one of the following special time periods:

<b>*THIS MONTH</b>	This month. The date falls in the current calendar month.
<b>*LASTMONTH</b>	Last month. The date falls in the previous calendar month.
<b>*NEXTMONTH</b>	Next month. The date falls in the following calendar month.
<b>*THISWEEK</b>	This week. The date falls in the current week.
<b>*LASTWEEK</b>	Last week. The date falls in the previous week.
<b>*NEXTWEEK</b>	Next week. The date falls in the next week.
<b>*LAST7DAYS</b>	Last 7 days. The date falls in the last seven days.
<b>*TODAY</b>	Today. The date is the current date
<b>*YESTERDAY</b>	Yesterday. The date is one day prior to the current date.
<b>*TOMORROW</b>	Tomorrow. The date is one day after the current date.

When **Field to test** is \*FORMULA, you must specify a formula of your own on this parameter element. If the result of the formula is true, the style associated with this rule will be applied. When specifying cell references in your formula, the row number should correspond to the data row in the worksheet, taking account of column headings and additional heading rows. Use a relative column reference to test each cell in the range separately or an absolute column reference to test the value of a specific column. Do NOT precede the formula by an equals sign = as you might do in a cell.

There is one other special value: **\*AVG**. This allows you test against the average value for the selected range. This is only permitted where:

- **Field to test** is \*CELLIS
- **Test to apply** is \*EQ, \*GT, \*LT, \*LE or \*GE

## Parameter value 2

The second parameter value required for the test defined above.

The default is **\*NONE**.

A value other than **\*NONE** must be specified if the test is **\*BETWEEN** or **\*NOTBETWEEN**. The value specified here must be greater than or equal to the value specified on **Parameter value 1**.

A value other than **\*NONE** not be specified for any other test.

### Apply style name

The name of the style to apply if the rule evaluates to true.

The style name must match the name of a style defined with WRKSTLDFN or CRTSTLDFN or specified on the DFNSTYLES parameter.

Note that Excel does not allow all of the attributes that can be defined on the DFNSTYLES parameter to be controlled by conditional formatting. For example, while you can change the text color or make the text bold or italic, you cannot change the font name or font size. If you attempt to modify these using conditional formatting, Excel will ignore that change.

### Stop if true

Determines whether Excel stops evaluating rules in the group as soon as one has evaluated to true or whether it carries on and checks the next rule.

Options are:

<b>*YES</b>	(Default). No rules with lower priority may be applied over this rule, when this rule evaluates to true
<b>*NO</b>	Other rules with a lower priority will also be evaluated and may override aspects of the formatting.

### Examples

The following examples assume that the WRKSTLDFN or CRTSTLDFN command, or the DFNSTYLES parameter (not shown here for the sake of clarity) has been used to define styles called RED, ORANGE and YELLOW (which might set the cell colors to have a red, orange or yellow background, for example).

#### Example 1:

```
CVTSPLXL
FROMFILE(CUSTACCT)
...
CNDFMTGRP(  (1 BALANCES DETAIL_ROW)
            (2 DUESOON *ALL 'DETAIL_ROW(N)'))
CNDFMTRULE( (1 1 M *LT 0 RED *YES)
            (1 2 M *BETWEEN 0 100 ORANGE)
            (2 1 N *TIMEPERIOD *NEXTMONTH *NONE YELLOW))
```

Here the customer accounts report is being converted to an Excel spreadsheet.

Two groups of conditional formatting rules are defined:

- Group 1 (named “BALANCES”) which is applied to all rows derived from the row group called DETAIL\_ROW.

This has two rules:

- If the value of column M is zero, the entire row will have the RED style applied to it
- If the value of column M is between 0 and 100, the entire row will have the ORANGE style applied to it.
- Group 2 (named “DUESOON”) which is applied just to the date in column N of rows derived from DETAIL\_ROW.

This has a single rule:

- If the date in column N is in the following calendar month, the YELLOW style is applied to it.

## COLOR - Colors

Parameter	<b>COLOR</b>
Applies to commands:	<b>CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSTMF</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and TOFMT(*PDF)</b>

This parameter allows you to define the color of text in the document that CoolSpools creates and the color of the background on which that text is presented.

There are two elements to this parameter:

- **Text color**
- **Background color**

In each instance, colors can be defined in one of two ways:

- **A predefined color name such as \*BLACK, \*WHITE, \*RED etc.**

The list of predefined names includes all those normally recognized in HTML by browser applications.

- **An RGB (Red-Green-Blue) color number, equivalent to HTML color numbers**

An RGB color number is a string consisting of six hexadecimal digits (0-F).

The first two digits represent the red color value (00-FF),

The next two digits represent the green color value (00-FF),

The last two digits represent the blue color value (00-FF),

For example, white is FFFFFFFF, while black is 000000, and red is FF0000.

Unlike HTML, a CoolSpools color number should not be prefixed with a hash symbol (#).

### Text color

The first element allows you to specify the color to be used to display black text in the original spooled file.

Any text in the spooled file which is not black will retain its original color. Any black text will assume the color specified here instead.

### Background color

The second element determines the color of the background on which the text appears.

Options for both the text and background color are as follows (the table also indicates the corresponding color value). The list of predefined color names available from CVTSPLSTMF is a subset of those available from the other commands which have the COLOR parameter.

Color codes are:

<b>*ALICEBLUE</b>	<b>F0F8FF</b>
<b>*ANTIQUEWHITE</b>	<b>FAEBD7</b>
<b>*AQUA</b>	<b>00FFFF</b>
<b>*AQUAMARINE</b>	<b>7FFFD4</b>

*AZURE	F0FFFF
*BEIGE	F5F5DC
*BISQUE	FFE4C4
*BLACK	000000
*BLANCHEDALMOND	FFEBCD
*BLUE	0000FF
*BLUEVIOLET	8A2BE2
*BROWN	A52A2A
*BURLYWOOD	DEB887
*CADETBBLUE	5F9EA0
*CHARTREUSE	7FFF00
*CHOCOLATE	D2691E
*CORAL	FF7F50
*CORNFLOWERBLUE	6495ED
*CORN SILK	FFF8DC
*CRIMSON	DC143C
*CYAN	00FFFF
*DARKBLUE	00008B
*DARKCYAN	008B8B
*DARKGOLDENROD	B8860B
*DARKGRAY	A9A9A9
*DARKGREY	A9A9A9
*DARKGREEN	006400
*DARKKHAKI	BDB76B
*DARKMAGENTA	8B008B
*DARKOLIVEGREEN	556B2F
*DARKORANGE	FF8C00
*DARKORCHID	9932CC
*DARKRED	8B0000
*DARKSALMON	E9967A
*DARKSEAGREEN	8FBC8F
*DARKSLATEBLUE	483D8B
*DARKSLATEGRAY	2F4F4F
*DARKSLATEGREY	2F4F4F
*DARKTURQUOISE	00CED1
*DARKVIOLET	9400D3
*DEEPPINK	FF1493
*DEEPSKYBLUE	00BFFF
*DIMGRAY	696969
*DIMGREY	696969
*DODGERBLUE	1E90FF
*FELD SPAR	D19275
*FIREBRICK	B22222
*FLORALWHITE	FFFAF0
*FORESTGREEN	228B22
*FUCHSIA	FF00FF
*GAINSBORO	DCDCDC
*GHOSTWHITE	F8F8FF
*GOLD	FFD700
*GOLDENROD	DAA520
*GRAY	808080



*GREY	808080
*GREEN	008000
*GREENYELLOW	ADFF2F
*HONEYDEW	F0FFF0
*HOTPINK	FF69B4
*INDIANRED	CD5C5C
*INDIGO	4B0082
*IVORY	FFFFF0
*KHAKI	F0E68C
*LAVENDER	E6E6FA
*LAVENDERBLUSH	FFF0F5
*LAWNGREEN	7CFC00
*LEMONCHIFFON	FFFACD
*LIGHTBLUE	ADD8E6
*LIGHTCORAL	F08080
*LIGHTCYAN	E0FFFF
*LIGHTGOLDENROD	FAFAD2
*LIGHTGRAY	D3D3D3
*LIGHTGREY	D3D3D3
*LIGHTGREEN	90EE90
*LIGHTPINK	FFB6C1
*LIGHTSALMON	FFA07A
*LIGHTSEAGREEN	20B2AA
*LIGHTSKYBLUE	87CEFA
*LIGHTSLATEBLUE	8470FF
*LIGHTSLATEGRAY	778899
*LIGHTSLATEGREY	778899
*LIGHTSTEELBLUE	B0C4DE
*LIGHTYELLOW	FFFFE0
*LIME	00FF00
*LIMEGREEN	32CD32
*LINEN	FAF0E6
*MAGENTA	FF00FF
*MAROON	800000
*MEDIUMAQUAMARINE	66CDAA
*MEDIUMBLUE	0000CD
*MEDIUMORCHID	BA55D3
*MEDIUMPURPLE	9370D8
*MEDIUMSEAGREEN	3CB371
*MEDIUMSLATEBLUE	7B68EE
*MEDIUMSPRINGGREEN	00FA9A
*MEDIUMTURQUOISE	48D1CC
*MEDIUMVIOLETRED	C71585
*MIDNIGHTBLUE	191970
*MINTCREAM	F5FFFA
*MISTYROSE	FFE4E1
*MOCCASIN	FFE4B5
*NAVAJOWHITE	FFDEAD
*NAVY	000080
*OLDLACE	FDF5E6
*OLIVE	808000

*OLIVEDRAB	6B8E23	
*ORANGE	FFA500	
*ORANGERED	FF4500	
*ORCHID	DA70D6	
*PALEBLUE	ADD8E6	
*PALEBROWN	CD853F	
*PALECYAN	E0FFFF	
*PALEGOLDENROD	EEE8AA	
*PALEGRAY	D3D3D3	
*PALEGREY	D3D3D3	
*PALEGREEN	98FB98	
*PALEMAG	DDA0DD	Pale magenta
*PALETURQUOISE	AFEEEE	
*PALEVIOLETRED	D87093	
*PALEYLW	FFFFE0	Pale yellow
*PAPAYAWHIP	FFEDB5	
*PEACHPUFF	FFDAB9	
*PERU	CD853F	
*PINK	FFC0CB	
*PLUM	DDA0DD	
*POWDERBLUE	B0E0E6	
*PURPLE	800080	
*RED	FF0000	
*ROSYBROWN	BC8F8F	
*ROYALBLUE	4169E1	
*SADDLEBROWN	8B4513	
*SALMON	FA8072	
*SANDYBROWN	F4A460	
*SEAGREEN	2E8B57	
*SEASHELL	FFF5EE	
*SIENNA	A0522D	
*SILVER	C0C0C0	
*SKYBLUE	87CEEB	
*SLATEBLUE	6A5ACD	
*SLATEGRAY	708090	
*SLATEGREY	708090	
*SNOW	FFFAFA	
*SPRINGGREEN	00FF7F	
*STEELBLUE	4682B4	
*TAN	D2B48C	
*TEAL	008080	
*THISTLE	D8BFD8	
*TOMATO	FF6347	
*TURQUOISE	40E0D0	
*VIOLET	EE82EE	
*VIOLETRED	D02090	
*WHEAT	F5DEB3	
*WHITE	FFFFFF	
*WHITESMOKE	F5F5F5	
*YELLOW	FFFF00	
*YELLOWGREEN	9ACD32	

***Example:***

***CVTSPLPDF      FROMFILE(SALES)...  
COLOR(\*BLUE \*PALEYLW)***

The sales report is converted to PDF. Any black text in the report will appear blue in Adobe Acrobat. Other colored text will retain its original color. The text will appear against a pale yellow background.

## COLWIDTHS – Column widths

Parameter	<b>COLWIDTHS</b>
Description	<b>Let you specify the width of columns</b>
Applies to commands:	<b>CVTSPLXL, CVTSPLXLS</b>
Dependent on:	<b>None</b>

The COLWIDTHS parameter allows you to define the widths of individual columns in an Excel workbook.

The single value \*NONE indicates that CoolSpools Spool Converter will use the default method of specifying column widths, taken from the EXCEL parameter.

This attribute has not been implemented for HTML and XML because browser behavior and support in this area is just too variable and unreliable.

There is a single value:

### **\*NONE**

(Default). Calculate the width of the column in the default manner, i.e. using the method specified on the **Column width option** element of the EXCEL parameter.

## Column id

The column identifier (letter or letters) identifying the column whose width is being defined.

## Column width

Set the width of the column for this field.

Options are:

### **column\_width**

Specify the column width in characters.

For Excel, this can be difficult for CoolSpools Spool Converter to calculate as it is dependent on the metrics of the fonts that are being used, which may not be available at the time the Excel workbook is created.

## CSV - CSV Options (CVTSPLCSV command)

Parameter **CSV**

Applies to commands: **CVTSPLCSV**

Dependent on: **None**

Specifies options for creating delimited text files such as Comma Separated Variable files (CSVs) and Tab Separated Variable Files (TSVs).

### Record delimiter

This element allows you to specify the characters to be used to indicate the end of a record in the CSV file.

Options are:

- |              |   |
|--------------|---|
| <b>*CRLF</b> | Carriage return and line feed. Both a carriage return (ASCII x'0D' or the equivalent in the CCSID selected) and a line feed (ASCII x'0A' or the equivalent in the CCSID selected) are used to denote the end of a record. |
| <b>*CR</b>   | Just a carriage return (ASCII x'0D' or equivalent) is used.   |
| <b>*LF</b>   | Just a line feed (ASCII x'0A' or equivalent) is used.   |

### String delimiter

This element allows you to define the character that encloses string (alphanumeric) data in the delimited file that is to be created.

Either type the character to be used, or select one of the special values:

- |                  |  |
|------------------|--|
| <b>*DBLQUOTE</b> | A double quote (") is used   |
| <b>*SGLQUOTE</b> | A single quote (') is used   |
| <b>*NONE</b>     | No delimiter is used. Alphanumeric data is not enclosed by any special character. Please note that this option could cause problems when the file is read if the string data includes the field delimiter character. |

**String\_delim** Type the delimiter character to be used.

The string delimiter will be output in the CCSID selected for the file.

### Field Delimiter

This element allows you to define the character that separates fields in the delimited file that is to be created.

Either type the character to be used, or select one of the special values:

- |               |   |
|---------------|---|
| <b>*COMMA</b> | A comma (,) is used   |
| <b>*TAB</b>   | A tab (ASCII x'09' or the equivalent in the CCSID selected) is used |

<b>*BLANK</b>	A space or blank (ASCII x'20' or the equivalent in the CCSID selected) is used
<b>*SEMICOLON</b>	A semicolon (;) is used.
<b>*PIPE</b>	A pipe character ( ) is used.
<b>Field_delim</b>	Type the delimiter character to be used.

The field delimiter will be output in the CCSID selected for the file.

### Keep page headings?

How CoolSpools handles page headings in the file.

Following statistical analysis of a sample of the data in the spooled file, CoolSpools will decide which lines are report data content and which not. Any lines which precede the first report data line, but which do not appear to be a column heading, will be considered a page heading. This element then determines how such lines are handled.

Options are:

<b><u>*FIRST</u></b>	The first occurrence of a unique page heading line is retained, but all subsequent occurrences of that line are dropped from the output.  Note that any variation in the page heading from one page to the next (such as a change in the time that is printed at the top of the page) may cause CoolSpools to retain a heading you would like to have dropped. You may need to consider using the EXCLLINBR or EXCLLINKEY parameters to exclude unwanted headings which CoolSpools does not successfully drop.
<b>*ALL</b>	All page headings are retained.
<b>*NONE</b>	All page headings are dropped.

### Keep column headings?

How CoolSpools handles column headings in the file.

Following statistical analysis of a sample of the data in the spooled file, CoolSpools will decide which lines are report data content and which not. Any lines which immediately precede the first report data line, and which overlap the data columns in the report, will be considered column headings.

This element then determines how such lines are handled.

Options are:

<b><u>*FIRST</u></b>	The first occurrence of a unique column heading line is retained, but all subsequent occurrences of that line are dropped from the output.  Note that any variation in the column heading from one page to the next may cause CoolSpools to retain a heading you would like to have dropped. You may need to consider using the EXCLLINBR or EXCLLINKEY parameters to exclude unwanted
----------------------	--

headings which CoolSpools does not successfully drop.

- \*ALL** All column headings are retained.
- \*NONE** All column headings are dropped.

### Spooled file currency symbol

This element defines the currency symbol that appears when printing currency values in the report.

It is important that CoolSpools knows what currency symbol is used in the report so that it can correctly identify columns of numbers that include a currency symbol as numeric data rather than treating them as text.

Options are:

- \*SYSVAL** The currency symbol defined by the QCURSYM system value is used in the report.
- currency\_symbol** Specify the currency symbol used in the report if this is different from the system currency symbol. For example, if you are processing a report containing values in Euros on a system where the currency symbol is a pound sign (£), specify €. CoolSpools will interpret data containing euro signs as numeric data not text.

### Spooled file decimal point

This element defines the decimal point that is used when printing numbers in the report.

It is important that CoolSpools knows what decimal point symbol is used in the report so that it can correctly identify columns of numbers as numeric data rather than treating them as text.

Options are:

- \*JOB** The decimal point defined by the DECFMT attribute of the current job is used in the report.
- \*SYSVAL** The decimal point defined by the QDECFMT system value is used in the report.
- Decimal\_point** Specify the decimal point used in the report. For example, if you are processing a report containing numbers that have a comma as the decimal point on a system where the normal decimal point is a period (.), specify , (comma). CoolSpools will interpret commas in numeric data as a decimal point, not a thousands separator.

### Spooled file 1000s separator

This element defines the thousands separator character that is used when printing numbers in the report.

It is important that CoolSpools knows what thousands separator character is used in the report so that it can correctly identify columns of numbers as numeric data rather than treating them as text.

Options are:

<b><u>*JOB</u></b>	The thousands separator character defined by the DECFMT attribute of the current job is used in the report.
<b>*SYSVAL</b>	The thousands separator character defined by the QDECFMT system value is used in the report.
<b>1000s_sep</b>	Specify the thousands separator character used in the report. For example, if you are processing a report containing numbers that have a period as the thousands separator character on a system where the normal thousands separator character is a comma (,)specify . (period). CoolSpools will interpret periods in numeric data as a thousands separator character, not a decimal point.

### Spoiled file date format

This element defines the date format that is used when printing dates in the report.

It is important that CoolSpools knows what date format is used in the report so that it can correctly identify dates and treat them as such.

Options are:

<b><u>*JOB</u></b>	The date format defined by the DATFMT attribute of the current job is used in the report.
<b>*SYSVAL</b>	The date format defined by the QDATFMT system value is used in the report.
<b>*DMY</b>	The date format used in the report is day-month-year. CoolSpools will identify data in the report which looks like a valid DMY date as a date (2-digit or 4-digit year).
<b>*MDY</b>	The date format used in the report is month-day-year. CoolSpools will identify data in the report which looks like a valid MDY date as a date (2-digit or 4-digit year).
<b>*YMD</b>	The date format used in the report is year-month-day. CoolSpools will identify data in the report which looks like a valid YMD date as a date (2-digit or 4-digit year).

### Spoiled file date separator

This element defines the date separator that is used when printing dates in the report.

It is important that CoolSpools knows what date separator is used in the report so that it can correctly identify dates and treat them as such.

Options are:



<b><u>*JOB</u></b>	The date separator defined by the DATFMT attribute of the current job is used in the report.
<b><u>*SYSVAL</u></b>	The date separator defined by the QDATFMT system value is used in the report.
<b>date_sep</b>	Specify the date separator character used in the report. For example, if you are processing a report containing dates that have a hyphen as the date separator on a system where the normal date separator character is a slash, specify - (hyphen).

### Spooled file word for 'Page'

This element defines the word “Page” as it appears in the report.

When excluding page headings, CoolSpools attempts to take account of lines which differ only by a change of page number. In order to do so, it looks for the word defined on this element followed by a number and treats that text as a page number and ignores it for the purposes of deciding whether a page heading is a new one or a repetition of a previous one.

Options are:

<b><u>*DFT</u></b>	The word for “Page” is taken from the text of message CVT0008 in message file CP_MSGF. This is shipped in the English version of CoolSpools as “Page”.  Please note that if you change the text in this message file, you will need to change it back again after applying PTFs or new versions.
<b>Word_for_page</b>	Specify the word for “Page” as it is used with page numbers in the report. For example, if it is abbreviated to “P.”, specify “P.” here. Similarly, if you are processing a Spanish-language report, you may need to specify Página.

### Example:

```
CVTSPLCSV      FROMFILE(SALESSTATS)...  
                CSV(*CRLF *DBLQUOTE *COMMA)
```

The Sales Stats report is converted to a delimited file is CSV (Comma-separated variable format). Records are terminated by a carriage return/line feed pair. Alphanumeric data is enclosed in double quotes. Fields are separated by commas.

## ***CUSTOMPAGE – Custom page size***

Parameter	<b>CUSTOMPAGE</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLTXT</b>
Dependent on:	<b>PAGESIZE(*CUSTOM)</b>

This parameter allows you to define a non-standard page size.

### **Page width**

Specify the width of the page in the units defined below.

### **Page length**

Specify the length of the page in the units defined below.

### **Unit of measure**

Specify the units in which the preceding dimensions are measured.

Options are:

<b>*MM</b>	Millimeters
<b>*CM</b>	Centimeters
<b>*INCH</b>	Inches

### ***Example:***

```
CVTSPLPDF      FROMFILE(SALES)...  
                PAGESIZE(*CUSTOM)  
                CUSTOMPAGE(10 10 *INCH)
```

The spooled file is converted to PDF format using 10-inch square paper.

## CVTFONTID - Convert font ids

Parameter	<b>CVTFONTID</b>
Applies to commands:	<b>CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSTMF</b>

This parameter gives you close control over how fonts that are identified by a font number (e.g. by means of the DDS FONT keyword or the FONT parameter of the CRTPRTF command) are processed when your spooled file is converted.

For each font identifier or combination of font identifier and font size, you can specify the predefined font to be used or a font object to be embedded. Where the specified “from” font id and font size appears in the spooled file, that font will be mapped to the font typeface and point size specified here, or the font object specified will be used to reproduce that font in PDF.

Up to 100 font mappings or embedded fonts may be defined.

When FONT(\*MAP) is specified, CoolSpools notifies you of the font mappings that it has chosen by sending a message to the job log when you run the command. If you are not satisfied with the appearance of your spooled file in the stream file, consider using this parameter or the CVTFNTRSC parameter to define your own font mappings. Refer to the discussion of the **FONT** parameter above for further information about how CoolSpools assists you in identifying the font ids that need to be mapped.

The default is **\*NONE**, which indicates that no user-defined font id mappings are specified.

Other than \*NONE, all other selections for this parameter consist of two pairs of elements:

- **From font id**

consisting of:

- **Font id**
- **Font size**
- **Size units**

and

- **To font**

consisting of:

- **Face**
- **Size**

### **From font id**

There is a single value:

**\*CPI**

This denotes the font used in spooled files where FONT(CPI) is specified, i.e. the font for basic text in the spooled file where the text in question is not associated with any of the DDS keywords FONT, FNTCHRSET or CDEFNT and the spooled file does not have any of the attributes:

- FONT(font\_ID)
- FNTCHRSET(font\_character\_set\_name)
- CDEFNT(coded\_font\_name)

Other values consist of:

### **Font Id**

Enter the font number which identifies the font in the spooled file. This will usually be a font number defined either on the **FONT** parameter of the **CRTPRTF** (Create Printer File) command or on the DDS **FONT** keyword, e.g. 11 = Courier 10 CPI).

It can also be a PCL built-in font number.

### **Font Size**

The default for the “from” font point size is \*FONTID, which indicates that the font point size implied by the font identifier should be assumed. Some IBM font identifiers (e.g. 11 = Courier 10 CPI) imply a specific font size, while others (e.g. 5707 = Times Roman Bold) do not. This is normally implemented on the system i through the use of the printer file DDS keyword **FONT** with or without the **\*POINTS** option.

### **Size units**

Options are:

<b>*POINTS</b>	The font size denotes the height of characters in points.
<b>*CPI</b>	The font size denotes the pitch (width) of characters in characters per inch.

### **To Font**

This consists of:

### **Face**

This specifies how the font will be implemented inside the PDF and can be any of:

- a built-in PDF font
- a system i font resource object
- a TrueType or PostScript Type 1 font file located in the IFS

The following built-in typefaces are available for selection:

<b>*COURIER</b>	<b>Courier</b>
<b>*COURIERB</b>	<b>Courier Bold</b>
<b>*COURIERO</b>	<b>Courier Oblique</b>
<b>*COURIERBO</b>	<b>Courier Bold Oblique</b>
<b>*HELVETICA</b>	<b>Helvetica</b>
<b>*HELVB</b>	<b>Helvetica Bold</b>
<b>*HELVO</b>	<b>Helvetica Oblique</b>
<b>*HELVB</b>	<b>Helvetica Bold</b>
<b>*HELVBO</b>	<b>Helvetica Bold Oblique</b>
<b>*TIMES</b>	<b>Times Roman</b>
<b>*TIMESB</b>	<b>Times Roman Bold</b>
<b>*TIMESI</b>	<b>Times Roman Italic</b>
<b>*TIMESBI</b>	<b>Times Roman Bold Italic</b>
<b>*SYMBOL</b>	<b>Symbol</b>
<b>*DINGBATS</b>	<b>Zapf Dingbats</b>

**Example:**

```
CVTSPLPDF      FROMFILE(SALES)
                TOSTMF(sales.pdf)
                FONT(*MAP)
                CVTFONTID(((11) (*COURIER 10)))
```

Here, the sales report is converted to PDF format. CoolSpools will attempt to map fonts, but font id 11 is explicitly mapped to Courier 10-point.

Alternatively, you can specify either a font resource object or a PostScript Type 1 font file.

Note that you still need to specify **FONT(\*EMBED)** if you want the font to be embedded in PDF. If you specify **FONT(\*MAP)**, CoolSpools will use the font specified on this element of the **CVTFONTID** parameter to select a suitable standard font to map to, but will still use a mapped standard font not an embedded font.

These parameters require an IFS path name. If you are referencing a font resource object, you must therefore specify the object name in IFS format, i.e.

```
/QSYS.LIB/library_name.LIB/object_name.FNTRSC
```

**Example:**

```
CVTSPLPDF...
CVTFONTID(((416)
('/QSYS.LIB/QFNTCPL.LIB/ C0S0CR10.FNTRSC))) FONT(*EMBED)
```

Font id 416 will be implemented by embedding font resource object C0S0CR10 in library QFNTCPL

**Example:**

```
CVTSPLPDF...
CVTFONTID ((416)
('/QIBM/PRODDATA/OS400/FONTS/PSFONTS/LATIN/COU.PFB'))
FONT(*EMBED)
```

Font id 416 should be implemented by embedding the Postscript courier font supplied by IBM at the path given.

Please note that **two** files are required in order to successfully embed a PostScript font:

- i) A PostScript Type 1 font file. This normally has a file extension of .pfb (e.g. /QIBM/PRODDATA/OS400/FONTS/PSFONTS/LATIN/HEL.PFB)
- ii) A PostScript Type 1 font metrics file. This normally has a file extension of .afm (e.g. /QIBM/PRODDATA/OS400/FONTS/PSFONTS/LATIN/HEL.AFM)

Specify the name of the font file (.pfb extension) on this parameter. CoolSpools will attempt to locate the corresponding font metrics file (.afm extension) in the same location. If either file cannot be found, or if either file is not recognized as the appropriate file type, an error will occur.

**Size**

You can also specify a font size in points.

The default is \*FONTID. This indicates that the size of the font used will be the same as the font size implied or specified by the from-font id or from-font-size part of this parameter. You may also specify a particular font size in points that you want to use.

## CVTFNTRSC – Convert font resources

Parameter	CVTFNTRSC
Applies to commands:	CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSTMF

This parameter allows you to define your own mappings for fonts used in the spooled file that are identified by a font resource name. You may define mappings here in order to improve the appearance of your report in PDF, RTF or HTML format if **FONT(\*MAP)** alone does not give satisfactory results. Up to 100 mappings may be defined.

The default is **\*NONE**, which indicates that no user-definable font resource mappings are specified.

Other than **\*NONE**, all other selections for this parameter consist of two pairs of elements:

- **From resource**

consisting of:

- **Font resource name**
- **Font size**
- **Size units**

and

- **To font**

consisting of:

- **Face**
- **Size**

Where the specified font resource name appears in the spooled file, it will be mapped to the font typeface and point size specified here in the resultant PDF, RTF or HTML file.

CoolSpools notifies you of the font mappings that it has chosen by sending a message to the job log when you run the command. If you are not satisfied with the appearance of your spooled file in the stream file, consider using this parameter or the CVTFONTID parameter to define your own font mappings. Refer to the discussion of the **FONT** parameter above for further information about how CoolSpools assists you in identifying the font ids that need to be mapped.

### **From Resource**

#### **Font Resource Name**

Enter the name of a font resource referenced in the spooled file. This will usually be either: a font character set defined on the **FNTCHRSET** parameter of the **CRTPRTF** (Create Printer File) command or on the DDS **FNTCHRSET** keyword; or a coded font defined on the **CDEFNT** parameter of the **CRTPRTF** (Create Printer File) command or on the DDS **CDEFNT** keyword.

#### **Font Size**

Options are:

<b><u>*FONTNAME</u></b>	The font size is implied by the font resource name.
<b>Font_size</b>	The font size in points.

Some font resource objects (typically raster fonts) imply a specific font size, while others (typically outline fonts) do not. A point size is normally specified for use with a font resource through the use of the printer file DDS keyword FNTCHRSET with the \*POINTSIZ option.

### Size units

Options are:

<b><u>*POINTS</u></b>	The font size denotes the height of characters in points.
<b>*CPI</b>	The font size denotes the pitch (width) of characters in characters per inch.

### To Font

#### Face

This specifies how the font will be implemented inside the PDF and can be any of:

- a built-in PDF font
- a system i font resource object
- a TrueType or PostScript Type 1 font file located in the IFS

The following built-in typefaces are available for selection:

<b>*COURIER</b>	<b>Courier</b>
<b>*COURIERB</b>	<b>Courier Bold</b>
<b>*COURIERO</b>	<b>Courier Oblique</b>
<b>*COURIERBO</b>	<b>Courier Bold Oblique</b>
<b>*HELVETICA</b>	<b>Helvetica</b>
<b>*HELVB</b>	<b>Helvetica Bold</b>
<b>*HELVO</b>	<b>Helvetica Oblique</b>
<b>*HELVB</b>	<b>Helvetica Bold</b>
<b>*HELVBO</b>	<b>Helvetica Bold Oblique</b>
<b>*TIMES</b>	<b>Times Roman</b>
<b>*TIMESB</b>	<b>Times Roman Bold</b>
<b>*TIMESI</b>	<b>Times Roman Italic</b>
<b>*TIMESBI</b>	<b>Times Roman Bold Italic</b>
<b>*SYMBOL</b>	<b>Symbol</b>
<b>*DINGBATS</b>	<b>Zapf Dingbats</b>

### Example:

```
CVTSPLPDF      FROMFILE(SALES)...
                FONT(*MAP)
                CVTFNTRSC(C0S0CR10 *COURIER 12)
```

Here, the sales report is converted to PDF format. CoolSpools will attempt to map fonts, but the font named C0S0CR10 is explicitly mapped to Courier 12-point.

Alternatively, you can specify either a font resource object or a PostScript Type 1 font file.



Note that you still need to specify **FONT(\*EMBED)** if you want the font to be embedded in PDF. If you specify **FONT(\*MAP)**, CoolSpools will use the font specified on this element of the **CVTFNTRSC** parameter to select a suitable standard font to map to, but will still use a mapped standard font not an embedded font.

These parameters require an IFS path name. We expect to extend these parameters to support TrueType and OpenType fonts at some point in the future. If you are referencing a font resource object, you must therefore specify the object name in IFS format, i.e.

**/QSYS.LIB/library\_name.LIB/object\_name.FNTRSC**

**Example:**

**CVTSPLPDF...  
CVTFNTRSC(((C0S0CR10)  
(/QSYS.LIB/QFNTCPL.LIB/C0S0CR10.FNTRSC))) FONT(\*EMBED)**

Font resource C0S0CR10 will be implemented by embedding font resource object C0S0CR10 in library QFNTCPL

**Example:**

**CVTSPLPDF...  
CVTFONTID(((C0S0CR10)  
(/QIBM/PRODDATA/OS400/FONTS/PSFONTS/LATIN/COU.PFB')))  
FONT(\*EMBED)**

Font resource C0S0CR10 should be implemented by embedding the Postscript courier font supplied by IBM at the path given.

Please note that **two** files are required in order to successfully embed a PostScript font:

- i) A PostScript Type 1 font file. This normally has a file extension of .pfb (e.g. /QIBM/PRODDATA/OS400/FONTS/PSFONTS/LATIN/HEL.PFB)
- ii) A PostScript Type 1 font metrics file. This normally has a file extension of .afm (e.g. /QIBM/PRODDATA/OS400/FONTS/PSFONTS/LATIN/HEL.AFM)

Specify the name of the font file (.pfb extension) on this parameter. CoolSpools will attempt to locate the corresponding font metrics file (.afm extension) in the same location. If either file cannot be found, or if either file is not recognized as the appropriate file type, an error will occur.

### **Size**

You can also specify a font size in points.

The default is \*FONTID. This indicates that the size of the font used will be the same as the font size implied or specified by the from-font resource or from-font-size part of this parameter. You may also specify a particular font size in points that you want to use.

## DBCS - DBCS conversion options

Parameter	DBCS
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSTMF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML

The **DBCS** (DBCS conversion options) parameter allows you to control various options relating to the processing of DBCS (Double-Byte Character Set) data, i.e. data in languages such as Japanese, Chinese and Korean.

### DBCS Coded Font

The first element specifies the qualified name of the DBCS coded font to be used to implement DBCS text in the spooled file.

It is not normally necessary to specify a value on this parameter since the names of DBCS fonts to be used will be derived from instructions in the spooled file data stream.

If a DBCS spooled file has been created using a printer file which specifies IGCCDEFNT(\*SYSVAL), the name of the coded font will be obtained from the QIGCCDEFNT system value. However, if the spooled file has been transferred to a system (e.g. a non-DBCS system) which has the system value QIGCCDEFNT set to \*NONE, CoolSpools will be unable to identify the appropriate coded font to use. In those circumstances, you can use this parameter to specify the name of the font resource object (coded font) that should be used to display DBCS text in the spooled file.

Values are:

#### **\*SPLF**

CoolSpools determines the DBCS coded font name from the spooled file attributes. Where the spooled file refers to the QIGCCDEFNT system value, the coded font is taken from the system value. If QIGCCDEFNT is set to \*NONE, an error will be reported and you should specify a coded font name on this parameter.

#### **\*IGNORE**

Tells CoolSpools to ignore data that appears to be DBCS and treat it as SBCS.

#### **Coded\_font**

Specify the qualified name of the font resource object to be used. The object specified must be of object type \*FNTRSC with object attributes CDEFNT.

### DBCS coded font size

The second element specifies the DBCS font size.

It is not normally necessary to specify a value on this parameter since the font size of DBCS fonts to be used will be derived from instructions in the spooled file data stream.

However, the font size can be overridden using this parameter.

Values are:

**\*SPLF**

CoolSpools determines the DBCS font size from the spooled file attributes.

**Font\_size**

Specify the font size you wish to use in points.

**DBCS in non-DBCS splf?**

This element tells CoolSpools what to do if it encounters what appears to be DBCS data in a spooled file the attributes of which indicate that the spooled file is not capable of holding DBCS data.

**\*NO**

CoolSpools ignores the apparently DBCS data and treats it as SBCS.

**\*YES**

CoolSpools treats the data as DBCS.

## DELIMITERS - Delimited file options

Parameter	<b>DELIMITERS</b>
Applies to commands:	<b>CVTSPLDLM (see also CSV parameter of CVTSPLCSV)</b>
Dependent on:	<b>None</b>

Specifies options for creating delimited text files such as Comma Separated Variable files (CSVs) and Tab Separated Variable files (TSVs).

### Record delimiter

This element allows you to specify the characters to be used to indicate the end of a record in the delimited file.

Options are:

- \*CRLF** Carriage return and line feed. Both a carriage return (ASCII x'0D' or the equivalent in the CCSID selected) and a line feed (ASCII x'0A' or the equivalent in the CCSID selected) are used to denote the end of a record.
- \*CR** Just a carriage return (ASCII x'0D' or equivalent) is used.
- \*LF** Just a line feed (ASCII x'0A' or equivalent) is used.

### String delimiter

This element allows you to define the character that encloses string (alphanumeric) data in the delimited file that is to be created.

Either type the character to be used, or select one of the special values:

- \*DBLQUOTE** A double quote (") is used
- \*SGLQUOTE** A single quote (') is used
- \*NONE** No delimiter is used. Alphanumeric data is not enclosed by any special character.
- String\_delim** Type the delimiter character to be used.

The string delimiter will be output in the CCSID selected for the file.

### Field Delimiter

This element allows you to define the character that separates fields in the delimited file that is to be created.

Either type the character to be used, or select one of the special values:

- \*COMMA** A comma (,) is used
- \*TAB** A tab (ASCII x'09' or the equivalent in the CCSID selected) is used
- \*BLANK** A space or blank (ASCII x'20' or the equivalent in the CCSID selected) is used

<b>*SEMICOLON</b>	A semicolon (;) is used.
<b>*PIPE</b>	A pipe character ( ) is used.
<b>Field_delim</b>	Type the delimiter character to be used.

### Decimal point

Specify the character to represent a decimal point in the delimited file.

<b>*<u>SYSVAL</u></b>	The character implied by the system value QDECFMT is used.
<b>*JOB</b>	The character implied by the job attribute DECFMT is used.
<b>*PERIOD</b>	A period (.) is used.
<b>*COMMA</b>	A comma (,) is used.
<b>dec_point</b>	Specify the character to use as the decimal point.

### Date format

Specify the format in which to output dates in the delimited file.

<b>*<u>JOB</u></b>	The format implied by the job attribute DATFMT is used.
<b>*SYSVAL</b>	The format implied by the system value QDATFMT is used.
<b>*EXCEL</b>	Excel format (a number representing a day count since 1 <sup>st</sup> January 1901)
<b>*ISO</b>	YYYY-MM-DD
<b>*JIS</b>	YYYY-MM-DD
<b>*USA</b>	MM/DD/YYYY
<b>*EUR</b>	DD.MM.YYYY
<b>*YMD</b>	YYMMDD
<b>*YYMD</b>	YYYYMMDD
<b>*MDY</b>	MMDDYY
<b>*MDYY</b>	MMDDYYYY
<b>*DMY</b>	DDMMYY
<b>*DMYY</b>	DDMMYYYY
<b>*CYMD</b>	CYYMMDD
<b>*CMDY</b>	CMMDDYY
<b>*CDMY</b>	CDDMMYY
<b>*JUL</b>	YYDDD
<b>*LONGJUL</b>	YYYYDDD

### Date separator character

Specify the separator to use when outputting dates in the delimited file.

<b>*<u>JOB</u></b>	The character implied by the job attribute DATSEP is used.
<b>*SYSVAL</b>	The character implied by the system value QDATSEP is used.
<b>*SLASH</b>	/
<b>*HYPHEN</b>	-
<b>*PERIOD</b>	.
<b>*COMMA</b>	,
<b>*COLON</b>	:
<b>*BLANK</b>	A blank (space)
<b>separator</b>	Specify the character to use

### Trim blanks from char fields

Whether CoolSpools trims blanks from character fields.

Options are:

<b>*<u>BOTH</u></b>	Both leading and trailing blanks are trimmed from character fields before they are output to the file.
<b>*LEADING</b>	Leading blanks but not trailing blanks are trimmed.
<b>*TRAILING</b>	Trailing blanks but not leading blanks are trimmed.
<b>*NONE</b>	No blanks are trimmed.

## DFNSTYLES – Define styles

Parameter	<b>DFNSTYLES</b>
Description	<b>Defines styles which control the appearance of data on screen</b>
Applies to commands:	<b>CVTSPLXL, CVTSPLXML</b>
Dependent on:	<b>None</b>

Allows you to define the attributes of the \*NORMAL default style or to specify user-defined named styles. Styles control the appearance of data on screen. For example, they govern items such as:

- font attributes
- color
- numeric formatting

There are two ways in which to associate a style with a piece of data in your output file (e.g. a cell in an Excel spreadsheet or an element of an XML document):

### 1. Implicitly

By defining the style name the same as the name of a row group in your Database-to-Excel map or an element in your Database-to-XML map, you implicitly apply that style to the data in question. For example, a style called REPORT\_HEADING will be implicitly and automatically applied to an Excel row group called REPORT\_HEADING.

***Note that style names are case-sensitive because XML element names need to be case-sensitive and for this association of names to work, the names must match exactly in terms of case.***

### 2. Explicitly

Alternatively, use the APYSTYLES parameter to define the styles you wish to apply to different parts of the file you create.

The precise set of attributes that can be controlled varies depending on the format of the data being created as some attributes are not relevant to certain output formats.

CoolSpools Spool Converter styles defined on this parameter will translate into Excel user-defined styles if converting to Excel format and CSS styles if converting to HTML/XML.

## Style name

Each style is identified by means of a style name, which can be up to 50 characters in length and is case-sensitive (in order to allow matching to XML elements, the names of which conform to the rules for XML names).

You can define your own named styles by choosing a name that is helpful to you. There is a single pre-defined style name which has a special meaning:

**\*NORMAL**                      The default style.

If you specify \*NORMAL for the name of the style, the attributes you specify will become the default attributes for data in the spreadsheet or stylesheet.

If the \*NORMAL style is not defined, the default attributes assigned are as shown in the table below:

Attribute	*NORMAL style default
Locked (Excel only)	Yes
Hidden (Excel only)	No
Horizontal alignment	General
Indent	0
Vertical alignment	Top
Wrap text	No
Shrink to fit (Excel only)	No
Vertical alignment	Top
Row height	*AUTOFIT
Font name (Excel)	Arial
Font name (HTML & XML)	sans-serif
Font size in point (Excel)	10
Font size in point (XML)	12
Bold	No
Italic	No
Underlined	No
Text color	Black
Background color	White
Pattern color (Excel only)	*AUTO
Pattern style (Excel only)	*NONE
Border style (Excel)	*NONE
Border style (XML)	*INSET
Border width (XML only)	1



Border color	*AUTO
Number format type (Excel only)	*DFT
Decimal places (Excel only)	*FIELD
Thousands separator (Excel only)	*FMT
Currency symbol (Excel only)	*FMT
Negative numbers (Excel only)	*FMT
Custom number format (Excel only)	*NONE
Cell padding (XML only)	1
Additional style declaration (XHTML only)	*NONE
Display option (XML only)	*BLOCK

**Example:**

**CVTSPLXL**

...

**DFNSTYLES((HIGHLIGHT \*YES \*NO \*GENERAL \*NONE \*BOTTOM \*NO \*NO  
\*AUTOFIT \*ARIAL 12 \*YES \*NO \*NO \*YELLOW \*BLUE \*AUTO \*NONE \*THIN))  
APYSTYLES((BALANCE \*ANY \*ANY HIGHLIGHT))**

This code defines a new style called “highlight” that uses Arial bold 12-point yellow on blue and applies that style to the row group called “BALANCE”.

**Example:**

**CVTSPLXL**

...

**DFNSTYLES((\*NORMAL \*YES \*NO \*GENERAL \*NONE \*BOTTOM \*NO \*NO  
\*AUTOFIT \*COURIER 12 \*NO \*NO \*NO \*RED \*SILVER \*AUTO \*NONE \*THIN))**

This code redefines the predefined \*NORMAL style and so modifies the default attributes for data rows to use Courier 12-point red on silver and the default attributes for column headings to use Courier 14-point bold red on silver.

The various options that can be defined are as follows:

### **Locked**

(Excel only)

Whether cells to which this style is applied are locked when worksheet protection is in effect.

Options are:

**\*YES**

(Default). When the worksheet is protected, the cell will be locked (protected).

**\*NO**

When the worksheet is protected, the cell will remain unlocked.

### Example:

CVTSPLXL

...

```
DFNSTYLES(  
    (*NORMAL *NO)  
    (HEADER_ROW *YES *NO *GENERAL *NO *BOTTOM *ARIAL 10 *NO *NO  
    *NO *WHITE *BLUE)  
    (LOCKED *YES *NO *GENERAL *NO *BOTTOM *ARIAL 10 *NO *NO *NO  
    *WHITE *BLUE))  
APYSTYLES((DETAIL_ROW 1 A LOCKED))  
XLSPROTECT(*YES)
```

This code redefines the \*NORMAL style such that, by default, data cells are not locked when worksheet protection is in effect. It then defines a new style called HEADER\_ROW which specifies styling for a row group called HEADER\_ROW. Finally, it defines a style called "LOCKED" such that cells to which this style is applied are locked and also appear white on blue. The style LOCKED is applied to column A of the first row generated by the row group DETAIL\_ROW. Worksheet protection is switched on.

The overall effect is to create a worksheet where the user can make changes to the data apart from the headings and the first cell of the detail rows, which appear white on blue rather than black on white to emphasize the fact they are different.

### Hidden

(Excel only)

Allows you to indicate that a column should be hidden. This might be useful if you do not wish the column to appear but want it to be available for calculations.

Options are:

<b>*NO</b>	(Default). The column is not hidden.
<b>*YES</b>	The columns will be hidden

### Horizontal alignment

Controls the horizontal alignment of data in a cell.

Options are:

<b><u>*GENERAL</u></b>	(Default). Character data is left-aligned. Numeric data and dates are right-aligned. In relation to header text, the alignment is dictated by the nature of the data in the column, not the header text, i.e. headings for columns of character data will align to the left and headings for numeric columns and date columns will align to the right.
<b>*LEFT</b>	Left-aligned.
<b>*RIGHT</b>	Right-aligned.
<b>*CENTER</b>	Center-aligned
<b>*FILL</b>	(Excel only) Fill. Repeats the data in the cell across the entire width of the column.
<b>*JUSTIFY</b>	Forces data to fill the entire width of the column, wrapping text to additional lines, if necessary.

**\*DISTRIBUTED** (Excel only) Distributed. Available only in Excel 2002 and above. It results in the cell contents being distributed across the width of the cell, to line up with both the left and right side.

## Indent

Sets the text indent level. The effects of this are somewhat different between Excel and HTML/XML.

Options are:

<b>*NONE</b>	(Default). No indent is applied.
<b>0-15</b>	(Excel) Sets the indentation level. Each indentation level is equivalent to 3 spaces. All text affected is indented to the same extent, i.e. where text wraps to more than one line, it is all indented to the same point.
<b>0-99</b>	(HTML/XML). Sets the text-indent property in ems (the width of an em is equivalent to the point size of the font). The first line of the text only is indented.

## Vertical alignment

Controls the vertical alignment of data in a cell.

Options are:

<b>*BOTTOM</b>	(Default). Information is aligned at the bottom of the cell.
<b>*TOP</b>	Information is aligned at the top of the cell.
<b>*CENTER</b>	information is aligned in the center of the cell.
<b>*JUSTIFY</b>	Text is spread evenly vertically across the height of the cell.
<b>*DISTRIBUTED</b>	(Excel only) Text is spread evenly between the top of the cell and the bottom. Effectively, blank space is placed between each line so that the complete cell is filled.

## Wrap text

Controls whether text wraps in cells.

Options are:

<b>*NO</b>	(Default). Text does not wrap in the cell. If the text does not fit in the column width, it is truncated.
<b>*YES</b>	Text wraps in the cell. If the text does not fit in the column width, it will flow on to multiple lines.

## Shrink to fit

(Excel only)

Determines whether the cell contents are shrunk to fit the available column width by reducing the font size.

Options are:

**\*NO**  
**\*YES**

(Default). Text is not shrunk to fit.  
Text is fitted to the available column width by reducing the font size, as required.

## Row height

Sets the height of rows.

Note that this attribute is only effective if set on a style which is applied to an entire row.

Options are:

**\*AUTOFIT**

(Default). The height of rows is automatically set by Excel or your browser (XML) based on the font size.

**0-409**

(Excel) Specify the row height in points (72 points = 1 inch)

**0-32767**

(XML) Specify the row height in points (72 points = 1 inch)

## Font name

Specifies the name of the font to be used.

Note that CoolSpools Spool Converter cannot validate whether the font name you have specified is valid or whether it will be available when the file is opened. If the font name is typed incorrectly or if the font is not available when the file is opened, Excel or your browser will substitute a different font.

Note also that when the font you use in Excel is not one of the “well known” fonts (Arial, Courier New or Times New Roman), CoolSpools Spool Converter may not be able to calculate column widths correctly because it has no access to the font metrics on which those calculations depend.

Excel options are:

**\*ARIAL**

(Default). Arial

**\*COURIER**

Courier New

**\*TIMES**

Times New Roman

**font\_name**

Specify the name of the font to use

HTML/XML options are:

**\*SANS**

(Default) Sans-serif font family.

**\*SERIF**

Serif font family

**\*MONO**

Monospaced font family.

**\*ARIAL**

(Default). Arial

**\*COURIER**

Courier New

**\*TIMES**

Times New Roman

**font\_name**

Specify the name of the font to use

## Font size in points

The point size of the font to use. The default is 10 for Excel and 12 for HTML/XML.

## Bold

Whether the font is bold or not. Note that setting this attribute will only result in a bold font if a suitable bold version of the font is available or if the normal font can be adapted.

Options are:

<u>*NO</u>	(Default). Normal font
*YES	Bold font.

### Italic

Whether the font is italic or not. Note that setting this attribute will only result in an italic font if a suitable italic version of the font is available or if the normal font can be adapted.

Options are:

<u>*NO</u>	(Default). Normal font
*YES	Italic font.

### Underlined

Whether the font is underlined or not and, if it is, the style of underlining.

Excel options are:

<u>*NO</u>	(Default). No underlining
*SINGLE	Single underlining
*DOUBLE	Double underlining
*SGLACC	Single accounting underlining
*DBLACC	Double accounting underlining

XML options are:

<u>*NO</u>	(Default). No underlining
*YES	Single underlining

### Text color

Determines the color of text in cells.

The Excel default is:

<u>*AUTO</u>	The Excel default text color (usually black)
--------------	--

Alternatively, you can use one of the 56 built-in Excel colors listed below with their RGB coding.

*BLACK	000000
*WHITE	FFFFFF
*RED	FF0000
*BRIGHTGREEN	00FF00
*BLUE	0000FF
*YELLOW	FFFF00
*PINK	FF00FF
*TURQUOISE	00FFFF
*DARKRED	800000

*GREEN	008000
*DARKBLUE	000080
*DARKYELLOW	808000
*VIOLET	800080
*TEAL	008080
*GRAY25	C0C0C0
*GRAY50	808080
*MAUVE	9999FF
*PLUM	993366
*YELLOWWHITE	FFFFCC
*LIGHTTURQUOISE	CCFFFF
*DARKPINK	660066
*BLUSH	FF8080
*MEDIUMBLUE	0066CC
*PALEMAUVE	CCCCFF
*SKYBLUE	00CCFF
*LIGHTGREEN	CCFFCC
*LIGHTYELLOW	FFFF99
*PALEBLUE	99CCFF
*ROSE	FF99CC
*LAVENDER	CC99FF
*TAN	FFCC99
*LIGHTBLUE	3366FF
*AQUA	33CCCC
*LIME	99CC00
*GOLD	FFCC00
*LIGHTORANGE	FF9900
*ORANGE	FF6600
*BLUEGRAY	666699
*GRAY40	969696
*DARKTEAL	003366
*SEAGREEN	339966
*DARKGREEN	003300
*OLIVEGREEN	333300
*BROWN	993300
*INDIGO	333399
*GRAY80	333333

When converting to \*XLSX format, you can also optionally specify your own RGB color code in the form of six hexadecimal digits (similar to the codes shown in the table above). Please note that this option is not supported when converting to \*XLS (BIFF8) format.

The XML default is:

<b><u>*BLACK</u></b>	Black
----------------------	-------

Alternatively, you can use one of the HTML colors listed below with their RGB coding.

<b>*ALICEBLUE</b>	<b>F0F8FF</b>
-------------------	---------------

*ANTIQUEWHITE	FAEBD7
*AQUA	00FFFF
*AQUAMARINE	7FFFD4
*AZURE	F0FFFF
*BEIGE	F5F5DC
*BISQUE	FFE4C4
*BLACK	000000
*BLANCHEDALMOND	FFEBCD
*BLUE	0000FF
*BLUEVIOLET	8A2BE2
*BROWN	A52A2A
*BURLYWOOD	DEB887
*CADETBBLUE	5F9EA0
*CHARTREUSE	7FFF00
*CHOCOLATE	D2691E
*CORAL	FF7F50
*CORNFLOWERBLUE	6495ED
*CORN SILK	FFF8DC
*CRIMSON	DC143C
*CYAN	00FFFF
*DARKBLUE	00008B
*DARKCYAN	008B8B
*DARKGOLDENROD	B8860B
*DARKGRAY	A9A9A9
*DARKGREY	A9A9A9
*DARKGREEN	006400
*DARKKHAKI	BDB76B
*DARKMAGENTA	8B008B
*DARKOLIVEGREEN	556B2F
*DARKORANGE	FF8C00
*DARKORCHID	9932CC
*DARKRED	8B0000
*DARKSALMON	E9967A
*DARKSEAGREEN	8FBC8F
*DARKSLATEBLUE	483D8B
*DARKSLATEGRAY	2F4F4F
*DARKSLATEGREY	2F4F4F
*DARKTURQUOISE	00CED1
*DARKVIOLET	9400D3
*DEEPPINK	FF1493
*DEEPSKYBLUE	00BFFF
*DIMGRAY	696969
*DIMGREY	696969
*DODGERBLUE	1E90FF
*FELDSPAR	D19275
*FIREBRICK	B22222
*FLORALWHITE	FFFAF0
*FORESTGREEN	228B22
*FUCHSIA	FF00FF
*GAINSBORO	DCDCDC
*GHOSTWHITE	F8F8FF

*GOLD	FFD700
*GOLDENROD	DAA520
*GRAY	808080
*GREY	808080
*GREEN	008000
*GREENYELLOW	ADFF2F
*HONEYDEW	F0FFF0
*HOTPINK	FF69B4
*INDIANRED	CD5C5C
*INDIGO	4B0082
*IVORY	FFFFFF0
*KHAKI	F0E68C
*LAVENDER	E6E6FA
*LAVENDERBLUSH	FFF0F5
*LAWNGREEN	7CFC00
*LEMONCHIFFON	FFFACD
*LIGHTBLUE	ADD8E6
*LIGHTCORAL	F08080
*LIGHTCYAN	E0FFFF
*LIGHTGOLDENROD	FAFAD2
*LIGHTGRAY	D3D3D3
*LIGHTGREY	D3D3D3
*LIGHTGREEN	90EE90
*LIGHTPINK	FFB6C1
*LIGHTSALMON	FFA07A
*LIGHTSEAGREEN	20B2AA
*LIGHTSKYBLUE	87CEFA
*LIGHTSLATEBLUE	8470FF
*LIGHTSLATEGRAY	778899
*LIGHTSLATEGREY	778899
*LIGHTSTEELBLUE	B0C4DE
*LIGHTYELLOW	FFFFE0
*LIME	00FF00
*LIMEGREEN	32CD32
*LINEN	FAF0E6
*MAGENTA	FF00FF
*MAROON	800000
*MEDIUMAQUAMARINE	66CDAA
*MEDIUMBLUE	0000CD
*MEDIUMORCHID	BA55D3
*MEDIUMPURPLE	9370D8
*MEDIUMSEAGREEN	3CB371
*MEDIUMSLATEBLUE	7B68EE
*MEDIUMSPRINGGREEN	00FA9A
*MEDIUMTURQUOISE	48D1CC
*MEDIUMVIOLETRED	C71585
*MIDNIGHTBLUE	191970
*MINTCREAM	F5FFFA
*MISTYROSE	FFE4E1
*MOCCASIN	FFE4B5
*NAVAJOWHITE	FFDEAD



*NAVY	000080
*OLDLACE	FDF5E6
*OLIVE	808000
*OLIVEDRAB	6B8E23
*ORANGE	FFA500
*ORANGERED	FF4500
*ORCHID	DA70D6
*PALEBLUE	ADD8E6
*PALEBROWN	CD853F
*PALECYAN	E0FFFF
*PALEGOLDENROD	EEE8AA
*PALEGRAY	D3D3D3
*PALEGREY	D3D3D3
*PALEGREEN	98FB98
*PALEMAG	DDA0DD
*PALETURQUOISE	AFEEEE
*PALEVIOLETRED	D87093
*PALEYLW	FFFFE0
*PAPAYAWHIP	FFEDD5
*PEACHPUFF	FFDAB9
*PERU	CD853F
*PINK	FFC0CB
*PLUM	DDA0DD
*POWDERBLUE	B0E0E6
*PURPLE	800080
*RED	FF0000
*ROSYBROWN	BC8F8F
*ROYALBLUE	4169E1
*SADDLEBROWN	8B4513
*SALMON	FA8072
*SANDYBROWN	F4A460
*SEAGREEN	2E8B57
*SEASHELL	FFF5EE
*SIENNA	A0522D
*SILVER	C0C0C0
*SKYBLUE	87CEEB
*SLATEBLUE	6A5ACD
*SLATEGRAY	708090
*SLATEGREY	708090
*SNOW	FFFAFA
*SPRINGGREEN	00FF7F
*STEELBLUE	4682B4
*TAN	D2B48C
*TEAL	008080
*THISTLE	D8BFD8
*TOMATO	FF6347
*TURQUOISE	40E0D0
*VIOLET	EE82EE
*VIOLETRED	D02090
*WHEAT	F5DEB3
*WHITE	FFFFFF

<b>*WHITESMOKE</b>	<b>F5F5F5</b>
<b>*YELLOW</b>	<b>FFFF00</b>
<b>*YELLOWGREEN</b>	<b>9ACD32</b>

You can also optionally specify your own RGB color code in the form of six hexadecimal digits (similar to the codes shown in the table above).

### Background color

Determines the color of the background of a cell.

The Excel default is:

<b><u>*AUTO</u></b>	The Excel default background color (usually white)
---------------------	--

Alternatively, you can use the same Excel options as listed for text color above.

The HTML/XML default is:

<b><u>*WHITE</u></b>	White
----------------------	-------

Alternatively, you can use the HTML color options as listed for text color above.

### Pattern color

(Excel only)

Determines the color of the any pattern applied to a cell.

The Excel default is:

<b><u>*AUTO</u></b>	The Excel default pattern color (usually black)
---------------------	---

Alternatively, you can use the same Excel options as listed for text color above.

### Pattern style

(Excel only)

Determines the style of any pattern applied to a cell.

The default is:

<b><u>*NONE</u></b>	No pattern
---------------------	------------

The available pattern options are the following names, which correspond to Excel's builtin patterns:

**\*SOLID**  
**\*GRAY75**  
**\*GRAY50**  
**\*GRAY25**  
**\*GRAY12.5**  
**\*GRAY6.25**  
**\*HRZSTRIPE**  
**\*VRTSTRIPE**  
**\*REVERSEDIAGSTRIPE**  
**\*DIAGSTRIPE**  
**\*DIAGCROSSHATCH**

\*THICKDIAGCROSSHATCH  
\*THINHRZSTRIPE  
\*THINVRTSTRIPE  
\*THINREVERSEDIAGSTRIPE  
\*THINDIAGSTRIPE  
\*THINHRZCROSSHATCH  
\*THINDIAGCROSSHATCH

### Border style (Excel)

Determines the style of the border around a cell.

Note that the DFNSTYLES parameter does not support the setting of different attributes for the top, bottom, left and right borders. The attributes defined here apply to all 4 borders. If you want to define different attributes for the different borders, you must use a user-defined named style (see Base Option manual, CRTSTLDFN Create Style Definition command)

The Excel default is:

**\*NONE**                                      No border

Other Excel options are the following list of names corresponding to Excel's builtin border styles:

\*THIN  
\*MEDIUM  
\*DASHED  
\*DOTTED  
\*THICK  
\*DOUBLE  
\*HAIR

The HTML/XML options correspond to the CSS border style options:

**\*INSET**                                      (Default) CSS inset border style  
\*DASHED  
\*DOTTED  
\*DOUBLE  
\*GROOVE  
\*HIDDEN  
\*OUTSET  
\*RIDGE  
\*SOLID

### Border width

(XML only)

The width of the cell border in pixels.

Note that the DFNSTYLES parameter does not support the setting of different attributes for the top, bottom, left and right borders. The attributes defined here apply to all 4 borders. If you want to define different attributes for the different borders, you

must use a user-defined named style (see Base Option manual, CRTSTLDFN Create Style Definition command)

### Border color

The color of the border. Options are the same as for text color above.

Note that the DFNSTYLES parameter does not support the setting of different attributes for the top, bottom, left and right borders. The attributes defined here apply to all 4 borders. If you want to define different attributes for the different borders, you must use a user-defined named style (see Base Option manual, CRTSTLDFN Create Style Definition command)

### Number format type

(Excel only)

Sets the category of number formatting applied to numbers in cells to which this style relates. The following options allow you to modify or override aspects of the default formatting determined by your choice for this parameter element.

Options are:

<b>*NONE</b>	CoolSpools Spool Converter will not apply any numeric formatting.
<b>*DFT</b>	CoolSpools Spool Converter will interpret the appearance of the field in the report and derive a format string from any editing that seems to be apparent.
<b>*GENERAL</b>	Ignore any editing associated with the field and format numeric data with general numbers in them.
<b>*FIXED</b>	Ignore any editing associated with the field and format numeric data with a fixed number of decimal places.
<b>*CURRENCY</b>	Ignore any editing associated with the field and format numeric data as a currency amount.
<b>*ACCOUNTING</b>	Ignore any editing associated with the field and format numeric data as an accounting value. The Accounting category is the same as the Currency category, except it will align currency symbols and decimal points.
<b>*DATE</b>	Ignore any formatting associated with the field and format it as a date. If the field does not contain a valid date, it will be formatted according to any editing associated with the field.
<b>*TIME</b>	Ignore any formatting associated with the field and format it as a time or date/time. If the field does not contain a valid time or timestamp, it will be formatted according to any editing associated with the field.

<b>*PERCENT</b>	Ignore any editing associated with the field, multiply the value by 100 and format numeric data as a percentage.
<b>*SCIENTIFIC</b>	Ignore any editing associated with the field, and format numeric data in scientific notation.
<b>*TEXT</b>	Ignore any editing associated with the field, and format numeric data as text.
<b>*CUSTOM</b>	Apply a custom number format specified on the custom number format element below.

## Decimal places

(Excel only)

Where a numeric format (other than \*DFT) that can include decimal places was specified on the number format type parameter, this parameter element determines the number of decimal places displayed.

Options are:

<b>*<u>FIELD</u></b>	The number of decimal places a numeric field displays in the report is retained.
<b>dec_places</b>	Specify the number of decimal places

## Thousands separator

(Excel only)

Where a numeric format (other than \*DFT) that can include thousands separators was specified on the number format type parameter, this parameter element determines whether thousands separators actually appear.

Options are:

<b>*<u>FMT</u></b>	Whether thousands separators appear depends on the number format type selected. Accounting and currency formatting will include thousands separators but other types will not.
<b>*YES</b>	Include thousands separators in the number format irrespective of the fact that the number format type specified does not normally include them. For example you can format percentage values with thousands separators using this option.
<b>*NO</b>	Do not include thousands separators in the number format irrespective of the fact that the number format type specified does normally include them. For example, you can format currency values without thousands separators using this option.

## Currency symbol

(Excel only)

Where a numeric format (other than \*DFT) that can include a currency symbol was specified on the number format type parameter, this parameter element determines whether a currency symbol actually appears and what that symbol should be.

Options are:

<b>*<u>FMT</u></b>	Whether a currency symbol appears depends on the number format type selected. Accounting and currency formatting will include a currency symbol but other types will not. The currency symbol will be derived from the system value QCURSYM.
<b>*SYSVAL</b>	Include a currency symbol in the number format irrespective of the fact that the number format type specified does not normally include one. The currency symbol will be derived from the system value QCURSYM.
<b>*NO</b>	Do not include a currency symbol in the number format irrespective of the fact that the number format type specified does normally include one. You can use this option to display a currency value with no currency symbol.
<b>currency_symbol</b>	Include a currency symbol in all numbers. The currency symbol will be the one specified here.

## Negative numbers

(Excel only)

Overrides the way in which negative numbers are displayed.

Options are:

<b>*<u>FMT</u></b>	The format of negative numbers is determined by the option specified for the number format type.
<b>*LEADING</b>	A leading minus sign is displayed.
<b>*TRAILING</b>	A trailing minus sign is displayed.
<b>*PARENTHESES</b>	Negative numbers appear in parentheses.
<b>*RED</b>	Negative numbers appear in red.
<b>*REDL</b>	Negative numbers appear in red with a leading minus sign.
<b>*REDT</b>	Negative numbers appear in red with a trailing minus sign.
<b>*REDP</b>	Negative numbers appear in red and in parentheses.

## Zero balances

(Excel only)

Determines whether zero values are displayed or not.

Options are:

<b>*<u>FMT</u></b>	Whether zero values are displayed, and how, is dependent on the option specified for the number format type.
<b>*YES</b>	Zero values are shown as zeros.
<b>*NO</b>	Zero values appear as blanks (empty cell).

### Custom number format

(Excel only)

Specify a custom number format. \*CUSTOM must be specified for the number format type element above.

Options are:

<b>*<u>NONE</u></b>	No custom number format is defined.
<b>number_format</b>	Specify the custom Excel number format to use.

### Cell padding

(HTML only)

The padding to apply to the cell, in pixels.

### Additional style declaration

(HTML only)

A free-format, unvalidated string of text which will be appended to the style declaration generated by the previous elements. This option enables you to specify additional CSS formatting not available from this parameter. However, you must ensure that the text you enter is a valid portion of a CSS style declaration.

For example, specifying '**font-variant: small-caps**' would cause the text to appear in small capitals.

### Display option (XML only)

Sets the CSS display style.

Options are:

<b>*<u>BLOCK</u></b>	(Default). Takes up the full width available, with a new line before and after.
<b>*INLINE</b>	Takes up only as much width as it needs, and does not force new lines

## ***DTACPR – Data compression***

Parameter	<b>DTACPR</b>
Applies to commands:	<b>CVTSPLSAV, SAVSPLF</b>
Dependent on:	<b>None</b>

This parameter allows you to specify the data compression level to be used when saving a spooled file as a compressed stream file archive with the CVTSPLSAV or SAVSPLF commands.

Data compression is a trade-off between file size and the time taken to create the file. The higher the compression ratio that is attempted, the longer the data will take to compress. The options below enable you to select whether you want a high compression ratio (giving the smallest archive files but taking longer to create) or the fastest conversion time (producing larger archive files but running more quickly).

Options are:

<b><u>*OPT</u></b>	Optimum. The data is compressed using a factor which provides a good degree of data compression while not taking unduly long to compress.
<b>*NONE</b>	The archive file is not compressed. The resultant archive files will be significantly larger than if data compression was applied, but will take less time to create.
<b>*MAX</b>	The maximum possible level of data compression is applied. The files will be as small as possible, but will take the longest time to create.
<b>*HIGHER</b>	A compression ratio higher than *HIGH but less than *MAX.
<b>*HIGH</b>	A compression ratio higher than *OPT but less than *HIGHER.
<b>*FAST</b>	A compression ratio less than *OPT but higher than *FASTER.
<b>*FASTER</b>	A compression ratio less than *FAST but higher than *FASTEST.
<b>*FASTEST</b>	The lowest and therefore fastest level of data compression.



## EMAIL – Email the output?

Parameter	EMAIL
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	None

This parameter lets you tell CoolSpools to email the output as an attachment.

Please note that this facility is only available if you the CoolSpools Email (product option 2) installed and licensed or on trial.

Options are:

**\*NO**

The output is not emailed automatically as part of running this command. You are still able to email the output separately, either by running a subsequent command or program or from a CoolSpools exit program.

**\*YES**

The output from this command will be emailed as an attachment or attachments according to the information you specify on the other email-related parameters. Every time a new output file is created, it will be emailed as a single attachment to the recipients specified.

**\*ONE**

The output from this command will be emailed as an attachment or attachments according to the information you specify on the other email-related parameters. All output files created will be emailed together as attachments at the end of the conversion run.

If you specify EMAIL(\*YES) or EMAIL(\*ONE) and you are splitting a spooled file into multiple stream files and the command you are running creates several stream files as a result, each stream file will be emailed to the all of the recipients you list on the EMAILTO parameter.

If each stream file that is created needs to go to different recipients, there are several methods available to you for supplying the email address or addresses to use:

- If the email address to be used is in the spooled file, you can extract it using [CoolSpools variables](#).
- Use the EMAILTO(\*EMAILFILE) option and related EMAILFILE parameter to tell CoolSpools to look up the email address(es) to be used in a specified file. [CoolSpools variables](#) can be used to extract data from the spooled file at run time to be used as keys to read the file. For example, you might take the customer number from the spooled file and use it to read a customer file to obtain the email address(es) for a particular invoice.

- Use the EMAILTO(\*EMAILSQL) option and related EMAILSQL parameter to tell CoolSpools to look up the email address(es) to be used by running an SQL statement. [CoolSpools variables](#) can be used to extract data from the spooled file at run time and replace parameter markers in the SQL statement. For example, you might take the invoice number from the spooled file and run a piece of SQL to join the invoice file to the customer file to obtain the email address(es) for a particular invoice.
- Specify EMAILTO(\*EXITPGM) and then use an exit program to define the recipients for each stream at run time by generating CS\_EMT01 structures. Refer to the CoolSpools Programmers Guide for further information on writing exit programs. Same source code is supplied in source file COOLSPV6R1/CS\_SRCFILE.

See the EMAILTO parameter below for further information.

You can also use the following additional methods of

### **Example 1**

A 100-page spooled file is split into 5 20-page PDF files. Two recipients are listed on the EMAILTO parameter and EMAIL(\*YES) is specified. The “Send multiple messages” element of the EMAILOPT parameter is set to \*YES.

Each recipient will receive 5 emails with one attachment per email.

Because “Send multiple messages” is \*YES, a separate message will be sent to each recipient, so 10 email message will be sent in total.

### **Example 2**

A 100-page spooled file is split into 5 20-page PDF files. Two recipients are listed on the EMAILTO parameter and EMAIL(\*ONE) is specified. The “Send multiple messages” element of the EMAILOPT parameter is set to \*NO.

Each recipient will receive 1 email with five attachments per email.

Because “Send multiple messages” is \*NO, only one email message will be sent, with two recipients specified.

## **EMAILFILE – Email address file**

Parameter	<b>EMAILFILE</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF</b>
Dependent on:	<b>EMAIL(*YES) EMAILTO(*EMAILFILE)</b>

This parameter allows you to specify that email addresses to which the email will be sent should be read at run time from a file.

You can either read all records in the file or read just selected records using keys that you supply. A maximum of 100 email addresses can be returned. If more than 100 email addresses are selected, only the first 100 are used.

The key values can be [CoolSpools variables](#). For example, you might extract a piece of data from the spooled file such as a customer number at run time and use that as a key to read email addresses from the file.

### **File name**

Specify the fully qualified name of the file from which email address information will be read.

The following special values may be specified for the library name:

- |                |  |
|----------------|--|
| <b>*LIBL</b>   | The file is located using the library list of the job. |
| <b>*CURLIB</b> | The file is located in the current library.            |

### **Email address field name**

Specify the name of the field in the above file that contains the email address to be used. The field must be either fixed or variable length character or graphic data, i.e. capable of holding an email address.

### **Email name field name**

Specify the name of the field in the above file that contains the recipient name to be used, or \*NONE to send the email with just an email address and no name. The field must be either fixed or variable length character or graphic data, i.e. capable of holding a name.

### **Email type field name**

Specify the name of the field in the above file that contains the recipient type to be used. The type must be either fixed or variable length character or graphic data and must contain one of the following values:

- \*PRI Primary recipient
- \*CC cc: recipient
- \*BCC bcc: recipient

Specify \*NONE if there is no such field. The type will default to “primary recipient”.

## Key fields

Specify up to 10 sets of key field information that will be used to read the file at run time.

There is a single value:

<b>*NONE</b>	No key fields are used. All records will be read from the specified file.
--------------	---

Alternatively, supply between 1 and 10 sets of key information in the following form.

## Field name

The name of the key field in the file specified above.

## Comparison

The test to apply.

Options are:

<b>*EQ</b>	Tests for the field being equal to the value specified below.
<b>*NE</b>	Tests for the field being not equal to the value specified below.
<b>*GT</b>	Tests for the field being greater than the value specified below.
<b>*LT</b>	Tests for the field being less than the value specified below.
<b>*GE</b>	Tests for the field being greater than or equal to the value specified below.
<b>*LE</b>	Tests for the field being less than or equal to the value specified below.
<b>*LIKE</b>	Tests for the field being like the value specified below. The value supplied can contain SQL-style generics using %.
<b>*NOTLIKE</b>	Tests for the field being not like the value specified below. The value supplied can contain SQL-style generics using %.
<b>*IN</b>	Tests for the field being in a list of values. The value supplied must be an SQL-style list enclosed in parentheses.
<b>*NOTIN</b>	Tests for the field being not in a list of values. The value supplied must be an SQL-style list enclosed in parentheses.

## Value or variable

The value with which the field is compared at run time. This can be either a constant or a [CoolSpools variable](#) name.

Do not code apostrophes or quotes around character values.

If a CoolSpools variable name is used, the comparison is carried out against the current value of that variable. For example, if the CoolSpools variable <:EXITPGMPOS1:> were specified for this parameter element, the key field would be compared to the current value of the item defined on the first element of the EXITPGMPOS parameter, which might be the invoice number or customer number in the current section of the spooled file.

**Example:**

```
CVTSPLPDF  
FROMFILE(STATEMENTS)  
EMAIL(*YES)  
EMAILTO(*EMAILFILE)  
EMAILFILE(CSTMST EMAIL CUSNAM *NONE ((CUSTNO *EQ <:CUST_NBR:>)))  
SPLIT(*POS)  
SPLITPOS((5 9 7))  
EXITPGM(*VAR)  
EXITPGMPRM(*POS)  
EXITPGMPOS((1 5 9 7 CUST_NBR))
```

Here, a spooled file called STATEMENTS is being converted to PDF and emailed. The spooled file contains a batch of customer account statements, and each separate statement needs to be emailed to the appropriate customer, which is different in each case.

The spooled file contains the customer number on line 5, position 9 for 7 characters, and a new PDF is started every time this value changes. The same value is also extracted from page 1 of the customer statement at run time by means of the EXITPGMPOS parameter, and given the user-defined name CUST\_NBR.

The EMAILFILE parameter specifies that the email addresses to be used are in customer file CSTMST and that the email address is held in field EMAIL and the name is field CUSNAM. There is no type field, so all emails will be sent as primary recipient (To:) emails rather than cc: or bcc: emails.

The correct customer record is read from the file at run time by using the field CUSTNO in file CSTMST as the key. The key value used is supplied in the form of a CoolSpools variable, namely the customer number item defined on the EXITPGMPOS parameter and extracted from page 1 of the statement.

**Example:**

```
CVTSPLPDF  
...  
EMAIL(*YES)  
EMAILTO(*EMAILFILE)  
EMAILFILE(CSTMST EMAIL CUSNAM *NONE ((CUSTNO *EQ ABC1234)))
```

Here, a single PDF will be created and emailed to the email address or addresses selected by reading records from CSTMST where the customer number equals ABC1234. Note that the value ABC1234 is not enclosed in quotes or apostrophes.

## EMAILFROM - Email sender information

Parameter	EMAILFROM
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	EMAIL(*YES)

This parameter allows you to specify the sender of the email and the email address to which a response should be sent.

The default value is **\*CURRENT**, which means that CoolSpools Email will try to retrieve the email address of the user sending the email from the System Distribution Directory. If no email address is defined for the user in the System Distribution Directory, you will need to enter the values you wish to use manually.

There are 2 elements to this parameter:

- **Email address**
- **Name.**

### Email address

This is where you enter the email address of the sender.

Note that while CoolSpools Email will check that the email address you enter conforms to the rules for valid email addresses, it is not possible to validate that the email address that you enter is correct or that any reply sent to the message will be deliverable.

For example, [sales.ariadnesoftware.co.uk](mailto:sales.ariadnesoftware.co.uk) is not a valid email address (since it does not contain an @ sign), and CoolSpools Email will reject it. However, [sales@ariadnesoftware.org.uk](mailto:sales@ariadnesoftware.org.uk) is a valid email address and CoolSpools Email will allow it, but it is not ariadne's correct email address (it should be [sales@ariadnesoftware.co.uk](mailto:sales@ariadnesoftware.co.uk)) and any reply sent to this email address will not be received.

CoolSpools variables may be specified on this parameter element.

### Name

If you would like your email message to display a sender's name rather than the sender's email address when it is delivered, enter the name here.

The default value is **\*NONE**, i.e. no name is provided and the email address will appear as the sender instead.

CoolSpools variables may be specified on this parameter element.

For example, if you specify:

**EMAILFROM((Sales@ariadnesoftware.co.uk \*NONE))**

when the message is received, the **From:** attribute will be shown as:

**From: Sales@ariadnesoftware.co.uk**

However, if you specify:

**EMAILFROM((Sales@ariadnesoftware.co.uk 'ariadne sales'))**

when the message is received, the **From:** attribute will be shown as:

**From: ariadne sales**

## EMAILMSG – Email message

Parameter	EMAILMSG
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	EMAIL(*YES)

This parameter allows you to specify the text of an email message to be sent with the attachment.

The message can be sent in plain text, HTML or alternate plain text/HTML formats.

You also have the option to base the message on the contents of a stream file. If you are running CVTSPLHTML, that stream file can be the HTML file created by the command.

### Message text or path name

This element has two possible uses:

- If **Text or path name specified** below is \*MSG, then this element represents the text of the message to be sent.
- If **Text or path name specified** below is \*STMG, then this element represents the path to a file containing the text of the message to be sent.

Up to 4096 characters of free-format text can be entered here. However, the command prompter limits this to 512 characters if F4 is pressed.

CoolSpools variables may be specified on this parameter element.

When received, the message will be displayed exactly as it is entered, with the following exceptions:

- If you want to force a line break, enter <br>. Even if the message is sent in plain text format, this HTML control will be interpreted and converted to a hard line break (carriage return-line feed sequence).
- Other HTML controls may be entered, but will only be interpreted as HTML controls if the message is sent and delivered in HTML format.

If you are running CVTSPLHTML and wish to use the file that is being created as the stream file to supply the message text, specify \*MSGTXT for the “Attach or embed?” element of the EMAILOPT parameter and specify \*TOSTMF on this parameter element.

### Message format

This is where you specify the format in which the message is sent.

Options are:

#### \*BOTH

The message is sent in alternate plain text/HTML format. This means that two versions of the message text will be sent: a plain text copy and an HTML copy. If the email client software used to receive the message can handle HTML messages,



the HTML version will be shown, otherwise the plain text copy will be shown.

**\*TEXT**

The message is sent in plain text format. The only HTML control which is interpreted is <br>, which CoolSpools Email will convert to a hard line break.

**\*HTML**

The message is sent in HTML format. You can include HTML formatting (e.g. <b> </b> or <u> </u> to control bold text and underlining). CoolSpools Email will take the text that you enter and wrap it with some basic HTML header and footer controls (<HTML> <HEAD> <BODY>). These controls should not therefore be included in the text of the message.

***Text or path name specified?***

Indicates how the first element of this parameter should be interpreted.

Options are:

**\*MSG**

The first element contains the text of a message to be sent.

**\*STMF**

The first element contains the path to a file containing the text of the message to be sent.

**Example:**

```
CVTSPLPDF...
EMAIL(*YES)
EMAILMSG('Here"s a message <br>with<br>line <br>breaks.'
*TEXT)
```

When this message is received, it will show as:

```
Here's a message
with
line
breaks.
```

**Example:**

```
CVTSPLXLS...
EMAIL(*YES)
EMAILMSG( 'Here"s a message with HTML controls.<br>
<b>This line is in bold, </b><br>
<u>While this line is underlined.</u>')
```

When this message is received, it will show as:

```
Here's a message with HTML controls.
This line is in bold,
While this line is underlined.
```

## EMAILOPT – Email options

Parameter	EMAILOPT
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	EMAIL(*YES)

This parameter allows you to specify various options relating to the sending of the output from the command as an email attachment.

These options apply only to the sending of the output using CoolSpools Email and have no effect on the sending of the output using subsequent calls to SNDDST and other email facilities.

### Delete after sending

This option allows you to indicate whether the output from the command should be deleted as soon as it has been sent as an attachment.

Use this option with caution: if the email fails to arrive for whatever reason, you may lose your data. Note in particular that CoolSpools Email considers the email to have been created if it is able to create the email message and pass it to Mail Server Framework (MSF) for further processing. If you specify \*YES for this option, CoolSpools will delete the file if CoolSpools Email notifies it that the message was created. There is no guarantee that the file will be delivered. For example, bear in mind that if you specified a valid (i.e. well formed) but incorrect (i.e. non-existent) email address, CoolSpools Email will consider that the email was created successfully and CoolSpools will delete the file, but the email will not reach its intended recipient

Possible values are:

**\*NO**

The output is not deleted.

**\*YES**

Once the email has been created, and the stream file attached to it, the stream file is deleted. See the warning above regarding the use of this option.

### Subject

This element allows you to define a subject line for the message. You can enter up to 256 characters of free-format text. When the email message is received, the text that you enter on this parameter element will appear in the subject line of the email.

CoolSpools variables may be specified on this parameter element.

### Attach or embed?

Only available in relation to those commands which produce files whose contents can be embedded, such as CVTSPLHTML and CVTSPLTXT. Binary data cannot be embedded.

This is where you specify the method by which the file is sent.

Note that there is a bug in Microsoft Outlook 2003 which means that attachments sent using the \*EMBED option are treated in the same way as attachments sent using \*ATTACH. See <http://support.microsoft.com/kb/814111>

Options are:

<b><u>*ATTACH</u></b>	The file is sent as an attachment. It will appear as an attached file separate from the text of the email.
<b>*EMBED</b>	The contents of the file are embedded in the text of the email and will follow the text of any message entered on the EMAILMSG parameter. Please note that your client email software is likely only to support the embedding of certain types of file, e.g. text and HTML.
<b>*MSGTXT</b>	CVTSPLHTML only and then only if the first element of the EMAILMSG parameter is *TOSTMF or the same as the path of the file being created. The contents of the file that is created are used as the message text.

### Priority

The priority option controls whether the email message is flagged as a high-priority or low-priority in your email client software.

Values are:

<b><u>*NORMAL</u></b>	The message is sent specifying normal priority. When the message arrives, the client email software will not mark it as high or low priority.
<b>*HIGH</b>	High priority. When the message arrives, the client email software will mark it as high priority.
<b>*LOW</b>	Low priority. When the message arrives, the client email software will mark it as low priority.

### Confirm Delivery

This option controls whether confirmation of delivery is requested from the receiver of the email.

Note that there is no way to oblige the receiver of the email to send confirmation of delivery: the recipient has the right to refuse to send confirmation of receipt.

Specify the email address to receive confirmation of delivery on the EMAILTO parameter by specifying a type of \*CFM.

Values are:

<b><u>*NO</u></b>	No confirmation of delivery is requested.
<b>*YES</b>	The message includes a request that the recipient return confirmation of delivery. When the message is opened, if you have not switched off this feature, the client software will either send a confirmation message back to the sender of the email or ask you whether you wish to send such a confirmation.

## Send Multiple Messages

This option controls whether a single message is sent to all recipients or separate messages to each recipient in turn.

Options are:

**\*NO**

If you are sending an email to more than one recipient, a single message will be created with multiple recipients. When the message is received, each reader will be able to see the names of all people to whom the message was sent.

**\*YES**

If you are sending an email to more than one recipient, multiple messages will be sent, one to each recipient. When the message is received, the reader will see only their own name and will not be able to see the names of the other people to whom the message was sent.

## Attachment name

The name of the attachment, as shown in the email when received.

Options are:

**\*TOSTMF**

The attachment name will be the same as name of the file that is created.

**attachment\_name** Specify an alternative name for the attachment.

## Zip attachment

Whether the attachment should be sent inside a zip file.

Options are:

**\*NO**

The attachment is not sent inside a zip.

**\*YES**

The attachment is sent inside a zip.

## Zip file password

The password for the zip file.

Options are:

**\*NONE**

The zip file is not password-protected or encrypted.

**password**

Specify the case-sensitive password to use when zipping the data.

## Encrypted password supplied

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

- |                   |   |
|-------------------|---|
| <b><u>*NO</u></b> | The password supplied on the previous element is in plain text format and not scrambled.  |
| <b>*YES</b>       | The password supplied on the previous element is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used. |

### Save message to allow resend

Whether or not CoolSpools will keep a copy of the email on the system after it has been sent in order to allow you to resend it using CoolSpools Email's RSNCMNMSG command.

Options are:

- |                   |                         |
|-------------------|-------------------------|
| <b><u>*NO</u></b> | The email is not saved. |
| <b>*YES</b>       | The email is saved.     |

### Retain for how many days

The retention period, in days, to assign to the saved email.

Note that saved emails are only deleted when you run the DLTCMNMSG (Delete Email Messages) command, at which time using the DLTSVMSG(\*MSG) option will delete saved emails that have gone past the end of the retention period.

Options are:

- |                      |                                  |
|----------------------|----------------------------------|
| <b><u>*NOMAX</u></b> | No retention period is assigned. |
| <b>nbr_of_days</b>   | Specify the number of days.      |

### Encryption method

If the zip file is to be encrypted, and a password has been supplied, this element determines the encryption method.

Options are:

- |                       |  |
|-----------------------|--|
| <b><u>*ENVVAR</u></b> | The value of environment variable CS_DFT_ZIP_ENCRYPTION sets the encryption method. If this environment variable exists, and is set to one of the other values permitted for this element (*ZIP, *AES128 or *AES256), that value is used, otherwise *ZIP is used.<br><br>This provides a simple means of setting the default value for this parameter element. |
| <b>*ZIP</b>           | The original zip encryption method. This method is now considered weak and AES is recommended if strong encryption is required. However, this encryption method is likely to be more widely supported than AES, which is recognized by WinZip and most major zip utilities, but not all zip software.  |
| <b>*AES128</b>        | 128-bit AES encryption.  |

**\*AES256**

256-bit AES encryption.

## EMAILSQL – Email address SQL

Parameter	EMAILSQL
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	EMAIL(*YES) EMAILTO(*EMAILSQL)

This parameter allows you to specify that email addresses to which the email will be sent should be read at run time by executing an SQL statement.

A maximum of 100 email addresses can be returned by the SQL statement. If more than 100 email addresses are selected, only the first 100 are used.

The key values can be [CoolSpools variables](#). For example, you might extract a piece of data from the spooled file such as a customer number at run time and use that as a variable in the SQL statement.

### SQL statement

Specify an SQL statement that will be executed at run time to select the email address information required to email the stream files CoolSpools Spool Converter creates.

Please note the following points:

- \*SYS naming must be used (i.e. library\_name/file\_name not collection.table)
- The maximum length of the SQL statement is 5,000 characters
- The SQL statement must return between 1 and 3 values. The first value returned will be interpreted as the email address. The second value returned, if any, will be interpreted as the recipient's name. The third value returned, if any, will be interpreted as the recipient type and must be one of: \*PRI (primary to: recipient), \*CC (cc: recipient) or \*BCC (bcc: recipient). Each of the fields returned must be either fixed or variable length character or graphic data.
- Although it is possible to code [CoolSpools variables](#) in the SQL statement itself and carry out tests against items of data extracted from the spooled file in that way, that technique is not recommended, as it requires the SQL statement to be prepared every time it is run. It is more efficient to code parameter markers in the SQL in the form of ? placeholders and then specify the values to be substituted for those placeholders at run time on **Variable values or names** element below, as this allows the SQL statement to be prepared just once.

### Variable values or names

Specify the constant values or CoolSpools variables names which will be used to replace the parameter markers in the SQL statement at run time.

Up to 10 variable values or names may be specified. The number specified should equal the number of ? parameter markers in the SQL statement, otherwise an error will occur when the SQL statement is prepared.

**Example:**

```
CVTSPLPDF  
FROMFILE(STATEMENTS)  
EMAIL(*YES)  
EMAILTO(*EMAILSQL)  
EMAILSQL('select EMAIL, CUSNAM from CSTMST where CUSTNO = ?'  
'<:CUST_NBR:>')  
SPLIT(*POS)  
SPLITPOS((5 9 7))  
EXITPGM(*VAR)  
EXITPGMPRM(*POS)  
EXITPGMPOS((1 5 9 7 CUST_NBR))
```

Here, a spooled file called STATEMENTS is being converted to PDF and emailed. The spooled file contains a batch of customer account statements, and each separate statement needs to be emailed to the appropriate customer, which is different in each case.

The spooled file contains the customer number on line 5, position 9 for 7 characters, and a new PDF is started every time this value changes. The same value is also extracted from page 1 of the customer statement at run time by means of the EXITPGMPOS parameter, and given the user-defined name CUST\_NBR.

The EMAILTO(\*EMAILSQL) option and associated EMAILSQL parameter indicate that the email address information to be used is to be retrieved at run time by executing the SQL statement specified on the EMAILSQL parameter, namely:

**select EMAIL, CUSNAM from CSTMST where CUSTNO = ?**

This SQL statement contains a single ? parameter marker corresponding to the value of the customer number.

The second part of the EMAILSQL parameter indicates that the ? parameter marker should be replaced at run time by the value of the CUST\_NBR CoolSpools variable.

Note that it would be possible to specify the SQL statement in the form:

**select EMAIL, CUSNAM from CSTMST where CUSTNO = <:CUST\_NBR:>**

rather than using a ? parameter marker, but this is not so efficient, as it requires the SQL statement to be prepared each time it is run rather than just once.



## EMAILTO - Email recipient(s)

Parameter	EMAILTO
Applies to commands:	CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	EMAIL(*YES)

This parameter allows you to specify the email addresses to which the email message should be sent.

You can define up to 32 recipients for the message on this command parameter. If you need to send the same email to more than 32 recipients simultaneously, you can do this by defining a CoolSpools Email address list and specifying the address list name on this parameter.

The default is the single value **\*SELECT**. CoolSpools Email will prompt you to enter email addresses or select email addresses from email address directories. This value is not permitted in batch mode.

There are 3 elements to this parameter:

- **Email address**
- **Name**
- **Type**

Single values are:

### **\*SELECT**

Allowed only if the job is interactive. CoolSpools will prompt you to enter at least one email address or to select the email addresses you want to use from an Email Address Directory.

### **\*EXITPGM**

The email address(es) to use will be supplied at run time by an exit program. See the CoolSpools Programmer's Guide for details of how to write an exit program to do this. Sample source code is provided in source file CS\_SRCFILE.

### **\*EMAILFILE**

Indicates that the related EMAILFILE parameter will be used to tell CoolSpools to look up the email address(es) to be used in a specified file. [CoolSpools variables](#) can be used to extract data from the spooled file at run time to be used as keys to read the file. For example, you might take the customer number from the spooled file and use it to read a customer file to obtain the email address(es) for a particular invoice.

See the EMAILFILE parameter below for further information.

### **\*EMAILSQL**

Indicates that the related EMAILSQL parameter will be used to tell CoolSpools to look up the email address(es) to be used by running an SQL

statement. [CoolSpools variables](#) can be used to extract data from the spooled file at run time and replace parameter markers in the SQL statement. For example, you might take the invoice number from the spooled file and run a piece of SQL to join the invoice file to the customer file to obtain the email address(es) for a particular invoice.

See the EMAILSQL parameter below for further information.

**\*USRDFNDA**

Indicates that USRDFNDA attribute of the spooled file contains one or more email addresses to be used. Optionally, the USRDFNDA attribute can also hold the names of the recipients and an indication of whether those recipients are primary recipients or cc: or bcc: recipients.

If the USRDFNDA attribute is used to supply the email addresses, the attribute contents must be formatted as follows:

1. The USRDFNDA attribute can contain either a single set of email address information or multiple sets of email address information. If there is more than one set of email address information, each set must be separated from the next by a semicolon.
2. Each set of email address information can optionally be prefixed by an indicator of the type of recipient, either to:, cc: or bcc:. The default if none is supplied is to: (primary recipient), equivalent to \*PRI being specified on the "Type" element below.
3. Each set of email address information can optionally include a recipient name, enclosed in double quote characters ", immediately after the type indicator, or at the start of the set of email information if there is no type indicator.
4. Any other information will be interpreted as an email address.
5. White space is ignored.

All of the following are therefore valid:

[noone@nowhere.com](#)

"Someone" [someone@somewhere.com](#) ; cc: [somebodyelse@somewhereelse.com](#)

to: "Mr Nobody" [nobody@nowhere.com](#)

"Mr Nobody" [nobody@nowhere.com](#)

Alternatively, enter the details of the sender on the parameter as follows.

## Email address

This is where you enter the email address to which the message is to be sent.

Note that while CoolSpools Email will check that the email address that you enter conforms to the rules for valid email addresses, it is not possible to validate that the email address that you enter is correct or that the message will be deliverable.

For example, [sales.ariadnesoftware.co.uk](mailto:sales.ariadnesoftware.co.uk) is not a valid email address (since it does not contain an @ sign), and CoolSpools Email will reject it. However, [sales@ariadnesoftware.org.uk](mailto:sales@ariadnesoftware.org.uk) is a valid email address and CoolSpools Email will allow it, but it is not ariadne's correct email address (it should be [sales@ariadnesoftware.co.uk](mailto:sales@ariadnesoftware.co.uk)) and the message will not be received.

You can also specify the name of an email address list on this element, in which case the message will be sent to all email addresses in the list. Specify \*ADL for the following two elements in this case.

CoolSpools variables such as <:CURUSEREMAIL:> and <:SPLUSEREMAIL:> can be used here.

## Name

If you would like your email message to display the recipient's name rather than the email address when it is delivered, enter the name here.

CoolSpools variables such as <:CURUSERNAME:> and <:SPLUSERNAME:> can be used here.

Options are:

### **\*NONE**

No name is provided and the email address will appear as the recipient instead. For example, if you specify:

**EMAILTO((Sales@ariadnesoftware.co.uk  
\*NONE))**

when the message is received, the **To:** attribute will be shown as:

**To:   Sales@ariadnesoftware.co.uk**

However, if you specify:

**EMAILTO((Sales@ariadnesoftware.co.uk  
'ariadne sales'))**

when the message is received, the **To:** attribute will be shown as:

**To:   ariadne sales**

### **\*ADRL**

Specify this value if you supplied the name of an email address list on the previous element.

### **email\_name**

Specify the name of the recipient.

## Type

Specify the type of recipient here.

Options are:

<b>*PRI</b>	Primary recipient.
<b>*CC</b>	Carbon copy recipient. A *CC recipient receives a copy of the message, and is identified to the primary recipient, but is not the primary recipient.
<b>*BCC</b>	Blind carbon copy recipient. A *BCC recipient receives a copy of the message, but is not identified to the primary recipient or *CC recipients.
<b>*ADRL</b>	CoolSpools Email address list. If you wish to send to an address list, this is the value that must be entered. Refer to the CoolSpools Email manual for details of how to create, manage and use email address lists.
<b>*CFM</b>	Confirmation-to email address. If confirmation of delivery is requested (EMAILOPT parameter), the confirmation of delivery email will be routed to this address. This option replaces the use of the EMAILCFM parameter in previous releases.  If no email address of this type is specified, the confirmation of delivery email (if any) is routed to the sender.
<b>*RPY</b>	Reply-to email address. The reply to the email will be routed to the email address specified. This option replaces the use of the EMAILRPY parameter in previous releases.  If no email address of this type is specified, the reply is routed to the sender.

#### Example:

Sending to ariadne sales as a primary recipient with a copy to ariadne marketing:

```
CVTSPLPDF...
EMAIL(*YES)
EMAILTO( (sales@ariadnesoftware.co.uk 'Sales' *PRI)
         (marketing@ariadnesoftware.co.uk 'Marketing' *CC))
```

#### Example:

Sending to an email address list called "Sales":

```
CVTSPLPDF...
EMAIL(*YES)
EMAILTO((Sales *ADRL *ADRL))
```

## EXCEL – Excel Options

Parameter	<b>EXCEL</b>
Applies to commands:	<b>CVTSPLXL, CVTSPLXLS</b>
Dependent on:	
Comments	<b>See also XLSPRPRTY parameter (CVTSPLXL)</b>

This parameter allows you to specify a number of options related to Excel output.

On the CVTSPLXLS command, this parameter has 24 elements:

- **Excel file format version**
- **Keep page headings?**
- **Keep column headings?**
- **Spooled file currency symbol**
- **Spooled file decimal point**
- **Spooled file 1000s separator**
- **Spooled file date format**
- **Spooled file date separator**
- **Spooled file word for 'Page'**
- **Excel date format**
- **Excel worksheet name**
- **Title**
- **Subject**
- **Author**
- **Manager**
- **Company**
- **Category**
- **Keywords**
- **Comments**
- **Page breaks**
- **Remove dot leaders**
- **Suppress underlining**
- **Column separator characters**
- **Number of column separators**

On the CVTSPLXL command, many of these are unnecessary and some have been moved to the XLSPRPRTY parameter. The elements for CVTSPLXL are:

- **Excel file format version**
- **Excel worksheet name**
- **Excel date format**
- **Max rows per worksheet.**
- **Hide unused columns**
- **Hide unused rows**
- **Number of rows to freeze**

Each element allows you to define pages to be excluded from the spooled file according to different criteria.

## Excel file format version

The type of file generated.

Options are:

### **\*XLS**

Excel 97-2003 workbook (.xls file).

This is a binary file format compatible with versions of Excel from Excel 97 onwards.

Please note that certain options, in particular the use of conditional formatting to change number formats or the font name, are not supported by versions of Excel prior to Excel 2007. In order to use those features, you will need to select either the \*XLS07 or \*XLSX file format.

### **\*XLSX**

Excel 2007 Open Office XML format (.xlsx file).

The new XML-based file format introduced with Excel 2007 and compatible with Excel 2007 and 2010.

### **\*XLS07**

Excel 2007 workbook format (.xls file).

This is an adaptation of the Excel 97-2003 format with extensions to support certain new features such as the ability to modify number formatting or the font name using conditional formatting. If you wish to use features introduced with Excel 2007 and do not wish to use \*XLSX format, you must use this format, but please note that the new features will not be available if the file is opened in Excel 97, Excel 2000 or Excel 2003.

Please also note that (confusingly) this is not the same as the Excel 2007 Excel binary workbook format (file extension .xlsb), which is not supported.

### **\*BIFF8**

The same as \*XLS, provided for backwards-compatibility.

The option to output in BIFF5 (Excel 95) format has now been retired.

## Keep page headings?

CVTSPLXLS only. In relation to CVTSPLXL, the treatment of page headings is determined by the way in which they are handled in the Report-to-Excel map.

How CoolSpools handles page headings in the file. This element is ignored unless the new method of allocating text to columns is selected.

Following statistical analysis of a sample of the data in the spooled file, CoolSpools will decide which lines are report data content and which not. Any lines which precede the first report data line, but which do not appear to be a column heading, will be considered a page heading. This element then determines how such lines are handled.

Options are:

**\*FIRST**

The first occurrence of a unique page heading line is retained, all subsequent occurrences of that line are dropped from the output.

Note that any variation in the page heading from one page to the next (such as a change in the time that is printed at the top of the page) may cause CoolSpools to retain a heading you would like to have dropped. You may need to consider using the EXCLLINNBR or EXCLLINKEY parameters to exclude unwanted headings which CoolSpools does not successfully drop.

**\*ALL**

All page headings are retained.

**\*NONE**

All page headings are dropped.

**Keep column headings?**

CVTSPLXLS only. In relation to CVTSPLXL, the treatment of column headings is determined by the way in which they are handled in the Report-to-Excel map.

How CoolSpools handles column headings in the file. This element is ignored unless the new method of allocating text to columns is selected.

Following statistical analysis of a sample of the data in the spooled file, CoolSpools will decide which lines are report data content and which not. Any lines which precede the first report data line, and which overlap the data columns in the report, will be considered column headings.

This element then determines how such lines are handled.

Options are:

**\*FIRST**

The first occurrence of a unique column heading line is retained, all subsequent occurrences of that line are dropped from the output.

Note that any variation in the column heading from one page to the next may cause CoolSpools to retain a heading you would like to have dropped. You may need to consider using the EXCLLINNBR or EXCLLINKEY parameters to exclude unwanted headings which CoolSpools does not successfully drop.

**\*ALL**

All column headings are retained.

**\*NONE**

All column headings are dropped.

**Spooled file currency symbol**

CVTSPLXLS only. In relation to CVTSPLXL, this is determined by the currency symbol option that was specified when the report definition on which the report-to-Excel map is dependent was created.

This element defines the currency symbol that appears when printing currency values in the report.

It is important that CoolSpools knows what currency symbol is used in the report so that it can correctly identify columns of numbers that include a currency symbol as numeric data rather than treating them as text.

Options are:

<b><u>*SYSVAL</u></b>	The currency symbol defined by the QCURSYM system value is used in the report.
<b>currency_symbol</b>	Specify the currency symbol used in the report if this is different from the system currency symbol. For example, if you are processing a report containing values in Euros on a system where the currency symbol is a pound sign (£), specify €. CoolSpools will interpret data containing euro signs as numeric data not text.

### Spoiled file decimal point

CVTSPLXLS only. In relation to CVTSPLXL, this is determined by the decimal point character option that was specified when the report definition on which the report-to-Excel map is dependent was created.

This element defines the decimal point that is used when printing numbers in the report.

It is important that CoolSpools knows what decimal point symbol is used in the report so that it can correctly identify columns of numbers as numeric data rather than treating them as text.

Options are:

<b><u>*JOB</u></b>	The decimal point defined by the DECFMT attribute of the current job is used in the report.
<b>*SYSVAL</b>	The decimal point defined by the QDECFMT system value is used in the report.
<b>Decimal_point</b>	Specify the decimal point used in the report. For example, if you are processing a report containing numbers that have a comma as the decimal point on a system where the normal decimal point is a period (.), specify , (comma). CoolSpools will interpret commas in numeric data as a decimal point, not a thousands separator.

### Spoiled file 1000s separator

CVTSPLXLS only. In relation to CVTSPLXL, this is determined by the thousands separator option that was specified when the report definition on which the report-to-Excel map is dependent was created.

This element defines the thousands separator character that is used when printing numbers in the report.

It is important that CoolSpools knows what thousands separator character is used in the report so that it can correctly identify columns of numbers as numeric data rather than treating them as text.

Options are:



<b><u>*JOB</u></b>	The thousands separator character defined by the DECFMT attribute of the current job is used in the report.
<b>*SYSVAL</b>	The thousands separator character defined by the QDECFMT system value is used in the report.
<b>1000s_sep</b>	Specify the thousands separator character used in the report. For example, if you are processing a report containing numbers that have a period as the thousands separator character on a system where the normal thousands separator character is a comma (,), specify . (period). CoolSpools will interpret periods in numeric data as a thousands separator character, not a decimal point.

### Spoiled file date format

CVTSPLXLS only. In relation to CVTSPLXL, this is determined by the date format option that was specified when the report definition on which the report-to-Excel map is dependent was created. This element defines the date format that is used when printing dates in the report.

It is important that CoolSpools knows what date format is used in the report so that it can correctly identify dates and treat them as such.

Options are:

<b><u>*JOB</u></b>	The date format defined by the DATFMT attribute of the current job is used in the report.
<b>*SYSVAL</b>	The date format defined by the QDATFMT system value is used in the report.
<b>*DMY</b>	The date format used in the report is day-month-year. CoolSpools will identify data in the report which looks like a valid DMY date as a date (2-digit or 4-digit year).
<b>*MDY</b>	The date format used in the report is month-day-year. CoolSpools will identify data in the report which looks like a valid MDY date as a date (2-digit or 4-digit year).
<b>*YMD</b>	The date format used in the report is year-month-day. CoolSpools will identify data in the report which looks like a valid YMD date as a date (2-digit or 4-digit year).

### Spoiled file date separator

CVTSPLXLS only. In relation to CVTSPLXL, this is determined by the date separator option that was specified when the report definition on which the report-to-Excel map is dependent was created. This element defines the date format that is used when printing dates in the report.

This element defines the date separator that is used when printing dates in the report.

It is important that CoolSpools knows what date separator is used in the report so that it can correctly identify dates and treat them as such.

Options are:

<b><u>*JOB</u></b>	The date separator defined by the DATFMT attribute of the current job is used in the report.
<b>*SYSVAL</b>	The date separator defined by the QDATFMT system value is used in the report.
<b>date_sep</b>	Specify the date separator character used in the report. For example, if you are processing a report containing dates that have a hyphen as the date separator on a system where the normal date separator character is a slash, specify - (hyphen).

### Spooled file word for 'Page'

CVTSPLXLS only. In relation to CVTSPLXL, this information is irrelevant as page headings are included or excluded dependent on what has been specified in the Report-to-Excel map.

This element defines the word "Page" as it appears in the report.

When excluding page headings, CoolSpools attempts to take account of headings which differ only by a change of page number. In order to do so, it looks for the word defined on this element followed by a number and treats that text as a page number and ignores it for the purposes of deciding whether a page heading is a new one or a repetition of a previous one.

Options are:

<b><u>*DFT</u></b>	The word for "Page" is taken from the text of message CVT0008 in message file CP_MSGF. This is shipped in the English version of CoolSpools as "Page".  Please note that if you change the text in this message file, you will need to change it back again after applying PTFs or new versions.
<b>Word_for_page</b>	Specify the word for "Page" as it is used with page numbers in the report. For example, if it is abbreviated to "P.", specify "P." here. Similarly, if you are processing a Spanish-language report, you may need to specify Página.

### Page breaks

CVTSPLXLS only.

Whether CoolSpools should insert page breaks in the Excel file at the end of each page in the original spooled file.

Options are:

<b><u>*NO</u></b>	No page breaks will be inserted.
-------------------	----------------------------------

**\*YES**

A page break will be inserted in the Excel file after the last row of each page in the original spooled file.

### **Remove dot leaders**

Whether CoolSpools should remove dot leaders from the output.

CVTSPLXLS only.

Dot leaders (such as in 'Customer number . . . . . : XXXXXX') are often used in reported to connect data items with their associated labels, but in Excel output they can make the document look messy, and can confuse the logic which determines where to place column breaks.

Options are:

**\*NO**

Dot leaders are not removed.

**\*YES**

Dot leaders are removed (replaced with spaces).

### **Suppress underlining**

CVTSPLXLS only.

Whether CoolSpools should remove consecutive underscores from the output.

Where underlining in the output is created using underscore characters (\_), while this can enhance the appearance of the printed page, it tends to detract from the appearance of the data in an Excel file. CoolSpools will remove any consecutive underscores it finds.

Options are:

**\*YES**

Underscores are removed.

**\*NO**

Underscores are not removed.

### **Column separator characters**

CVTSPLXLS only.

When COLUMNOPT(\*TOKEN) is specified, the values you select on this parameter determine the way in which CoolSpools splits data in the spooled file up into columns in the Excel or delimited file being created.

The default is \*SPLF. This tells CoolSpools to break the spooled file data into columns based on the way in which the data is organized in the spooled file.

Where the spooled file is created from an externally described printer file, this method will probably give the best results, since the data in the spooled file is likely to be organized so that each natural or logical item of data appears as a separately identifiable element.

However, if the spooled file is created from an internally described printer file, or from an application such as Query/400, it is likely that the data in the spooled file will be presented to CoolSpools as a single, unstructured data block for each line of the report. If this is the case then better results will probably be achieved by defining a column separator character to control the splitting of the data in the report.

Every time CoolSpools encounters n consecutive characters of the type defined on this parameter element, it will start a new column, n being the number defined for the next element ("Number of column separators").

For example, if you specify \*BLANK for this element and 2 for the next, CoolSpools will create a new column every time 2 or more consecutive blanks are found in the spooled file.

If CoolSpools recognizes that the spooled file has been created from a file without DDS or output by a Query/400 query, then it will automatically switch to using the equivalent of \*BLANK for "Column separator" and 2 for the "Number of column separators" (see next).

Options are:

<b><u>*SPLF</u></b>	Split the data in the report based on the internal organization of the data in the file.
<b>*BLANK</b>	The separator character is the blank (space) character.
<b>character-value</b>	The separator character to use to identify column breaks.

### Number of column separators

CVTSPLXLS only.

When COLUMNOPT(\*TOKEN) is specified, the number of consecutive column separator character that must appear before a column break occurs.

<b><u>*NONE</u></b>	This method is not used.
<b>1-9</b>	The number of consecutive characters required.

### Excel date format

The formatting applied to dates in the Excel spreadsheet.

Using the information specified above concerning the format and separators used for dates in the report, CoolSpools will attempt to identify data items in the report which are dates. These will be output as standard Excel dates (a day count since the era) in numeric cells but appropriate date formatting will be applied as specified here.

<b><u>*MM</u></b>	A two-digit numeric month will be used, e.g. 09/08/2010.  The date format will otherwise be determined by your Excel settings and the regional settings of the PC.
<b><u>*MMM</u></b>	A three-character month will be used, e.g. 09-Aug-2010 or Aug-09-2010.  The date format will otherwise be determined by your Excel settings and the regional settings of the PC.

### Excel worksheet name

The name that CoolSpools will give to the worksheet it creates in the Excel file.

If this name ends in a number (e.g. "Sheet1"), CoolSpools will generate names for subsequent worksheets by incrementing this number (e.g. "Sheet2", "Sheet3"). If the name specified does not end in a number, CoolSpools will generate the name of

subsequent worksheets by appending a numeric suffix (e.g. if the sheet name specified is "Invoices", the next sheet will be "Invoices2" etc.).

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*DFT</u></b>	The worksheet name is taken from the text of message CVT0021 in message file CP_MSGF. This is shipped in the English version of CoolSpools as "Sheet1".  Please note that if you change the text in this message file, you will need to change it back again after applying PTFs or new versions.
<b>Sheet_name</b>	Specify the name of the worksheet CoolSpools should create.

### Title

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The title that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no title.
<b>Title</b>	The title that should appear.

### Subject

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The subject that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no subject.
<b>Subject</b>	The subject that should appear.

### Author

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The author's name that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no author's name.
<b>*USRPRF</b>	The user profile of the user running the CoolSpools command will appear as the author's name.

<b>*JOB</b>	The name of the job running the CoolSpools command will appear as the author's name.
<b>*QUALJOB</b>	The fully qualified name of the job running the CoolSpools command will appear as the author's name (i.e. job_number/user_profile/job_name).
<b>Author</b>	The author's name that should appear.

### Manager

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The manager's name that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no manager's name.
<b>Manager</b>	The manager's name that should appear.

### Company

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The company name that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no company name.
<b>Company</b>	The company name that should appear.

### Keywords

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The keywords that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no keywords
<b>Keywords</b>	The keywords that should appear.

### Comments

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The comments that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	The file will have no comments
<b>Comment</b>	The comments that should appear.

## Category

CVTSPLXLS only. In relation to CVTSPLXL, this information is specified on the XLSPRPRTY parameter.

The category that should appear in the Excel file properties.

CoolSpools variables may be specified on this parameter element.

Options are:

**\*NONE**

The file will have no category

**Category**

The category that should appear.

## Max rows per worksheet.

CVTSPLXL only.

The maximum number of records to write to a worksheet before starting a new worksheet.

Options are:

**\*XLSVER**

The limit is determined by the version of Excel:

\*BIFF8 (\*XLS): 65,535

\*XLSX: 1,048,576

**Max\_rows**

Specify the maximum number of rows. The value must be less than the limit implied by the Excel version selected.

## Hide unused columns

CVTSPLXL only.

Whether unused columns are hidden or not

Options are:

**\*NO**

Unused columns are not hidden and empty columns appear to the right of the last used column.

**\*YES**

Unused columns are hidden and no empty columns appear to the right of the last used column.

## Hide unused rows

Whether unused rows are hidden or not

Options are:

**\*NO**

Unused rows are not hidden and empty rows appear below the last used row in the last worksheet.

**\*YES**

Unused rows are hidden and empty rows no empty rows appear below the last used row in the last worksheet.

### Number of rows to freeze

How many rows should be included in a frozen pane at the top of each worksheet. A frozen pane remains visible while the rest of the rows scroll.

Options are:

**\*NONE** No rows are frozen.

**number\_of\_rows** Specify the number of rows in the frozen pane.



## **EXCLLINKEY - Exclude lines by key**

Parameter	<b>EXCLLINKEY</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, CVTSPLSTMF</b>
Dependent on:	<b>None</b>

The **EXCLLINKEY** (Exclude Lines by Key) parameter specifies sets of lines on the report which should be excluded from the output based on the appearance in the line of a key string.

There are two elements to this parameter.

- **Exclude lines containing text**
- **Number of lines**

The CVTSPLCSV, CVTSPLTXT and CVTSPLXLS have two additional elements:

- **From page**
- **To page**

Up to 20 sets of lines can be specified on this parameter.

### **Exclude lines containing text**

Specify a key string. Every line which contains the key string will start a set of lines to be excluded. The number of lines specified on the following parameter element will be dropped from the output from that point onwards.

### **Number of lines**

Specify the number of lines to be dropped from the output starting at the each line containing the string defined on the previous parameter element.

### **From page**

The lines will only be dropped starting from the page specified.

Please note that this page number refers to the relative page number within the group of pages selected by splitting, not the absolute page number in the original spooled file. For example, if a 30-page spooled file is split into 3 10-page sections, specifying a page number of 2 on this element would refer to pages 2, 12 and 22 in the original spooled file.

The default is \*FIRST, denoting the first page in the section of the relevant spooled file.

### **To page**

The lines will only be dropped up to the page specified.

Please note that this page number refers to the relative page number within the group of pages selected by splitting, not the absolute page number in the original spooled file. For example, if a 30-page spooled file is split into 3 10-page sections, specifying a page number of 2 on this element would refer to pages 2, 12 and 22 in the original spooled file.

The default is \*LAST, denoting the last page in the section of the relevant spooled file.

## **EXCLLINBR – Exclude Line Numbers**

Parameter	<b>EXCLLINBR</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, CVTSPLSTMF</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and TOFMT(*CSV), TOFMT(*TEXT) or TOFMT(*XLS)</b>

The **EXCLLINBR** (Exclude Line Numbers) parameter specifies lines in the spooled file being converted which should be excluded from the output.

This option can be useful for dropping items such as page and column headings from output where it is not required (e.g. Excel spreadsheets, CSV or text files).

There are two elements to this parameter on the CVTSPLSTMF command.

- **From line number**
- **Number of lines**

The CVTSPLCSV, CVTSPLTXT and CVTSPLXLS have two additional elements:

- **From page**
- **To page**

Up to 20 sets of lines can be specified on this parameter.

### **From line number**

Specify the line number on the page at which exclusion is to begin. Starting with the line number specified, the number of lines input on the next parameter element will be dropped from the data when the output is created.

### **Number of lines**

The number of lines to be dropped from the output, starting at the line number specified above, on each page.

### **From page**

The lines will only be dropped starting from the page specified.

Please note that this page number refers to the relative page number within the group of pages selected by splitting, not the absolute page number in the original spooled file. For example, if a 30-page spooled file is split into 3 10-page sections, specifying a page number of 2 on this element would refer to pages 2, 12 and 22 in the original spooled file.

The default is \*FIRST, denoting the first page in the section of the relevant spooled file.

### **To page**

The lines will only be dropped up to the page specified.

Please note that this page number refers to the relative page number within the group of pages selected by splitting, not the absolute page number in the original spooled file. For example, if a 30-page spooled file is split into 3 10-page sections, specifying a page number of 2 on this element would refer to pages 2, 12 and 22 in the original spooled file.

The default is \*LAST, denoting the last page in the section of the relevant spooled file.

## **EXCLPAGKEY – Exclude pages by key string**

Parameter	<b>EXCLPAGKEY</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSPLF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>
Dependent on:	<b>None</b>

With EXCLPAGNBR, this parameter replaces the EXCLPAGES parameter of the CVTSPLSTMF command. It provides options for excluding pages from the output based on the appearance or non-appearance of a key string. For example, pages which appear in the spooled file but which are not required in the output such as batch header and trailer sheets can be dropped by means of this parameter.

It consists of the following elements:

- **Key string**
- **Option**
- **Pages to exclude**

Up to 100 options may be specified.

The single value \*NONE indicates that no pages are to be excluded by key string.

### **Key string**

A key string identifying pages to be dropped.

Every page on which the key string appears, or every page on which the key string does not appear (dependent on the following option), will be excluded

### **Option**

The way the key string operates.

Options are:

<b>*CT</b>	“Containing”. Any page which contains the specified key string will be excluded.
<b>*NC</b>	“Not containing”. Any page which does not contain the specified key string will be excluded.

For example, if your spooled file contains batch header sheets, and these contain the word “Batch” on them, they can be excluded from the stream file by specifying “Batch” as the key string on this parameter and \*CT for the exclude option.

### **Pages to exclude**

How many pages to exclude when the key string is found (\*CT) or not found (\*NC).

#### **Example:**

```
CVTSPLPDF FROMFILE(INVOICES)...  
EXCLPAGKEY(( 'Batch' *CT 1))
```

The spooled file contains batch header sheets and these are not required in the PDF files. They are dropped because they contain the text string ‘Batch’.

## **EXCLPAGNBR – Exclude pages by page number**

Parameter	<b>EXCLPAGNBR</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSPLF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>
Dependent on:	<b>None</b>

With EXCLPAGKEY, this parameter replaces the EXCLPAGES parameter of the CVTSPLSTMF command. It provides options for excluding pages from the output. For example, pages which appear in the spooled file but which are not required in the output such as batch header and trailer sheets can be dropped by means of this parameter.

It consists of the following elements:

- **Exclusion point**
- **Pages to exclude or \*BLANK**

Each element allows you to define pages to be excluded from the spooled file according to different criteria.

Up to 100 options may be specified.

The single value \*NONE indicates that no pages are to be excluded by page number.

### **Exclusion point**

The point in the spooled file where the page(s) to be excluded are located.

Options are:

<b>*SPLFSTR</b>	At the start of the spooled file. The specified number of pages are dropped from the start of the spooled file.
<b>*SPLFEND</b>	At the end of the spooled file. The specified number of pages are dropped from the end of the spooled file.
<b>*STMFSTR</b>	At the start of each stream file. The specified number of pages are dropped from the start of the each set of pages selected to create a new stream file.
<b>*STMFEND</b>	At the end of the stream file. The specified number of pages are dropped from the end of the each set of pages selected to create a new stream file.
<b>*PAGNBR</b>	Indicates that the “Pages to exclude or *BLANK” element denotes a page number, not a number of pages.

The page number specified will be dropped from the output.

**\*ANY**

Anywhere in the file. This option can only be used, in conjunction with the value \*BLANK for the next element, in order to drop blank pages found at any point in the spooled file.

**Pages to exclude or \*BLANK**

The number of pages to exclude or the page number to exclude.

Options are:

**Nbr\_of\_pages**

The number of pages to be excluded at the specified position, or, in the case of \*PAGNBR, the page number to be excluded.

**\*BLANK**

All blank pages are excluded:

- up to the next non-blank page (if \*SPLFEND or \*STMFEND was specified for the previous element.
- back to the preceding non-blank page (if \*SPLFEND or \*STMFEND)
- anywhere in the file (if \*ANY)

**Example:**

**CVTSPLPDF**

**FROMFILE(INVOICES)...  
EXCLPAGNBR((\*SPLFSTR 1))**

The spooled file being processed here contains an unwanted header sheet at the beginning of the file. This is dropped from the PDF file being created because this parameter indicates that the first page of each file should be excluded.

## EXITPGM – Exit Programs

Parameter	<b>EXITPGM</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF</b>
Dependent on:	<b>None</b>

This parameter allows you to specify the up to 100 user-written exit programs which CoolSpools will call at various different pre-defined exit points within the CoolSpools processing cycle.

The default is the single value **\*NONE**, indicating that no exit programs are to be called.

An alternative single value is **\*VAR**. Specifying EXITPGM(\*VAR) indicates that you do not wish to call any exit programs, but you do wish to use the EXITPGMPRM, EXITPGMPOS and/or EXITPGMKEY parameters for the purposes of defining variables to be extracted from the spooled file and referred to through the CoolSpools variable names <:EXITPGMPOSn:> (where n = 1-99) or <:EXITPGMKEYn:> (where n = 1-99).

Where multiple programs are defined at the same exit point, CoolSpools will call them in the order in which they are listed on the parameter.

For each program to be called, you must specify 3 items:

- **Program**
- **Format of program parameters**
- **Exit point**

### Program

Specify the fully qualified name of the program to be called.

The following special values may be specified for the library name:

- \*LIBL** The program is located using the library list of the job.
- \*CURLIB** The program is located in the current library.

### Format of program parameters

This element defines the parameters which will be passed to the exit program. These parameters will include a standard list of parameters (such things as the spooled file name and the name of stream file) as well as any user-defined exit program parameters extracted from the spooled file as a result of the use of the EXITPGMPRM, EXITPGMPOS and EXITPGMKEY parameters.

It is critical that you select the value for this element which corresponds to the parameter list expected by the program to be called, otherwise errors will probably occur due to mismatched parameters between caller and called program.

These parameter formats are discussed in detail in the CoolSpools Programmer's Guide.



There are four possible formats.

<b>*TYPE3</b>	The format introduced with Version 5. Still the default for reasons of backwards compatibility.
<b>*TYPE4</b>	A new format introduced with Version 6 which passes all information as a single program parameter in the form of a data structure. Any future enhancements will be made to this format only.
<b>*TYPE2</b>	Introduced with Version 3. Provided mainly for reasons of backwards compatibility.
<b>*TYPE1</b>	Introduced with Version 2. Provided mainly for reasons of backwards compatibility.

## Exit point

These exit points are discussed in detail in the CoolSpools Programmer's Guide.

Briefly, they are:

<b>*SPLFSTR</b>	Start of spooled file. This is the first exit point to be called and it is called just once. This is a good point at which to call initialization routines.
<b>*PAGECTL</b>	Page control. This is a special exit point provided in order to allow an exit program to indicate, for each page in the stream file about to be created, whether that page should be included in or excluded from the output.
<b>*STMFSTR</b>	Start of stream file. This exit point is called once before starting to create each stream file. If you are splitting a single spooled file into multiple stream files, this exit point will be called once for each stream file that is generated. This is a good point to override items specific to the stream file such as its name and password.
<b>*PAGESTR</b>	Start of page. Called once for each page before the data for the page is converted.
<b>*PAGEEND</b>	End of page. Called once for each page after the data for the page has been converted.
<b>*STMFEND</b>	End of stream file. This exit point is called once after finishing creation of each stream file. If you are splitting a single spooled file into multiple stream files, this exit point will be called once for each stream file that is generated. This is a good point to do things like renaming, moving or otherwise post-processing the file just created.
<b>*SPLFEND</b>	End of spooled file. This is the last exit point to be called and it is called just once. This is a good point at which to call housekeeping routines.

<b>*SHEETSTR</b>	Start of sheet. Used only by CVTSPLXL. Exit programs defined at this exit point are called just before a new worksheet is started, giving you the opportunity to change attributes such as the sheet name.
<b>*SHEETEND</b>	End of sheet. Used only by CVTSPLXL. Exit programs defined at this exit point are called just after a worksheet is finished.

For further details on how to use exit programs and for ideas on the kinds of applications for which exit programs can be used, refer to the CoolSpools Programmer's Guide.

Sample exit program source code for various purposes is available from ariadne software on request.

## EXITPGMPRM - Exit Program Parameters

Parameter	EXITPGMPRM
Applies to commands:	CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXLS, SAVSPLF
Dependent on:	Not EXITPGM(*NONE)

This parameter allows you to define whether CoolSpools should pass any user-definable parameter strings to the exit program(s) defined on the EXITPGM parameter, and, if so, on what basis.

Refer to the CoolSpools Programmer's Guide for further details of how to write and call an exit program.

If EXITPGM(\*VAR) is specified, no exit programs will be called but you can use the EXITPGMPRM, EXITPGMPOS and/or EXITPGMKEY parameters for the purposes of defining variables to be extracted from the spooled file and referred to through the CoolSpools variable names <:EXITPGMPOSn:> (where n = 1-99) or <:EXITPGMKEYn:> (where n = 1-99).

### Type of parameters

Options are:

<b>*NONE</b>	No text will be extracted from the report.  For an exit program with a Type 1 parameter list, the user-definable parameter will consist of all blanks.  For an exit program with a Type 2 or Type 3 parameter list, the user-definable parameter count will be zero and no user-definable parameters will be passed.
<b>*POS</b>	You will use the EXITPGMPOS parameter to define one or more areas of the page from which text will be extracted and passed to the exit program(s) as user-definable parameters.
<b>*KEY</b>	You will use the EXITPGMKEY parameter to define one or more areas of the page from which text will be extracted and passed to the exit program(s) as user-definable parameters.
<b>*POSKEY</b>	You will use both the EXITPGMPOS and EXITPGMKEY parameters to define one or more areas of the page from which text will be extracted and passed to the exit program(s) as user-definable parameters.
<b>*BOTH</b>	Same as *POSKEY. Available from CVTSPLSTMF and provided for reasons of backwards compatibility only.

## CCSID of parameter data

Determines the CCSID (Coded Character Set Identifier) which should be used when passing data to exit programs.

Some spooled files (notably those of type \*USERASCII) will hold data internally in ASCII. If the data is passed in its original form, it may not be easy to process in an exit program. If you would like CoolSpools to convert this data to a more user-friendly CCSID (e.g. an EBCDIC CCSID) before passing it to the exit program, specify the CCSID to be used on this parameter.

Options are:

<b><u>*EBCDIC</u></b>	If the spooled file is of type *USERASCII, the data is converted to the CCSID of the job before being passed; otherwise it is not converted at all. This is intended to ensure that ASCII data is not returned from *USERASCII spooled files.
<b>*SPLF</b>	The data is passed in its original form as extracted from the spooled file. Note that this could be ASCII, for example where the spooled file is of type *USERASCII.
<b>*JOB</b>	The CCSID of the current job is used.
<b>*SYSVAL</b>	The system CCSID (QCCSID system value) is used.
<b>*USER</b>	The CCSID of the current user (from the user profile) is used.
<b>CCSID_value</b>	Specify the CCSID in which the data should be passed.

## EXITPGMKEY - Exit program parameters string key

Parameter	<b>EXITPGMKEY</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF</b>

This parameter allows you to define key strings which CoolSpools will use as triggers for extracting text from the report to pass as parameters to the exit program(s) you specified on the **EXITPGM** parameter.

If EXITPGM(\*VAR) is specified, no exit programs will be called but you can use the EXITPGMKEY parameter for the purposes of defining variables to be extracted from the spooled file and referred to through the CoolSpools variable name <:EXITPGMKEYn:> (where n = 1-99, corresponding to the order in which the different parameters are defined on the EXITPGMKEY parameter).

Up to 100 parameters may be selected in this way.

Refer to the CoolSpools Programmer's Guide for further details of how to write and call an exit program.

### Page number

The page number from which the text should be extracted. CoolSpools will extract the text from the position on the page specified below and will pass it as a parameter to the exit program(s) defined on the EXITPGM parameter, but only for the page specified here.

Note that this is the page number from the output file, not the input file. For example, if CoolSpools splits a 10-page spooled file into two 5-page stream files, then a page number of 1 on this parameter would refer to the first page in the two stream files, i.e. pages 1 and 6 from the original spooled file.

Alternatively, specify **\*ALL** and CoolSpools will pass the value at the specified location on each page in the output file. If the output file consists of 5 pages, 5 parameters will be passed, one for each page.

In the main, use DSPSPLF as your guide to determine the offset length below. However, where you specify an alternative CPI and/or LPI value on the TEXT parameter, column and/or line numbers will differ from DSPSPLF (which always uses the spooled file attribute CPI and LPI settings). You might find it helpful to convert the spooled file to text with CVTSPLTXT using the same CPI and LPI settings and calculate offset and length from the text file thus produced.

### Key string

Specify the key string which will trigger the selection of parameter text.

This value is case-sensitive.

### Occurrence

Where the key string appears more than once on each page, the number you enter on this parameter element will determine which occurrence of the key string will trigger the selection of parameter text.

## Offset

Enter the offset in characters from the start of the key string to the start of the text to be selected as a parameter.

If a positive number is entered, this is interpreted as indicating that the parameter text is to the right of the key string, whereas a negative number indicates that the parameter text is to the left of the key string.

## Length

Enter the length of the parameter text in characters.

## Variable name

A name you can optionally assign to data selected from the spooled file using this parameter.

Options are:

### **\*NONE**

No user-defined name is assigned to this variable. If you want to refer to this item of data through a [CoolSpools variable](#), the only way to do it is to use the form <:EXITPGMKEYn:> where n is the ordinal number of the element of the EXITPGMKEY parameter (first element = 1, second element = 2 etc.)

### **var\_name**

Specify a variable name, without the variable markers <:...>. The name you specify must be a valid OS/400 name up to 20 characters in length.

## Example:

```
CVTSPLPDF      FROMFILE(INVOICES)
                TOSTMF('<:CUSTOMER_NUMBER:>.pdf') ...
                SPLIT(*PAGE)
                SPLITPAGE(1)
                EXITPGM((Invexit))
                EXITPGMPRM(*KEY)
                EXITPGMKEY(( 1
                             'Customer name:'
                             1
                             15
                             10
                             CUSTOMER_NUMBER))
```

In this example, the invoices spooled file is converted to separate PDF files for each page of the report. Every time a file has been completed, a program called INVEXIT will be called. The program will be passed 10 characters of text extracted from the spooled file starting 15 characters to the right of the first occurrence of the string 'Customer name:' on the first page written to each stream file.

This value is assigned the user-defined name CUSTOMER\_NUMBER and this is referenced as a CoolSpools variable on the TOSTMF parameter to construct a name of the stream file consisting of the customer number followed by an extension of ".pdf".

## EXITPGMPOS - Exit program parameters string position

Parameter	EXITPGMPOS
Applies to commands:	CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXLS, SAVSPLF

This parameter allows you to define positions in the report from which CoolSpools will extract items of text and pass them as parameters to the exit program(s) you specified on the **EXITPGM** parameter.

If EXITPGM(\*VAR) is specified, no exit programs will be called but you can use the EXITPGMPOS parameter for the purposes of defining variables to be extracted from the spooled file and referred to through the CoolSpools variable name <:EXITPGMPOSn:> (where n = 1-99, corresponding to the order in which the different parameters are defined on the EXITPGMPOS parameter).

If you specify your own variable name on **Variable name** below, you can also refer to the item of data extracted by this parameter using that name.

Up to 100 parameters may be selected in this way.

Refer to the CoolSpools Programmer's Guide for further details of how to write and call an exit program.

### Page number

The page number from which the text should be extracted.

CoolSpools will extract the text from the position on the page specified below and will pass it as a parameter to the exit program(s) defined on the EXITPGM parameter, but only for the page specified here.

Note that this is the page number from the output file, not the input file. For example, if CoolSpools splits a 10-page spooled file into two 5-page stream files, then a page number of 1 on this parameter would refer to the first page in the two stream files, i.e. pages 1 and 6 from the original spooled file.

Alternatively, specify **\*ALL** and CoolSpools will pass the value at the specified location on each page in the output file. If the output file consists of 5 pages, 5 parameters will be passed, one for each page.

In the main, use DSPSPLF as your guide to determine the line number and column position. However, where you specify an alternative CPI and/or LPI value on the TEXT parameter, column and/or line numbers will differ from DSPSPLF (which always uses the spooled file attribute CPI and LPI settings). You might find it helpful to convert the spooled file to text with CVTSPLTXT using the same CPI and LPI settings and calculate line and column numbers from the text file thus produced.

### Line number

Enter the line number on which the parameter text appears in the spooled file.

In the main, use DSPSPLF as your guide to decide the line number. However, where you specify an alternative CPI and/or LPI value on the TEXT parameter, column and/or line numbers will differ from DSPSPLF (which always uses the spooled file attribute CPI and LPI settings). You might find it helpful to convert the spooled file to

text with CVTSPLTXT using the same CPI and LPI settings and calculate line and column numbers from the text file thus produced.

### Character position

Enter the column number on which the parameter text appears in the spooled file.

### Length

The number of characters which the parameter text occupies in the spooled file.

### Variable name

A name you can optionally assign to data selected from the spooled file using this parameter.

Options are:

#### **\*NONE**

No user-defined name is assigned to this variable. If you want to refer to this item of data through a [CoolSpools variable](#), the only way to do it is to use the form <:EXITPGMPOSn:> where n is the ordinal number of the element of the EXITPGMPOS parameter (first element = 1, second element = 2 etc.)

#### **var\_name**

Specify a variable name, without the variable markers <...>. The name you specify must be a valid OS/400 name up to 20 characters in length.

### Example:

```
CVTSPLPDF      FROMFILE(INVOICES)
                TOSTMF('<:INVOICE_NUMBER:>.pdf') ...
                SPLIT(*PAGE)
                SPLITPAGE(5)
                EXITPGM((Invexit)
                EXITPGMPOS(*ALL 10 20 7 'INVOICE_NUMBER' )
```

In this example, the invoices spooled file is converted to separate PDF every 5 pages. Every time a file has been completed, a program called INVEXIT will be called. The program will be passed 5 user-defined parameters representing the 7 characters of text located on line 10 starting at column 20 on each of the 5 pages of the stream file.

This value is assigned the user-defined name INVOICE\_NUMBER and this is referenced as a CoolSpools variable on the TOSTMF parameter to construct a name of the stream file consisting of the invoice number followed by an extension of “.pdf”.



## FONT – Font options

Parameter	FONT
Applies to commands:	<b>CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSTMF, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>

This parameter lets you manage the way CoolSpools handles fonts when creating a stream file from your spooled file.

### Face

The first element is the Font Face.

For CVTSPLPDF, options are as follows.

The following special options are available:

#### **\*MAP**

In relation to PDF, CoolSpools maps system i fonts to an equivalent PC font from the PDF basic font set (listed below). This option will minimize the size of the PDF file created, but the appearance of the text in the PDF file may not necessarily exactly reproduce the appearance of the text in the system i spooled file when printed, but will normally be very close to it.

In relation to HTML and RTF, CoolSpools will likewise select a standard PC font equivalent to the font used in the system i spooled file. Typically this will be Courier for fixed-pitch fonts, Arial for sans-serif proportional fonts and Times for other proportional fonts.

This is a single value on the CVTSPLHTML, and CVTSPLRTF commands and is not available for CVTSPLXLS.

#### **\*CONVERT**

Same as \*MAP, which has replaced it. This value is available only from CVTSPLSTMF, where it is supported for reasons of backwards compatibility.

#### **\*EMBED**

This option is only available with CVTSPLPDF or CVTSPLSTMF with TOFMT(\*PDF).

Where possible, CoolSpools will embed the font in the PDF file. This guarantees that the font will be available when the PDF file is viewed in Acrobat, and will reproduce the appearance of the system i font on the printed page as closely as possible. However, it may also significantly increase the size of the resultant PDF.

Please note also that embedding low-resolution system i raster fonts in PDF may not give good results when the resultant PDF file is viewed

online. This is simply due to limitations of the raster font technology. When printed, the appearance should be comparable to the quality when the font is used to print documents from your system (though clearly this depends to some extent on the printers used in each case).

In addition, you can select one of the following predefined font names corresponding to the basic PDF font set. When used with PDF, these fonts are guaranteed by Adobe Acrobat to be always available when you view a PDF file that uses them. All text in the stream file will use the font selected.

<b>*COURIER</b>	Courier. All text will appear in Courier font.
<b>*COURIERB</b>	Courier Bold
<b>*COURIERO</b>	Courier Oblique
<b>*COURIERBO</b>	Courier Bold Oblique
<b>*HELVETICA</b>	Helvetica
<b>*HELVB</b>	Helvetica Bold
<b>*HELVO</b>	Helvetica Oblique
<b>*HELVB</b>	Helvetica Bold
<b>*HELVBO</b>	Helvetica Bold Oblique
<b>*TIMES</b>	Times Roman
<b>*TIMESB</b>	Times Roman Bold
<b>*TIMESI</b>	Times Roman Italic
<b>*TIMESBI</b>	Times Roman Bold Italic
<b>*SYMBOL</b>	Symbol
<b>*DINGBATS</b>	Zapf Dingbats

For CVTSPLXL and CVTSPLXLS, specify the name of the font to use in the file. The following special values are predefined but any font name can be specified:

<b>*ARIAL</b>	Arial.
<b>*COURIER</b>	Courier
<b>*TIMES</b>	Times New Roman

## Size

The second element is the Font Size.

Options are:

<b>*SCALE</b>	CoolSpools selects an appropriate font size based on the font, CPI and LPI information in the spooled file. This method is the same as *CALC (see below) except in relation to fonts specified by FGID (font identifier), such as those defined with the DDS FONT keyword. In the case of fonts specified by FGID, when the font is reproduced in PDF by a mapping, the font point size is calculated based on
---------------	--

the LPI value and then condensed using a horizontal scaling to the appropriate CPI value. This usually reproduces the appearance of the original printed spooled file more closely than \*CALC.

This option is only available from CVTSPLPDF (where it is the default value) and CVTSPLSTMF.

**\*CALC**

CoolSpools selects an appropriate font size based on the font and CPI information in the spooled file.

This is the default for CVTSPLSTMF, CVTSPLHTML and CVTSPLRTF.

**font\_size**

Specify a font size in points to be applied to all text.

For CVTSPLXLS, the default is 10 points.

### Font types to embed

The third element is only relevant when \*EMBED is selected and is only available in relation to PDF output. It allows you to control which types of system i fonts are embedded in the resultant PDF.

Single options are:

**\*NONE**

No fonts are embedded. This value is not allowed if \*EMBED is specified for the first element of the FONT parameter.

**\*ALL**

CoolSpools will embed all font types which can currently be embedded.

Other options are:

**\*PSTYPE1**

Postscript Type 1 fonts (also called outline fonts on system i) will be embedded.

**\*CIDKEYED**

CID-keyed fonts (PostScript Type 0 fonts) will be embedded. These are DBCS fonts.

**\*RASTER**

Raster (bitmap) fonts are embedded.

**\*FONTID**

Fonts specified by a font identifier (e.g. by means of the DDS FONT keyword) are embedded. This is dependent on a suitable font resource object being available for embedding.

Please note that system i raster (bitmap) fonts are relatively low resolution (typically 240 or 300 pels per inch) and are imported into PDF in the form of bitmap images. When displayed on screen in Adobe Acrobat, these bitmaps can appear jagged and uneven and the presentation quality is generally rather poor. This is a feature of the font technology, not a deficiency on the part of CoolSpools. When printed, the PDF accurately reproduces the print quality of the system i font on the page.

**Example:**

**CVTSPLPDF      FROMFILE(SALES)...  
FONT(\*MAP)**

Here the sales report is converted to PDF format and CoolSpools will attempt to select suitable replacement fonts for those used in the spooled file.

**Example:**

**CVTSPLPDF      FROMFILE(SALES)...  
FONT(\*EMBED)**

The same report is converted, but where possible CoolSpools will embed a copy of each system i font in the resultant PDF.

**Example:**

**CVTSPLXLS      FROMFILE(SALES)...  
FONT(\*ARIAL 10)**

Here the same report is converted, but this time to Excel format, and 10-pt Arial will be used throughout.

CoolSpools will also assist you in improving the appearance of your reports in PDF, RTF and HTML format by providing messages in the job log to inform you about the font mappings it has made. If you are not satisfied with the appearance of your report in PDF, RTF or HTML, you should examine the job log of the job in which the command was executed, and locate any messages of the form:

Courier 10 substituted for font id 11

or Courier Bold 10 substituted for font resource C0S0CB10

These messages are intended to help you identify which font and font resource names need to be mapped. You can try alternative font mappings through the user-definable font mapping facility, implemented via the **CVTFONTID** and **CVTFNTRSC** parameters described below.

## FTP – FTP parameters

Parameter	<b>FTP</b>
Description	<b>FTP options to be used when TOSTMF(*FTP) specified</b>
Dependent on:	<b>TOSTMF(*FTP)</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF</b>
Migration notes	<b>Three new parameter elements have been added in the middle of this parameter for the new format-specific commands. If you have code which references this parameter, it will need to be modified before you migrate to the new release.</b>

The FTP parameter allows you to define parameters needed to transfer the output to an FTP server when TOSTMF(\*FTP) is specified.

There are two single values that can be specified:

<b><u>*NONE</u></b>	Indicates that you do not intend to use FTP. Invalid if TOSTMF(*FTP) specified.
<b>*EXITPGM</b>	The FTP parameters will be defined at run time by an exit program. The exit program should generate a CS_FTP01 structure.

### Remote system name/IP address

Specify the name of IP address of the system to which the data should be transmitted by FTP.

If you specify a name, the system must be able to resolve that name to an IP address either by means of a DNS (Domain Name Server) or by looking up the name in the system Host Table.

### Remote file path

Specify the full path where the output should be saved on the server. This should include both the name of the file to be created and the directory tree in which it should be saved.

Note that names on the server may be case-sensitive, especially if it is a UNIX system or similar, and may need to be enclosed in single quotes.

CoolSpools variables may be specified on this parameter element.

### Port number

The port number to use,

Options are:

<b><u>*FTP</u></b>	The default port for FTP (21) will be used.
<b>*SECURE</b>	The default port for secure FTP (990) will be used.
<b>Port_number</b>	A valid port number between 1 and 65535.

## Secure connection

Specifies the type of security mechanism to be used for protecting information transferred on the FTP control connection (which includes the password used to authenticate the session with the FTP server). Transport Layer Security (TLS) and Secure Sockets Layer (SSL) are compatible protocols which use encryption to protect data from being viewed during transmission and verify that data loss or corruption does not occur.

Options are:

<b><u>*NONE</u></b>	CoolSpools does not use encryption when connecting to the specified FTP server.
<b>*IMPLICIT</b>	CoolSpools immediately attempts to use TLS/SSL when connecting to the specified FTP server (without sending an AUTH subcommand to the server). If the server does not support implicit TLS/SSL on the specified port, or the TLS/SSL negotiation fails for any reason, the connection is closed.
<b>*SSL</b>	After connecting to the specified FTP server, CoolSpools sends an AUTH (authorization) subcommand requesting an SSL protected session. If the server does not support SSL, the connection is closed.
<b>*TLS</b>	After connecting to the specified FTP server, CoolSpools sends an AUTH (authorization) subcommand requesting a TLS protected session. If the server does not support TLS, the connection is closed.

## Data protection

Specifies the type of data protection to be used for information transferred on the FTP data connection. This connection is used to transfer file data and directory listings. The FTP protocol does not allow protection of the data connection if the control connection is not protected.

Note: This element controls the use of the PROT (protection) FTP server subcommand.

Options are:

<b>*DFT</b>	If the Secure Connection option specifies a protected control connection, *PRIVATE is used; otherwise, *CLEAR is used.
<b>*PRIVATE</b>	Information sent on the FTP data connection is encrypted. If the Secure Connection option specifies that the FTP control connection is not encrypted, *PRIVATE cannot be specified.
<b>*CLEAR</b>	Information sent on the FTP data connection is not encrypted.

## Remote user id

The user id to use when logging on. Names may be case sensitive and may need to be enclosed in single quotes.

## Remote password

The password to use when logging on. Passwords may be case sensitive and may need to be enclosed in single quotes.

The FTP parameter allows you to define parameters needed to transfer the output to an FTP server when TOSTMF(\*FTP) is specified.

See the next element for details of how to supply this password in a scrambled form to avoid having to hold passwords in plain text form in source code.

When prompting the command, if you need to enlarge the size of this parameter element to allow specification of a hex string, enter an ampersand (&) then press return and OS/400 will increase the size the field.

If you need to enter a hex string, use the form X'0123456789ABCDEF' etc.

## Encrypted password supplied

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

**\*NO**

The password supplied on the previous element is in plain text format and not scrambled.

**\*YES**

The password supplied on the previous element is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used.

## Set permissions

Determines whether CoolSpools sets permissions on the file on the FTP server.

There is a single value:

**\*NO**

CoolSpools does not set permissions on the FTP server.

Other values:

## Owner permissions

Determines the owner permissions. Options are:

**\*R**

Read only

**\*W**

Write only

**\*X**

Execute only

**\*RW**

Read and write

<b>*RX</b>	Read and execute
<b>*WX</b>	Write and execute
<b>*RWX</b>	Read, write and execute (all)
<b>*NONE</b>	No authority

### ***Group permissions***

Determines the permissions for the group. Options are as per “Owner permissions”.

### ***Public permissions***

Determines the public permissions. Options are as per “Owner permissions”.

### ***Example:***

```
CVTSPLPDF      FROMFILE(SALES)...  
                TOSTMF(*FTP)  
                FTP(SalesSvr '/Sales/Sales.pdf' *FTP 'BILL' 'soccer')
```

The sales report is converted to FTP and the output is sent directly to a server known to the system as “SalesSvr” by FTP. The file will be saved in the “Sales” directory as “Sales.pdf”. The port number will be 21. The connection will be established by logging on as BILL with the password “soccer”.



## HTML – HTML options

Parameter        **HTML**

Applies to        **CVTSPLSTMF, CVTSPLHTML**  
commands:

Dependent on:    **CVTSPLSTMF: PMTADLPARM(\*YES) and TOFMT(\*HTMLCSS)**

This parameter defines HTML-related options.

### Script file to include

CoolSpools variables may be specified on this parameter element.

Options are:

<b><u>*NONE</u></b>	No script file will be included.
Script_path	Specify a full IFS path specifying the name and location of a script file (e.g. javascript). CoolSpools will retrieve the contents of the script file and embed it in the HTML it generates.

### End of page marker

Determines how the end of a page is indicated.

<b><u>*NONE</u></b>	There is no end-of-page marker.
*HR	The end of each page is denoted by a horizontal rule (<hr>).

### Top margin

Defines the top margin to be assumed when calculating the page length.

When printing the HTML file, the browser may apply a top margin. Unless CoolSpools allows for this top margin when calculating the position of the start of each page, page throws may not occur in the right place. You should therefore specify the top margin you wish to use here and then ensure that the same margin is specified in the page options when you print the file.

<b><u>0</u></b>	No top margin is assumed.
0.0-999.999	Specify the top margin to be assumed.

### Bottom margin

Defines the bottom margin to be assumed when calculating the page length. When printing the HTML file, the browser may apply a bottom margin. Unless CoolSpools allows for this bottom margin when calculating the position of the start of each page, page throws may not occur in the right place. You should therefore specify the bottom margin you wish to use here and then ensure that the same margin is specified in the page options when you print the file.

<b><u>0</u></b>	No bottom margin is assumed.
0.0-999.999	Specify the bottom margin to be assumed.

## Unit of measure

Defines the units in which the preceding two parameters are given.

<b><u>*INCH</u></b>	Inches
<b>*CM</b>	Centimeters
<b>*MM</b>	Millimeters

## HTML standard

Defines the HTML standard or style of the HTML code to be generated.

<b><u>*CSS2</u></b>	Cascading Style Sheets Version 2. This gives the best results but may not be supported by earlier versions of browsers and email clients.
<b>*HTML1</b>	HTML Version 1. This gives less good results but is universally supported by browsers and email clients.
<b>*PRE</b>	Pre-formatted text. Text is more likely to line up correctly but changes of font typefaces and sizes will be lost.

## Adjust for screen resolution

Whether font sizes are adjusted to take account of the screen resolution or not

<b><u>*NO</u></b>	No adjustment is made.
<b>*YES</b>	Font sizes are adjusted.

## Screen resolution (DPI)

The screen resolution to assume when font sizes are adjusted.

<b><u>96</u></b>	96 DPI. The Windows default.
<b>72</b>	72 DPI. The mac default.

## Convert images and graphics

Whether CoolSpools converts images and graphics in the spooled file.

<b><u>*YES</u></b>	CoolSpools will attempt to reproduce images and graphics in the spooled file.  CoolSpools will attempt to reproduce images in the spooled file by converting them to an HTML-compatible format (JPEG).
<b>*NO</b>	CoolSpools does not convert images and line graphics.

## ***INCLFILE – Include image files***

Parameter	<b>INCLFILE</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLPDF</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and TOFMT(*PDF)</b>

The INCLFILE parameter relates only to PDF output.

This parameter allows you to specify up to 20 stream files which are to be included in the PDF stream file when it is created.

This option can be used for various purposes:

- including a company logo, watermark or other graphic to enhance the appearance of largely textual report
- including a scanned image or other graphic of your pre-printed stationery in the PDF file so that it reproduces the appearance of the printed form exactly.

The included image files(s) must be in JPEG or GIF format.

CoolSpools needs to be able to access the image file at run time. The file must therefore be located either on the system itself or at a location that can be accessed through an IFS path name.

The default value is the single value **\*NONE**, which indicates that no such files are to be included.

A large number of additional elements are available. However, in the majority of cases only one or two elements of these will need to be defined.

### **Included file name**

The IFS path name identifying the file to be included.

CoolSpools will use this path to locate the file at the time the command is executed. If it cannot be located, or if the file is not in JPEG or GIF format, an error will be reported.

Refer to the TOSTMF parameter for further details of how to define an IFS path name.

CoolSpools variables may be specified on this parameter element.

### **Included file format**

There are two possibilities:

#### **\*JPG**

The file specified on the previous parameter is a JPEG.

#### **\*GIF**

The file specified on the previous parameter is a GIF.

### **Inclusion method**

The method by which the image file is included in the PDF.

The only possibility is now:

## **\*EMBEDDED**

The image file is embedded in the PDF file that is created.

This approach has the advantage that you have only a single file to manage or distribute, and the image file is guaranteed to be available when the PDF file is opened. However, it may significantly increase the size of the resulting PDF file.

The previously supported value \*EXTERNAL is no longer available as Adobe has withdrawn this option.

### **Included on pages**

The pages on which the image should be included.

Options are:

<b><u>*ALL</u></b>	All pages.
<b>*ODD</b>	Odd-numbered pages only.
<b>*EVEN</b>	Even-numbered pages only.
<b>*FIRST</b>	The first page only.
<b>*LAST</b>	The last page only.
<b>*BFRLAST</b>	All pages before the last page.
<b>*AFTERFST</b>	All pages after the first page.
<b>*BACK</b>	An extra page is inserted after each page in the spooled file and the image is included on this extra page. This option is useful where you have a pre-printed form with information printed on the reverse. You can include this information in the PDF file on an additional page by using this option.
<b>*FRONT</b>	As with *BACK, an extra page is inserted after each page in the spooled file. Unlike *BACK, the image is included on the original page, not the inserted page.
<b>*HEADER</b>	An extra page is inserted at the start of each group of pages which forms a single PDF file and the image is included on this additional page. This option can be useful if you wish to have a header sheet at the start of the PDF file.
<b>*TRAILER</b>	An extra page is inserted at the end of each group of pages which forms a single PDF file and the image is included on this additional page. This option can be useful if you wish to have a trailer sheet at the end of the PDF file.
<b>*KEYABS</b>	The image is included if the key string ("Key string" parameter below) occurs on the page. The coordinates are interpreted as absolute coordinates.

<b>*KEYREL</b>	The image is included if the key string ("Key string" parameter below) occurs on the page. The coordinates are interpreted as relative coordinates, relative to the key string.
<b>*PAGNBR</b>	Specify the page number in the spooled file on which the image should appear on the "Include on page number" element below.
<b>*PDFODD</b>	The image is included on odd pages in the PDF file.
<b>*PDFEVEN</b>	The image is included on even pages in the PDF file.
<b>*PDFPAGNBR</b>	Specify the page number in the PDF where the image should appear on the "Include on page number" element below.

Page numbers in the PDF and page numbers in the spooled file are not necessarily the same, especially if splitting is done or pages are excluded using EXCLPAGNBR or EXCLPAGKEY.

Please note that when determining whether a page is odd- or even-numbered, CoolSpools uses the natural order of pages in the spooled file and takes no account of any internal page numbering. For example, if your spooled file has an unnumbered batch header sheet, and your page numbering starts at 1 on page 2 of the spooled file, CoolSpools will take no account of this and will count the first page odd, the second page even etc.

### **X coordinate**

The X coordinate (horizontal distance across from left to right) of the position on the page where the image should appear.

This is interpreted as an absolute position on the page unless a key string is specified, in which case (unless "Included on pages" is \*KEYABS), this is interpreted as relative to the start of the key string.

### **Y coordinate**

The Y coordinate (vertical distance down from top to bottom) of the position on the page where the image should appear.

This is interpreted as an absolute position on the page unless a key string is specified, in which case (unless "Included on pages" is \*KEYABS), this is interpreted as relative to the start of the key string.

### **Unit of measure**

Options for the unit of measure are:

<b>*MM</b>	Millimeters
<b>*CM</b>	Centimeters
<b>*INCH</b>	Inches

### **External reference**

No longer supported. Ignored and provided only for reasons of backwards compatibility.

## External reference type

No longer supported. Ignored and provided only for reasons of backwards compatibility.

## Scale factor

The scaling factor (default 1.00 i.e. no scaling). This allows you to expand and contract the size of the JPG image as it appears in the PDF file.

## Rotation angle (degrees)

The rotation angle to be applied to the image when it is included.

The rotation angle can be used to ensure that the orientation of the included image is correct when viewed in Acrobat, for example where the page itself is rotated.

Options are:

<b><u>0</u></b>	No rotation is applied.
<b>90</b>	A 90-degree rotation is applied.
<b>180</b>	A 180-degree rotation is applied.
<b>270</b>	A 270-degree rotation is applied.
<b>360</b>	A 360-degree rotation is applied.

## Key string

How to interpret the X and Y coordinates defined earlier.

If the image you wish to include should always appear in the same, absolute, fixed position on the page, you should specify \*ABS for this element (this is the default value).

However, if the image position needs to vary, one option is to specify the location of the image relative to a piece of text (the "key string") on the page.

If you specify \*KEYABS or \*KEYREL for the "Included on pages" option above, this element defines the key string to be checked for when determining whether the image should appear on the page.

If you specify \*KEYABS, the image will appear at the position specified by the X and Y coordinates above.

If you specify \*KEYREL, the image will appear at the offset specified by the X and Y coordinates above relative to the position of the key string.

Options are:

<b><u>*ABS</u></b>	The X and Y coordinates defined above are interpreted as absolute coordinates, not relative to a key string.
<b>Key_string</b>	<p>If "Include on pages" is not *KEYABS or *KEYREL, the X and Y coordinates defined above are interpreted as offsets relative to the start of this key string.</p> <p>If "Include on pages" is *KEYREL, the image only appears if the key string occurs on the page and the X and Y coordinates defined above are</p>

interpreted as offsets relative to the start of this key string.

If "Include on pages" is \*KEYABS, the image only appears if the key string occurs on the page and the X and Y coordinates defined above are interpreted as absolute coordinates on the page.

### Key string occurrence

Which occurrence of the key string on the page determines the positioning of the image.

If the key string occurs more than once on the page, you can specify which occurrence to use on this parameter.

Options are:

**\*FIRST**

The first occurrence of the key string is the one that determines the positioning of the image.

**Occurrence**

Specify an occurrence between 1 and 999.

### Key string action

Whether the key string is included in the output or deleted.

If you have included the key string in the spooled file simply to indicate the location where an image should be positioned, you can ensure that it is not visible in the final PDF file by telling CoolSpools to remove it.

Options are:

**\*KEEP**

Keep the key string in the output.

**\*REMOVE**

Remove the key string from the output.

### Include on page number

When \*PAGNBR or \*PDFPAGNBR is specified for "Included on pages" above, specify the actual page number on which to include the image here.

Options are:

**\*NONE**

Neither \*PAGNBR nor \*PDFPAGNBR was specified for "Included on pages" above.

**page\_number**

When \*PAGNBR is specified for "Included on pages" above, specify the page number in the original spooled file where the image should appear. The image will appear on the corresponding page in the PDF.

When \*PDFPAGNBR is specified for "Included on pages" above, specify the page number in the PDF where the image should appear.

**Example:**

```
CVTSPLPDF      FROMFILE(SALES)...  
                INCLFILE( ('/images/salesform.jpg'  
                          *EMBEDDED  
                          *ALL  
                          0 0 *INCH)
```



## **MARGINS - PDF margins and alignment**

Parameter	<b>MARGINS</b>
Applies to commands:	<b>CVTSPLPDF, CVTSPLSTMF</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and TOFMT(*PDF)</b>

The MARGINS (Additional margins) parameter relates only to PDF output.

This parameter allows you to specify additional margins or make adjustments to the alignment of text within the spooled file.

If no value is specified on this parameter, CoolSpools will reproduce the margins defined in the spooled file and align text exactly as specified in the printer data stream, which may not necessarily be the way the data appears on the page when printed from your printer. You can use this parameter to make slight adjustments to try to ensure that your PDF reproduces the appearance of the printed page as closely as possible.

### **Left**

This first element allows you to define an additional left margin for the PDF. This option may be useful where otherwise data appears too close to the left edge of the page to be easily read or printed on a PC printer with a no-print border.

Please note that if the spooled file is rotated, the term “left” refers to the page prior to rotation, in other words the margin may appear at the top, bottom or on the right, depending on the angle through which the page is rotated.

Specify a value between -99.999 and 99.999. This value is measured in the units defined on the “Unit of measure” element of this parameter. The default is 0, i.e. no additional margin.

### **Top**

This second element allows you to define an additional top margin for the PDF. This option may be useful where otherwise data appears too close to the top edge of the page to be easily read or printed on a PC printer with a no-print border.

Please note that if the spooled file is rotated, the term “top” refers to the page prior to rotation, in other words the margin may appear at the bottom or on the left or right, depending on the angle through which the page is rotated.

Specify a value between -99.999 and 99.999. This value is measured in the units defined on the “Unit of measure” element of this parameter. The default is 0, i.e. no additional margin.

### **Overlays Left**

This third element allows you to define an additional left margin or horizontal shift for overlays only in the PDF.

Text, images and other items that are held in overlays will be shifted horizontally by the amount you specify on this parameter. A positive value will cause data to shift to the right and a negative value will cause data to shift to the left.

This may be useful where the PDF you create with the default parameters appears to have the overlays slightly misaligned from the other content of the spooled file. This can occur, for example, where your printer is for some reason positioning the overlay differently from what would be expected based on the instructions contained in the printer data stream alone (perhaps because data is falling in the no-print border and could not otherwise be printed), and you have programmed your application in such a way as to produce the correct results on that particular printer.

Please note that if the spooled file is rotated, the term “left” refers to the page prior to rotation, in other words the shift may appear at the top, bottom or on the right, depending on the angle through which the page is rotated.

Specify a value between -99.999 and 99.999. This value is measured in the units defined on the “Unit of measure” element of this parameter. The default is 0, i.e. overlay horizontal shift is required.

### **Overlays Top**

This fourth element allows you to define an additional top margin or vertical shift for overlays only in the PDF.

Text, images and other items that are held in overlays will be shifted vertically by the amount you specify on this parameter. A positive value will cause data to shift down the page and a negative value will cause data to shift up the page.

This may be may be useful where the PDF you create with the default parameters appears to have the overlays slightly misaligned from the other content of the spooled file. This can occur, for example, where your printer is for some reason positioning the overlay differently from what would be expected based on the instructions contained in the printer data stream alone (perhaps because data is falling in the no-print border and could not otherwise be printed), and you have programmed your application in such a way as to produce the correct results on that particular printer.

Please note that if the spooled file is rotated, the term “top” refers to the page prior to rotation, in other words the shift may appear at the bottom or to the left or right, depending on the angle through which the page is rotated.

Specify a value between -99.999 and 99.999. This value is measured in the units defined on the “Unit of measure” element of this parameter. The default is 0, i.e. overlay vertical shift is required.

### **Rotation left margin shift**

This fifth element allows you to define a distance across the page by which, in the context of a rotation, the contents of the spooled file will be shifted from the position at which they would normally be expected to be found.

When pages are rotated, especially when auto-rotation and/or COR (Computer Output Reduction) is applied, certain printers may shift data down the page in order to avoid printing data in the no-print border. CoolSpools cannot know if your particular printer will do this or not. As a result, the PDF that you create may not reproduce the appearance of the printed page 100% accurately. Where this occurs, applying a shift on this element or the next can usually correct the situation.

Specify a value between -99.999 and 99.999. This value is measured in the units defined on the “Unit of measure” element of this parameter.

The default value is the special value \*CALC, which tells CoolSpools to decide whether and how large a shift is required based on the information available to it.

### Rotation top margin shift

This sixth element allows you to define a distance down the page by which, in the context of a rotation, the contents of the spooled file will be shifted from the position at which they would normally be expected to be found.

When pages are rotated, especially when auto-rotation and/or COR (Computer Output Reduction) is applied, certain printers may shift data down the page in order to avoid printing data in the no-print border. CoolSpools cannot know if your particular printer will do this or not. As a result, the PDF that you create may not reproduce the appearance of the printed page 100% accurately. Where this occurs, applying a shift on this element or the previous one can usually correct the situation.

The default value is the special value \*CALC, which tells CoolSpools to decide whether and how large a shift is required based on the information available to it.

### Unit of measure

This seventh element defines the units in which the preceding options are measured.

Options for the unit of measure are:

<b>*MM</b>	Millimeters
<b>*CM</b>	Centimeters
<b>*INCH</b>	Inches

### Increase page size?

This eighth element indicates whether, when an additional margin is applied to the PDF, the size of the page should be modified to accommodate the additional margin, or whether data is simply shifted across the page.

Options are:

<b><u>*NO</u></b>	The page size remains the same and the data is simply shifted across or up/down the page. If there is insufficient room on the page, data may fall off the edge.
<b>*YES</b>	The page size is increased by the amount of the margin.

### Example:

```
CVTSPLPDF FROMFILE(SALES)...  
MARGINS(1 0 0 0 *CALC *CALC *INCH)
```

The spooled file is converted to PDF format with an additional 1 inch left margin but no additional top margin.

When your document prints in landscape mode as a result of a rotation, especially an automatic rotation triggered by the spooled file attribute PAGRTT(\*AUTO), PAGRTT(\*COR) or PAGRTT(\*DEVD), you may find that text in your spooled file appears slightly out of alignment with overlays and graphics. This is the result of your printer applying a margin to the document as it rotates it.

We recommend that you try adjusting the values of the “Rotation shift” elements of this parameter until your document alignment is correct.

## OPTIONS – Miscellaneous command options

Parameter	OPTIONS
Applies to commands:	<b>CVTSPLCSV , CVTSPLDLM, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLSPLF, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF</b>
Dependent on:	<b>None</b>

The OPTIONS parameter is intended to provide a convenient place where miscellaneous command options can be added to commands without unduly cluttering the command with new parameters.

There are few options available at present but it is anticipated that new options will be added over time as the need arises.

Each command option consists of a key identifying the type of option being specified and a value element which enables the corresponding option value to be defined.

### Option key

Options are:

<b>*DIAGNOSTIC</b>	Whether the command should be run in diagnostic mode or not. You should only use this option is instructed to do so by ariadne as in diagnostic mode the command may generate large volumes of logging data, produce inefficient output (e.g. uncompressed PDFs) and take an extended time to run.
<b>*CRTDIR</b>	Whether any directories in the path specified on the TOSTMF parameter will be created if they do not already exist.
<b>*RSCDIR</b>	Specifies the name of the resource directory where PCL resources (PCL macros and soft fonts) will be looked for. This option replaces the previous RSCDIR parameter which has now been retired.
<b>*ADOBEPATH</b>	The path to be used to locate Adobe Reader on the PC when the CVTSPLPDF ... TOSTMF(*VIEW) or TOSTMF(*PRINT) option is used. See the TOSTMF parameter for further details.
<b>*TMPPATH</b>	The path to be used to the /tmp directory on the system i when the CVTSPLPDF ... TOSTMF(*VIEW) or TOSTMF(*PRINT) option is used. See the TOSTMF parameter for further details.

### Option value

A value corresponding to the key above.

The list of possible valid values is dependent on the corresponding key.

Key	Valid values	Description	Notes
<b>*DIAGNOSTIC</b>	*NO	Do not run in diagnostic mode (default)	Diagnostic mode should only be used when instructed by ariadne. Performance degradation and the creation of abnormally large output files could result.
	*YES	Run in diagnostic mode	
<b>*CRTDIR</b>	*NO	Directories in the directory path specified on the TOSTMF parameter must already exist. If they do not, an error will occur (default);	You can set the system default action by creating an environment variable called CS_CRT_DIR_PATH set to either *YES or *NO.
	*YES	Directories in the directory path specified on the TOSTMF parameter will be created if they do not already exist.	
<b>*RSCDIR</b>	directory_path	Specify the directory path where the system should look for PCL resources such as macros and soft fonts.  The default is *TODIR, which indicates that the system should look for resources in the directory into which the output file is being created;  *CURDIR can also be specified and indicates that the system should look in the current	Replaces the RSCDIR parameter, which has now been retired.  The system default can also be specified by creating an environment variable called CS_RSC_DIR with its value set to the directory path where resources can be located or one of the special values *TODIR or *CURDIR.

		directory of the job.	
<b>*ADOBEPATH</b>	A full PC file path to the Adobe Reader program on the PC	The path used to locate Adobe Reader on the PC when CVTSPLPDF ... TOSTMF(*VIEW) or TOSTMF(*PRINT) is used	e.g. <b>c:\Program Files\Adobe\Reader 9.0\Reader\AcroRd32.exe</b>  would be a typical path for Adobe Reader 9.  The path will vary depending on the version of Adobe Reader installed.
<b>*TMPPATH</b>	A full PC path to the /tmp directory on the system i	Used to provide Adobe Reader on the PC with a path back to the temporary PDF file on the system i. This will therefore typically be a NetServer file share path.	e.g. '\\192.168.0.1\root\tmp'  where 192.168.0.1 is the IP address of the system i and "root" is the name of a NetServer file share that shares the root of the IFS.

Example:

### **CVTSPLPDF**

...  
**TOSTMF('/dir1/subdir1/subdir2/subdir3/subdir4/subdir5/output.pdf')**  
**OPTIONS((\*CRTDIR \*YES) (\*RSCDIR '/resources'))**

Directories in the path specified on TOSTMF will be created if they do not already exist. PCL resources will be located in the directory called **/resources**.

## **OUTPTY – Output priority**

Parameter	<b>OUTPTY</b>
Applies to commands:	<b>CVTSPLSPLF</b>
Dependent on:	<b>None</b>

When creating spooled files from an original spooled file with CVTSPLSPLF, this option defines the output priority to be assigned to the new spooled files that are created.

Options are:

**\*JOB**

The output queue priority defined by the OUTPTY attribute of the job running the command is used.

**\*SPLF**

The output priority of the original spooled file is used. However, if this output priority exceeds the maximum output priority allowed for the user who is restoring the spooled file, the restore operation will fail. This error can be avoided by specifying a different (lower) output priority on this parameter.

**Output\_pty**

Specify the priority to be used (1-9).



## **OUTQ – Output queue**

Parameter	<b>OUTQ</b>
Applies to commands:	<b>CVTSPLSPLF</b>
Dependent on:	<b>None</b>

When creating spooled files from an original spooled file with CVTSPLSPLF, this option defines the output queue on which the new spooled files should be created.

Options are:

<b><u>*JOB</u></b>	The output queue defined by the OUTQ attribute of the job running the command is used.
<b>*SPLF</b>	The output queue on which the original spooled file is located is used.
<b>Outq_name</b>	Specify the fully qualified name of the output queue to use.

## **OWNER – New spooled file owner**

Parameter	<b>OWNER</b>
Applies to commands:	<b>CVTSPLSPLF</b>
Dependent on:	<b>None</b>

When creating spooled files from an original spooled file with CVTSPLSPLF, this option defines the user profile that should own the new spooled files that are created.

Options are:

<b><u>*CURRENT</u></b>	The spooled files are owned by the user running the command.
<b>*SPLF</b>	The owner should be the same as the owner of the original spooled file. If the owner's user profile does not exist on the system, an error will occur.
<b>User_profile</b>	Specify the user profile that should own the new spooled files.

## PAGEOPTION – Page options

Parameter	<b>PAGEOPTION</b>
Applies to commands:	<b>CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLTXT,</b>
Dependent on:	<b>None</b>

This parameter defines various options related to pages and consists of three elements:

- **Auto-rotation in effect?**
- **Horizontal scaling**
- **Vertical scaling**

The following single option is the default value:

**\*CALC** CoolSpools will attempt to calculate the best orientation and scaling based on the spooled file attributes.

### Auto-rotation in effect?

This element determines whether automatic rotation and/or Computer Output Reduction (COR) are applied, simulating the effects of the PAGRTT(\*AUTO), PAGRTT(\*COR) or PAGRTT(\*DEVD) attribute on certain printers.

If your spooled file has the attribute PAGRTT(\*AUTO), PAGRTT(\*COR) or PAGRTT(\*DEVD), automatic page rotation will occur when the spooled file is printed on a printer and the spooled file does not fit on the page in its standard orientation. For example, if the attributes of your spooled file indicate that it is 132 columns wide at 10 CPI and 66 lines long at 6 LPI (i.e. 13.2 inches by 11 inches), and you print it to a printer which uses letter or A4 paper, the spooled file is too large to fit on the paper. Your printer will automatically reduce the size of the spooled file data (COR) and rotate the spooled file data (auto-rotation) in order to make it fit the paper.

Unlike the CVTSPLSTMF command, which does not implement an automatic page rotation when a spooled file has PAGRTT(\*AUTO), PAGRTT(\*COR) or PAGRTT(\*DEVD), the format-specific commands CVTSPLPDF, CVTSPLHTML and CVTSPLRTF will by default attempt to reproduce the behavior of the majority of modern printers will automatically rotate and, if necessary, scale down the contents of the spooled file.

Please note that the paper size specified on the first two elements of this parameter is interpreted as the shape and format of the paper **prior to rotation**. For example, if your document prints on letter cut sheet paper (11 x 8.5 inches), you should specify PAGESIZE(\*LETTER \*PORTRAIT) even if the document prints in landscape mode, since the paper is physically printed in portrait mode and the document contents rotated to fit on it.

Options are:

<b>*SPLF</b>	CoolSpools will itself decide whether to apply auto-rotation and/or COR.
<b>*NO</b>	Auto-rotation is not applied.
<b>*YES</b>	Auto-rotation is applied but NOT COR.
<b>*COR</b>	Auto-rotation and COR will both be applied.

## Horizontal scaling

## Vertical scaling

The second and third elements of the PAGESIZE parameter of the CVTSPLPDF, CVTSPLHTML and CVTSPLRTF commands indicate the horizontal and vertical scaling to be applied to data in the spooled file to make it fit the page.

If you are changing the page size from that defined in the spooled file (e.g. to convert a 13.2 x 11 inch spooled file to a 11 x 8.5 inch PDF, suitable for printing on a PC printer), you may need to scale the contents of the spooled file to get the best fit to the new page size and the best possible readability on screen.

Options are:

<b>*NONE</b>	No scaling is applied.
<b>*CALC</b>	If the conditions for COR (Computer Output Reduction) are met, CoolSpools will calculate an appropriate scaling based on the dimensions of the original spooled file, the new page size and any margins requested.
<b>*FITPAGE</b>	Irrespective of whether the conditions for COR (Computer Output Reduction) are met, CoolSpools will calculate a scaling factor which will fit the spooled file contents to the paper size and orientation selected on the PAGESIZE parameter.
<b>Scaling_factor</b>	A scaling factor between 00.01 and 99.99, where 1.00 means that no scaling occurs. For example, a scaling of 0.5 will halve the width or length of the spooled file contents and a scaling of 2.0 will double the width or length of the spooled file contents.

## PAGESIZE - Page Size

Parameter	<b>PAGESIZE</b>
Applies to commands:	<b>CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLTXT, CVTSPLXLS, CVTSPLXL</b>
Dependent on:	<b>None</b>

Specifies the page size to use.

The following single options are available for both elements:

### **\*CALC**

CoolSpools will assume a paper size based on the country code of the current job according to the following table:

Country Code	Paper Size
US	*LETTER
CA	*LETTER
All others	*A4

### **\*SPLF**

CoolSpools will use the paper size specified in the attributes or data stream content of the spooled file. These normally correspond to the page width and length specified on the CRTPTF command when the printer file was created.

### **\*CUSTOM**

You will specify the precise page size on the CUSTOMPAGE parameter. This option is useful if you want to use a paper size not provided as one of the standard options listed below.

### **\*DEVD**

CoolSpools will derive the page size from the attributes of the printer device specified on the PRTDEV parameter.

## Paper Size

This element specifies the paper size which CoolSpools will simulate when creating the output file.

<b>*A3</b>	420 x 297 mm
<b>*A4</b>	297 x 210 mm
<b>*A5</b>	210 x 148 mm
<b>*B3</b>	364 x 257 mm
<b>*B4</b>	257 x 182 mm
<b>*LEGAL</b>	14 x 8.5 in.
<b>*LETTER</b>	11 x 8.5 in.
<b>*EXEC</b>	10.5 x 7.25 in.
<b>*LEDGER</b>	17 x 11 in

## Orientation

The second element of this parameter controls the orientation of the page as reproduced in the stream file.

Options are:

<b><u>*SPLF</u></b>	The orientation is derived from the dimensions of the spooled file. If the width of the spooled file exceeds the length of the spooled file, the page will appear in landscape mode, otherwise it will be in portrait mode. This is the only value permitted if the page size element is *SPLF or *CUSTOM.
<b>*LANDSCAPE</b>	Landscape mode.
<b>*PORTRAIT</b>	Portrait mode.

**Please note that when specifying the orientation for a page which will be rotated, you should specify the orientation of the unrotated page.**

For example, if your report is printed in landscape mode on an A4 printer by means of page rotation, you should specify **PAGESIZE(\*A4 \*PORTRAIT)**, not **PAGESIZE(\*A4 \*LANDSCAPE)**. This is because, in reality, the spooled file orientation is portrait, but text is printed rotated through 90 degrees to give the effect of landscape printing.

The CVTSPLXLS command has some additional options related to the printing of Excel files. For the CVTSPLXL command, see the XLSPRINT parameter instead for these options.

## Page scaling method

How to scale the data to fit the page.

Options are:

<b>*NONE</b>	No scaling is applied.
<b>*FIT</b>	The data is fitted to a specified number of pages wide by a specified number of pages tall. The number of pages wide and tall are given on the 5th and 6th elements of this parameter.
<b>*ADJUST</b>	The data is adjusted to fit the page by scaling it by a percentage. The percentage is specified on the 4th element of this parameter.

## Percentage adjustment

The percentage scaling to apply when \*ADJUST is specified for the scaling method.

Options are:

100	100% (no change).
0-400	Specify the scaling percentage.

## Fit to pages wide

The number of pages wide (horizontal scaling) to which the data is fitted.

Options are:

*AUTO	Excel will calculate the required number of pages wide.
-------	---

0-65535	Specify the number of pages.
---------	------------------------------

### **Fit to pages tall**

The number of pages tall (vertical scaling) to which the data is fitted.

Options are:

*AUTO	Excel will calculate the required number of pages wide.
-------	---

0-65535	Specify the number of pages.
---------	------------------------------

### **Print gridlines**

Whether gridlines are printed.

Options are:

*NO	Gridlines are not printed.
-----	----------------------------

*YES	Gridlines are printed.
------	------------------------

## PASSWORD – PDF Security

Parameter	<b>PASSWORD</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLPDF</b>

The PASSWORD (PDF passwords) parameter allows you to password-protect your PDF files and/or restrict the operations that can be performed on them.

Password protecting a PDF file allows you to e-mail it safe in the knowledge that, if the e-mail goes astray or is intercepted, it will not be possible to open the PDF file without the necessary password. Similarly, sensitive business documents can be stored safely on your company server and will not be accessible by anyone who has not been given the passwords to open them.

Restricting access rights to a file allows you to control what operations can be performed on it, for example whether it can be modified or printed or text copied from it). You can do this in conjunction with a password or without one.

PDF passwords are implemented using Adobe's standard encryption method. This highly secure encryption technique employs the RSA Data Security, Inc. MD5 Message-Digest algorithm (described in Internet RFC 1321, The MD5 Message-Digest Algorithm) and the public-domain ArcFour encryption algorithm.

Prior to version 1.4 of the PDF specification, PDF's standard encryption handler limited the encryption key to 5 bytes (40 bits) in length, in accordance with U.S. cryptographic export requirements, and 40-bit encryption is still the default. However, you can also use the \*PWD128BIT and \*RST128BIT options to request 128-bit encryption.

A PDF file may be allocated an "owner" password and a "user" password.

The "owner" password gives full access to all features of the document, i.e. entering the "owner" password in Adobe Acrobat (as opposed to Acrobat Reader) will enable you to modify, copy, print and annotate the document.

The "user" password gives either full access or limited access to the document, depending on the user privileges that were granted when the file was created.

The privileges that can be granted are:

- whether the document may be printed
- whether text in the document may be copied
- whether the document can be modified (requires Acrobat)
- whether notes can be added to the document (requires Acrobat)



It is also possible to restrict any or all of the above functions without requiring a password to be entered. When that is done, no one can perform any of the above functions on the file, even the owner.

## **PLEASE NOTE THAT PASSWORDS ARE CASE-SENSITIVE.**

If you forget your password, you will not be able to open your document.

ariadne software takes no responsibility for documents that cannot be opened as a result of a lost or forgotten password and has no means to recover documents that have become unusable as a result.

There is one single value:

### **\*NO**

The PDF file will not be password protected and no restrictions will be applied to the operations that can be performed on it.

## **Password protect PDF file?**

The first element indicates whether you wish to password protect the document.

Options are:

<b>*YES</b>	At least a user password is needed to open the file
<b>*EXITPGM</b>	A pre-file creation exit program will be used to supply the password(s).
<b>*RESTRICT</b>	Do not require a password to open the file, but prevent one or more operations from being applied to the file (printing, modification, annotation or copying of text).
<b>*PWD40BIT</b>	Equivalent to *YES. 40-bit encryption is used.
<b>*RST40BIT</b>	Equivalent to *RESTRICT. 40-bit encryption is used.
<b>*PWD128BIT</b>	Equivalent to *YES, except that 128-bit encryption is used.
<b>*RST128BIT</b>	Equivalent to *RESTRICT, except that 128-bit encryption is used.

## **User password**

The second element is the user password. If \*YES is specified for the previous element, a user password must be entered (cannot be left blank). If \*RESTRICT is specified for the previous element, a user password may not be entered (as \*RESTRICT indicates that the file should have restricted access rights without a password).

The password can be any string of characters and numbers. The minimum length is 1 character and the maximum is 32. The password is case-sensitive.

## Owner password

The third element is the owner password. If no owner password is entered (i.e. it is left blank), the document will not have an owner password. This means that it will not be possible for anyone to perform any actions not permitted according to the user rights defined in the following parameters.

If \*RESTRICT is specified for the first element, an owner password may not be entered (as \*RESTRICT indicates that the file should have restricted access rights without a password).

The owner password can be any string of characters and numbers. The minimum length is 1 character and the maximum is 32. The password is case-sensitive.

Please note that if \*EXITPGM is specified on the first element of this parameter, any values typed for the user or owner password in the second and third elements of this parameter are only used if the exit program returns blanks for the corresponding password.

The next four elements to this parameter control the rights granted when the document is opened by entering the "user" password.

These are:

- Allow printing?
- Allow modifications?
- Allow copying of text?
- Allow annotation?

All of these parameters take the following form:

<b><u>*YES</u></b>	The action is permitted
<b>*NO</b>	The action is not permitted

If \*RESTRICT is specified for the first parameter element, at least one of the above four elements must be \*NO.

## Encrypted password supplied

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

**\*NO**

The password supplied on the previous element is in plain text format and not scrambled.

**\*YES**

The password supplied on the previous is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used.

## PDF – PDF options

Parameter	<b>PDF</b>
Applies to commands:	<b>CVTSPLPDF</b>
Dependent on:	<b>None</b>

This parameter specifies options for PDFs.

### PDF viewer type

Indicates the type of viewer you intend to open the resultant PDF file with.

Options are:

- \*WINDOWS**                      A Microsoft ® Windows PDF viewer will be used
- \*OTHER**                         A viewer other than a Microsoft ® Windows viewer will be used.

This option is provided largely for reasons of backwards compatibility and has no effect on the output generated.

### Initial bookmark action

This element allows you to specify whether any PDF bookmarks that have been generated when the file was created should be visible when the report is first opened or whether the user will need to select the option to display them from the menu:

Options are:

- \*SHOW**                         Show the bookmarks when the document is opened
- \*HIDE**                         Do not show the bookmarks when the document is opened.

It should be noted that some improvement in the time taken to open a PDF file can be obtained if BMARKACT(\*HIDE) is selected. However, your users will need to make the bookmarks visible before they can be used to navigate around the document.

### Example:

```
CVTSPLSTMF      FROMFILE(SALES)...  
                  BOOKMARK(*PAGNBR)  
                  BMARKACT(*HIDE)
```

The sales report is converted to PDF format and bookmarks are generated for each page of the report. However, the bookmarks are not displayed when the PDF file is first opened.

### Initial zoom when PDF opened

This element allows you to define an initial magnification to be used when a PDF file is first opened.

Options are:

<b><u>*PDFDFT</u></b>	The default magnification defined in your viewer options is used.
<b>*FITWDW</b>	Adjust the magnification so that the entire page just fits in the viewer window.
<b>*FITWIDTH</b>	Adjust the magnification so that the width of the page just fits in the viewer window.
<b>*FITVIS</b>	Adjust the magnification so that the text and graphics on the page fit in the viewer window.
<b>*ACTUAL</b>	Adjust the magnification so that the page is viewed at its actual size.
<b>Zoom factor</b>	The percentage magnification to apply.

### ***PDF keywords for indexing***

This element allows you to define a set of keywords to be included in the Document Info section of the PDF file. These can be used by indexing and document management applications.

Specify the keywords as a single character string with keywords separated by a comma or semicolon.

### ***Data Compression***

Here you can indicate whether stream data in a PDF files should be compressed, and, if so, what compression level to apply.

Data compression is a trade-off between compression ratio and time. The higher the compression ratio that is attempted, the longer the data will take to compress. The options below enable you to select whether you want a high compression ratio (giving the smallest PDF files but taking longer to create) or the fastest conversion time (producing larger PDF files but running more quickly).

Options are:

<b><u>*OPT</u></b>	Stream data in PDF files is compressed. The level of compression that is applied provides a good degree of data compression while not taking unduly long to compress.
<b>*YES</b>	Provided for compatibility with previous releases. Equivalent to *OPT.
<b>*NONE</b>	Stream data in PDF files is not compressed. The resultant PDF files will be significantly larger than if data compression was applied, but will take less time to create.
<b>*NO</b>	Provided for compatibility with previous releases. Equivalent to *NONE.
<b>*MAX</b>	The maximum possible level of data compression is applied. The PDF files will be as small as possible, but will take the longest time to create.
<b>*HIGHER</b>	A compression ratio higher than *HIGH but less than *MAX.

<b>*HIGH</b>	A compression ratio higher than *OPT but less than *HIGHER.
<b>*FAST</b>	A compression ratio less than *OPT but higher than *FASTER.
<b>*FASTER</b>	A compression ratio less than *FAST but higher than *FASTEST.
<b>*FASTEST</b>	The lowest and therefore fastest level of data compression.

### ***Fast Web View***

Determines whether the PDF "Fast Web View" option is implemented.

This option can improve the time taken to open PDF files across a network.

Options are:

<b><u>*YES</u></b>	Fast web view is applied.
<b>*NO</b>	Fast web view is not applied.

### ***Rotated pages shown unrotated?***

Determines the appearance of PDFs that contain rotated pages or images.

<b><u>*YES</u></b>	If the page is rotated, or if auto-rotation is applied (see next element), the page will be automatically rotated back into the standard orientation for easier viewing.
<b>*NO</b>	The page, if rotated, is viewed in the rotated orientation.
<b>*PAG</b>	Equivalent to *YES. Whether the page is rotated in the PDF is dependent on whether the page is rotated in the spooled file or not.
<b>*OVL</b>	Whether the page is rotated in the PDF is dependent on whether the page overlay is rotated or not. For example, if the page rotation is 0, but the overlay rotation is 90 degrees, the PDF page will not be rotated if *PAG is specified for this parameter element, but will be rotated if *OVL is specified.
<b>*IMG</b>	Whether the page is rotated in the PDF is dependent on whether the first image in the page overlay is rotated or not. For example, if the first image in the page overlay has a rotation of 90 degrees, the PDF page will be rotated back through 90 degrees to compensate.
<b>*ROTATE</b>	The page is rotated through the angle specified on the next parameter element, irrespective of what rotations are defined in the spooled file.

### ***Rotation angle (degrees)***

The angle through which the page is rotated in the PDF when \*ROTATE is specified for the previous element.

Options are:

<b>*NONE</b>	No rotation is applied.
90	90 degrees.
180	180 degrees.
270	270 degrees.

### ***Compliant with PDF standard***

The PDF standard with which the generated PDF should comply.

Options are:

<b>*ADOBE</b>	Adobe's PDF Specification.
<b>*PDFA</b>	The PDF/A-1 standard. PDF/A-1 is an international standard document document file format for long-term preservation. It is also known as ISO 19005-1.

### ***Conformance level***

The level of conformance to the standard selected on the previous element.

Options are:

<b>*NONE</b>	No conformance level applies. This value is mandatory if the previous element has the value *ADOBE and is prohibited if the previous element has the value *PDFA.
<b>*LEVEL1A</b>	Level 1A conformance to the PDF/A-1 standard.
<b>*LEVEL1B</b>	Level 1B conformance to the PDF/A-1 standard.

### ***Language code***

A code denoting the language of the text in the PDF document.

Options are:

<b><u>*SYSVAL</u></b>	The language code is derived from the QLANGID system value.
<b>*JOB</b>	The language code is derived from the LANGID job attribute.
<b>language_code</b>	Enter a valid 2-character ISO language code e.g. FR=French, ES=Spanish.

### ***Country code***

A code used to qualify the language code specified on the previous element. For example, if the previous element has the value EN (English), this code can further qualify the language as US (en-US) or British (en-GB) English etc.

Options are:

<b><u>*SYSVAL</u></b>	The country code is derived from the QCNTRYID system value.
<b>*JOB</b>	The country code is derived from the CNTRYID job attribute.
<b>country_code</b>	Enter a valid 2-character ISO country e.g. GB=Great Britain, US=USA.

### ***Author***

Enter the name of the author of the document. Options are:

<b><u>*NONE</u></b>	No author is specified.
<b>character-value</b>	Specify the author's name.

### ***Subject***

The subject of the document.

Options are:

<b><u>*NONE</u></b>	No subject is specified.
<b>character-value</b>	Specify the subject of the document.

### ***Print number of copies***

Presets the number of copies to print option in the Print dialog.

Options are:

<b><u>*DFT</u></b>	The value is not preset in the Print dialog. When the PDF file is opened and the print dialog is displayed, the value for this print attribute will be set by the PDF reader application, not by the PDF file created by this command.
<b>*SPLF</b>	The value will be derived from the NBRCOPIES attribute of the spooled file.
<b>nbr_of_copies</b>	Specify the number of copies with which to preset the Print dialog.  Note that PDF only supports values between 1 and 5 and any other value will be ignored.

### ***Print scaling***

Presets the scaling option in the Print dialog.

Options are:

<b><u>*DFT</u></b>	The value is not preset in the Print dialog. When the PDF file is opened and the print dialog is displayed, the value for this print attribute will be set by the PDF reader application, not by the PDF file created by this command.
<b>*NONE</b>	The print dialog will be preset to select the option "No scaling"

### ***Choose source by PDF page size***



Presets the “choose source by PDF page size” option in the Print dialog.

Options are:

<b><u>*DFT</u></b>	The value is not preset in the Print dialog. When the PDF file is opened and the print dialog is displayed, the value for this print attribute will be set by the PDF reader application, not by the PDF file created by this command.
<b>*YES</b>	The option is preset to “Yes”
<b>*NO</b>	The option is preset to “No”

## PRTDEV – Printer device

Parameter	<b>PRTDEV</b>
Applies to commands:	<b>CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLTXT</b>
Dependent on:	<b>None</b>

This parameter allows you to specify the name of a printer device from which attributes will be when spooled file attributes are set to \*DEV D.

The results you get when you print a particular spooled file can be dependent on the model of printer that you use and the way that printer is configured. When creating PDFs and other file types, CoolSpools attempts to emulate a true IPDS printer configured to use the most commonly used settings, but if you are using a different type of printer or have selected other settings, the results you obtain with CoolSpools may not match what you were expecting.

This is particularly true if the printer you are using is configured to use IBM's Host Print Transform (HPT) as HPT implements a number of features slightly differently from a true IPDS printer, for example overlay positioning and margin handling. See this IBM document for further information on this topic: [http://www-912.ibm.com/n\\_dir/nas4apar.nsf/51d11a683a56a5cc862564c000763b23/65611dda1e4a84ca86256d08006bc80d?OpenDocument](http://www-912.ibm.com/n_dir/nas4apar.nsf/51d11a683a56a5cc862564c000763b23/65611dda1e4a84ca86256d08006bc80d?OpenDocument)

You can overcome some of these issues by using this parameter to tell CoolSpools which printer device your spooled file has been designed to print on and to specify some of the settings to assume.

### Device name

Options are:

<b><u>*SYSVAL</u></b>	The default printer device is used. If an environment variable called CS_DFT_PRT_DEV exists, its value will be interpreted as specifying the name of the default printer device to assume. Otherwise, the printer specified on the QPRTDEV system value is used.
<b>*HPT</b>	Emulate a generic Host Print Transform device.
<b>Device_name</b>	Specify the name of the printer device to be used.

CVTSPLPDF has some additional options:

### Margin offset down

Specify the vertical margin to assume at the top and bottom of each page.

Options are:

<b><u>*DEV D</u></b>	The printer device specified on the previous element is used to estimate the margin setting.
<b>margin</b>	The margin to assume, specified in the units defined below.

### Margin offset across

Specify the horizontal margin to assume to the left and right of each page.

Options are:

<b><u>*DEV</u></b>	The printer device specified on the previous element is used to estimate the margin setting.
<b>margin</b>	The margin to assume, specified in the units defined below.

### Margin offset unit

Specify the units used to define the margins above.

Options are:

<b><u>*IN</u></b>	Inches
<b>*CM</b>	Centimeters
<b>*MM</b>	Millimeters

## ***RPLXLSSHT– Replace Excel worksheet names***

Parameter	<b>RPLXLSSHT</b>
Applies to commands:	<b>CVTSPLXL</b>
Dependent on:	<b>STMFOPT(*RPLXLSSHT)</b>

Lists one or more worksheets in an existing Excel file which will be replaced by new data written to the file.

The options are:

**\*NONE**

No existing worksheets are replaced.

worksheet\_name

Specify between 1 and 10 worksheets to be replaced.

## ***RSCDIR – Resource directory***

Parameter	<b>RSCDIR</b>
Applies to commands:	<b>None – deleted in this release</b>
Dependent on:	<b>None</b>

This parameter has been deleted in this release.

This parameter allowed you to specify an IFS path where CoolSpools would look for resources needed during the conversion of a spooled file.

Currently these resources are restricted to PCL soft fonts and macros saved with the RTVPCLRSC command.

This information can now be supplied to CoolSpools by creating an environment variable called CS\_RSC\_DIR.

Options are:

<b><u>*TODIR</u></b>	The directory in which the output is being created. This is not necessarily a value specified on the TODIR directory: it may be derived from the path specified on the TOSTMF parameter.
<b>*CURDIR</b>	The current directory of the job running the command.
<b>Path_name</b>	Specify the full IFS directory path name of the directory in which CoolSpools should look for resources.

## RTF – RTF options

Parameter	<b>RTF</b>
Applies to commands:	<b>CVTSPLRTF</b>
Dependent on:	<b>None</b>

This parameter allows you to control margins and several other factors governing the appearance of text in an RTF document.

System i spooled files are often developed such that the text in the spooled file appears very close to the edges of the page. When the spooled file is converted to an RTF document, and the RTF document is opened in a word processor application such as Microsoft Word, this can give problems because the word processor will typically apply a margin related to the no-print border of the default printer. As a result, the page may not fit correctly and word wrap may occur.

Adjusting the margins that are defined in the document by means of this parameter may help overcome this issue.

CVTSPLRTF defaults are 0, 0, 0, 0, \*MM and \*CALC..

### Left

Specify a value between 0 and 999.99 for the left page margin. The value is measured in the unit of measured defined on the fourth element of this parameter.

### Right

Specify a value between 0 and 999.99 for the right page margin. The value is measured in the unit of measured defined on the fourth element of this parameter.

### Top

Specify a value between 0 and 999.99 for the top page margin. The value is measured in the unit of measured defined on the fourth element of this parameter.

### Bottom

Specify a value between 0 and 999.99 for the right page margin. The value is measured in the unit of measured defined on the fourth element of this parameter.

### Unit of measure

Options for the unit of measure are:

<b>*MM</b>	Millimeters
<b>*CM</b>	Centimeters
<b>*INCH</b>	Inches

### Paragraph spacing

The sixth element is the spacing to be used between paragraphs in the RTF document, measured in points. A point is 1/72 of an inch.

The paragraph spacing determines the vertical positioning of text on the page.

Options are:

<b>*CALC</b>	<p>Spacing between paragraphs is calculated so that data fills the available vertical space on the page. The calculation is based on the vertical coordinate or line number of the data and the overflow line number</p> <p>This is the default for CVTSPLRTF.</p>
<b>*SPLF</b>	<p>An alternative calculation provided for reasons of backwards compatibility only.</p>
<b>Spacing_value</b>	<p>The spacing to apply, in points. Since a fixed spacing value is used, text may move up or down the page compared to the original spooled file, depending on whether the font size has been increased or decreased.</p>

## RTVPRMSET – Retrieve Parameter Set

Parameter	RTVPRMSET
Applies to commands:	CVTSPLPDF, CVTSPLCSV, CVTSPLDBF, CVTSPLDLM, CVTSPLHTML, CVTSPLRTF, CVTSPLSAV, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML, SAVSPLF
Dependent on:	None

Specifies the [parameter set](#) from which parameters will be retrieved.

You must have use authority to the parameter set in order to use it.

The default authority to change or delete the parameter set can be modified by a user with \*ALLOBJ authority or who already has change authority to the parameter set in question by running the CHGPRMSET command.

Individual user authorities to the report can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a parameter set is

### ARIADNE\_PRM\_SET\_nnnnnnnnnn\_CHG

where nnnnnnnnn is the internal parameter set ID, which is displayed by DSPPRMSET.

Options are:

#### **\*SPLF**

The system will search for a parameter set where the parameter set attributes match the spooled file specified on the FROMFILE parameter (or selected from a list subsequently if FROMFILE(\*SELECT) is specified).

Parameter sets are considered in the order of their evaluation priority attribute (lowest-numbered priorities first). The system will first look for a matching parameter set where the command user attribute matches the user profile of the user running the command. If none is found, the system will then look for a matching system default parameter set (one where the command user attribute is \*SYSDFT).

#### **\*NONE**

No parameter set will be used. The parameters for the command must be explicitly specified or the default values will be used.

#### **parameter\_set**

Specify the name of the parameter set to use.

If \*SPLF is specified, and a matching parameter set is located, or if a valid parameter set name is specified, the command parameter values will be retrieved from that parameter set. If a parameter value has been specified on the command line, that parameter value takes precedence over any parameter value retrieved from the parameter set. If no parameter value has been specified on the command line, any value stored with the parameter set overrides the default value.



## ***SIGNATURE - Digital signing options***

Parameter	<b>SIGNATURE</b>
Applies to commands:	<b>CVTSPLPDF</b>

The SIGNATURE (Digital signing options) parameter allows you to apply a digital signature to your PDF files as they are created.

To apply a digital signature to an existing PDF, use the ADDPDFSGN (Add PDF Signature) command.

Digital signatures provide a means of authenticating PDF documents by proving that they originated from the person claiming to have created them and that they have not been subsequently modified.

You will need a file containing a PKCS12 digital certificate in order to digitally sign a PDF. See the section [Digital Signatures](#) below for details of how to obtain and export a certificate for this purpose.

There are two single values:

**\*NO**

The PDF file will not be digitally signed.

**\*EXITPGM**

Details will be supplied by an exit program.

Other options are:

### **Add digital signature?**

Options are:

**\*YES**

Indicates that you wish to digitally sign the file as it is created.

### **Certificate file path**

Specifies the path to the stream file containing a PKCS12 digital certificate key.

### **Certificate password**

Specifies the password associated with the certificate key file.

### **Encrypted password supplied**

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will

unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

- |                   |   |
|-------------------|---|
| <b><u>*NO</u></b> | The password supplied on the previous element is in plain text format and not scrambled.  |
| <b>*YES</b>       | The password supplied on the previous is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used. |

### Reason for signing

Allows you to describe the reason why the document is being signed.

Options are:

- |                     |   |
|---------------------|---|
| <b><u>*NONE</u></b> | No reason is specified.                             |
| <b>reason_text</b>  | Free format text describing the reason for signing. |

### Location

Allows you to describe the location where the document is being signed.

Options are:

- |                      |  |
|----------------------|--|
| <b><u>*NONE</u></b>  | No location is specified.                            |
| <b>location_text</b> | Free format text describing the location of signing. |

### Signing contact information

Allows you to specify a contact for enquiries relating to the signature.

Options are:

- |                     |  |
|---------------------|--|
| <b><u>*NONE</u></b> | No contact is specified.               |
| <b>contact_text</b> | Free format text specifying a contact. |

### Visible signature?

Whether the signature will have some visible representation in the file.

Options are:

- |                        |                                  |
|------------------------|----------------------------------|
| <b><u>*VISIBLE</u></b> | The signature will be visible.   |
| <b>*INVISIBLE</b>      | The signature will be invisible. |

### Show on page number

Which page of the PDF the signature should appear on. Ignored if the signature is not visible.

Options are:

- |                      |  |
|----------------------|--|
| <b><u>*FIRST</u></b> | The signature will appear on the first page.               |
| <b>*LAST</b>         | The signature will appear on the last page.                |
| <b>page_number</b>   | Specify the page number where the signature should appear. |

### Display image file

Specifies the path to an image file (e.g. a JPEG) which will be used to provide a pictorial representation of the signature.

Options are:

<b><u>*NONE</u></b>	There is no pictorial presentation of the signature.
<b>image_path</b>	Specify the path to the image file.

### X coordinate

Specifies the horizontal coordinate of the graphical representation of the signature.

Options are:

<b><u>*LEFT</u></b>	At the left margin of the page
<b>*RIGHT</b>	At the right margin of the page
<b>*CENTER</b>	In the center of the page
<b>X_coordinate</b>	Specify the X coordinate in the units defined below.

### Y coordinate

Specifies the vertical coordinate of the graphical representation of the signature.

Options are:

<b><u>*TOP</u></b>	At the top margin of the page
<b>*BOTTOM</b>	At the bottom margin of the page
<b>*CENTER</b>	In the center of the page
<b>Y_coordinate</b>	Specify the Y coordinate in the units defined below.

### Width

Specifies the horizontal dimension of the graphical representation of the signature.

Options are:

<b><u>*DFT</u></b>	The actual width of the image as defined in the image properties.
<b>width</b>	Specify the width of the image in the units defined below.

### Height

Specifies the vertical dimension of the graphical representation of the signature.

Options are:

<b><u>*DFT</u></b>	The actual height of the image as defined in the image properties.
<b>height</b>	Specify the height of the image in the units defined below.

### Unit of measure

Defines the units used to specify the dimensions and coordinates.

Options are:

<b><u>*MM</u></b>	Millimeters
<b>*CM</b>	Centimeters
<b>*INCH</b>	Inches

## ***SPLFCCSID – Spooled File CCSID***

Parameter	<b>SPLFCCSID</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSAV, CVTSPLTIFF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>
Dependent on:	<b>Only shown if F10 pressed</b>

This parameter allows you to indicate the CCSID (Coded Character Set Identifier) which CoolSpools should assume when converting the data content of the spooled file, in the absence of any other indication of the appropriate CCSID to use. The CCSID specifies the encoding scheme used to represent the data and determines how particular code point values will be interpreted and converted in the stream file that is to be created.

Although more advanced printer data streams such as AFP and IPDS will include information which indicates the encoding scheme used to represent data in the spooled file, SCS spooled files often contain no explicit information to allow CoolSpools to determine the CCSID of the data.

You may use one of the special values:

<b><u>*RPTDFN</u></b>	(Default for CVTSPLXL). The CCSID specified when you created the report definition that is associated with the report map you are using.
<b><u>*SPLF</u></b>	(Default other commands) CoolSpools will use whatever information is available from the spooled file to determine the correct CCSID to use.
<b>*SYSVAL</b>	The value of the QCCSID system value is used.
<b>*JOB</b>	The CCSID of the current job is used. If the CCSID of the job is 65535, the default CCSID attribute of the job is used.
<b>*USER</b>	The CCSID specified in the user profile of the user running the command is used.
<b>CCSID_value</b>	Specify the CCSID to be used.

### ***Example:***

***CVTSPLPDF      FROMFILE(GREEK)...  
                     SPLFCCSID(875)***

Assume we are converting a spooled file received from a Greek customer on an English-language system and that the spooled file is an \*SCS spooled file with CHRID(\*DEV) specified. CoolSpools has no way of knowing that the spooled file contains Greek data, and it would be inappropriate to use the local CCSID. CoolSpools must be told to use an appropriate Greek EBCDIC CCSID (875) to convert the data.

Note that this would not be necessary if the Greek customer were running the conversion him or herself.

## ***SPLIT - Split spooled file***

Parameter	<b>SPLIT</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSPLF, CVTSPLSTMF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>

The **SPLIT** (Split Spooled File) parameter allows you to request that CoolSpools create several stream files from a single spooled file, splitting the spooled file based on criteria that you specify on the **SPLITPAGE**, **SPLITPOS** and/or **SPLITKEY** parameters.

This option can be useful if your program produces a single spooled file which has traditionally been split up into separate pages or groups of pages prior to distribution on paper. You can ask CoolSpools to split the spooled file every so many pages, or when a specified key string appears in the spooled file.

CoolSpools generates names for the stream files it creates by appending a sequential number to the part of the stream file name specified on the **TOSTMF** parameter preceding any file extension. For example, if you specify:

TOSTMF(spool.pdf)

And an option other than **SPLIT(\*NONE)**

CoolSpools will create stream files called **spool1.pdf**, **spool2.pdf**, **spool3.pdf** etc.

If you wish to give each stream file a name more appropriate to its contents (e.g. naming it after the customer to whom it relates), this can be achieved by means of an exit program. The customer number or name should be extracted from the spooled file and passed to the exit program as a parameter. You can then either:

a) Call the exit program before the stream file is created (i.e. at the \*STMFSTR exit point) and override the stream file name by generating an option structure of type CS\_STM01.

or

b) Call the exit program after the stream file has been created (i.e. at the \*STMFEND exit point) and rename the stream file by calling a command such as REN”.

See the CoolSpools Programmer’s Guide for further details.

### **Split based on**

The first element indicates the method you wish to use for identifying split points in your spooled files.

Options are:

#### **\*NONE**

Do not split the spooled file. A single stream file is created.

#### **\*PAGE**

Split the spooled file into separate stream files every so many pages. This option is useful if, for example, you want to create separate stream files for each customer in the report, and the section of

the report relating to a single customer is always a fixed number of pages long.

**\*KEY**

Split the spooled file into separate stream files every time a given key string appears in the spooled file. This option can be useful if, for example, you wish to split the report every time a piece of text (e.g. a field label such as 'Customer number') appears in the report.

Alternatively, this method can also be used to identify split points by checking the value of the text at a particular area of the spooled file, where that text is located by means of its offset position from a specified key string.

For example, if the customer number in your spooled file is preceded by the string 'Customer number', you can use 'Customer number' as the key string to locate the customer number on the page, then use changes to the customer number as the trigger for the creation of a new PDF file.

You will define the precise splitting criteria on the SPLITKEY parameter.

**\*POS**

Split the spooled file into separate stream files based on checking the value of the text in the spooled file at a specified position on the page. The position is identified by means of coordinates down the page from top to bottom or line numbers and coordinates across the page from left to right or columns numbers.

You will define the precise splitting criteria on the SPLITPOS parameter.

**\*POSKEY**

A combination of positional and key splitting criteria will be used.

You will define the precise splitting criteria on both the SPLITPOS and the SPLITKEY parameter.

This option is not supported by CVTSPLSTMF.

**\*PAGGRP**

The spooled file will be split every time a new page group is started. Page groups can be defined in the spooled file by means of the DDS PAGGRP keyword.

## Split method

The second element of the SPLIT parameter indicates whether splitting should occur before the split point or after it.

Options are:

- \*BEFORE** (Default). The spooled file is split before the split point. The page on which the split point occurs will become the first page of the new PDF file.
- This option is typically used where the text which triggers the creation of a new stream file occurs in a heading at the start of the new section of the spooled file, i.e. the trigger identifies the start of a new section.
- \*AFTER** The spooled file is split after the split point. The page on which the split point occurs will be the last page prior to the start of a new PDF file.
- This option is typically used where the text which triggers the creation of a new stream file occurs in a footing at the end of the previous section of the spooled file, i.e. the trigger identifies the end of the section.

Note that each stream file must consist of at least one complete page. CoolSpools cannot split a single page across stream files.

### Suffix separator character

The third element of the SPLIT parameter determines the separator character, if any, that is inserted between the body of the file name you specify on the **TOSTMF** parameter (i.e. the name prior to the extension) and the numeric suffix which CoolSpools appends to that name to create a file name for each stream file generated.

Options are:

- \*NONE** No separator is used.
- \*UNDERSCORE** An underscore character (\_).
- separator\_char** Any other character allowed in a file name.

### Create new workbook or sheet

This element is available only from CVTSPLXL and CVTSPLXLS. It allows you to specify, when a spooled file is being converted to Excel format and splitting is requested, whether each split point generates a new workbook (Excel file) or a new worksheet.

Options are:

- \*WORKBOOK** Each split creates a new workbook (Excel file).
- \*WORKSHEET** Each split creates a new worksheet within the same workbook.

This option establishes the default action. You can specify the action to be taken for each split criterion on the SPLITPOS and SPLITKEY parameters of CVTSPLXL and CVTSPLXLS.



**Example:**

```
CVTSPLPDF      FROMFILE(INVOICES)  
                TOSTMF(invoice.pdf)...  
                SPLIT(*PAGE *BEFORE *NONE)  
                SPLITPAGE(1)
```

Here CoolSpools will create stream files called **invoice1.pdf**, **invoice2.pdf**, **invoice3.pdf** etc. since you have specified **\*NONE** for the separator character.

However, if you specify:

```
CVTSPLPDF      FROMFILE(INVOICES)  
                TOSTMF(invoice.pdf)...  
                SPLIT(*PAGE *BEFORE *UNDERSCORE)  
                SPLITPAGE(1)
```

CoolSpools will create stream files called **invoice\_1.pdf**, **invoice\_2.pdf**, **invoice\_3.pdf** etc. as you have requested that an underscore be used for the separator character.

## ***SPLITKEY – Split by key options***

Parameter	<b>SPLITKEY</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSPLF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>
Dependent on:	<b>Others: SPLIT(*KEY) or SPLIT(*POSKEY)</b>

The **SPLITKEY** (Split Key String) parameter can be used only if **SPLIT(\*KEY)** is selected, or, in relation to the format-specific commands only, **SPLIT(\*POSKEY)**.

CVTSPLSTMF allows only a single key string to be defined for splitting. The format-specific commands allow up to 100 key strings to be defined.

This parameter has two related functions.

The first function allows you to specify a key string which will trigger the creation of a new stream file every time it appears in the spooled file.

### ***Example:***

```
CVTSPLPDF      FROMFILE(SALES)...  
                SPLIT(*KEY  
                SPLITKEY('Branch code:')
```

Here the sales report is split into separate PDF files every time the string 'Branch code:' appears in the report.

The second function allows you to locate an area of the spooled file on the page by means of an offset position from the given key string. You can then perform comparisons on the value of the text at the position in the spooled file thus located and use these to control the splitting of the spooled file.

### ***Example:***

```
CVTSPLPDF      FROMFILE(SALES)...  
                SPLIT(*KEY)  
                SPLITKEY( 'Branch code:'  
                        *IF 12 10 *ROWCOL *NE *PRV)
```

Here the sales report is split into separate PDF files based on the 10 characters of text that appear 12 characters to the right of the string 'Branch code:' If this text is not equal to the previous value at this same position, a split will occur.

### ***Split key string***

The first element is the key string itself.

This is a case-sensitive value.

### ***Split by key method***

The second element is the method of operation of this parameter

Options are:

<b>*ALWAYS</b>	Split the spooled file every time the key string appears in the spooled file. This is the default value.
<b>*IF</b>	Use the key string to locate an area of the spooled file and then conditionally split the spooled based on performing a comparison on the text at that position.

The remaining elements are relevant only if **\*IF** is specified for the second part of the **SPLITKEY** parameter.

### Offset

Enter the offset in characters from the start of the key string to the start of the text to be checked for splitting purposes.

If a positive number is entered, this is interpreted as indicating that the value to check is to the right of the key string, whereas a negative number indicates that the value to check is to the left of the key string.

### Length

Enter either length of the value to check in characters.

### Measurement method

The only option is now:

<b><u>*ROWCOL</u></b>	Rows and columns. Use DSPSPLF as your guide.
-----------------------	---

### Comparison

A comparison operator. This allows you to indicate the type of comparison to be performed on the area of the spooled file identified by the preceding parameters.

Options are:

<b>*EQ</b>	Equal to.
<b>*NE</b>	Not equal to
<b>*GT</b>	Greater than
<b>*LT</b>	Less than
<b>*GE</b>	Greater than or equal to
<b>*LE</b>	Less than or equal to
<b>*CT</b>	“Contained in”, i.e. the comparison string appears somewhere in the area identified
<b>*NC</b>	“Not contained in”, i.e. the comparison string does not appear anywhere in the area identified

### String to compare

The last element is the string against which the area of the spooled file selected should be compared, using the comparison operator specified in the preceding parameter. Any string may be specified, but the default value is the special value **\*PRV**, which denotes the previous value at the same location.

### **Create new workbook or sheet**

CVTSPLXL and CVTSPLXLS only.

Whether a split triggered by this rule creates a new workbook (Excel file) or a new worksheet within the Excel file.

Options are:

<b><u>*DFT</u></b>	The default action specified on the SPLIT parameter is taken.
<b>*WORKBOOK</b>	A new workbook (Excel file is created).
<b>*WORKSHEET</b>	A new worksheet is created within the workbook.

Note that it is possible to define multiple split criteria, some specifying \*WORKBOOK and some \*WORKSHEET. This enables multi-level splitting where a top-level split criterion causes multiple workbooks to be created and a secondary split criterion causes separate worksheets within the workbook to be generated.

#### **Example:**

```
CVTSPLPDF      FROMFILE(INVOICES)...  
                SPLIT(*KEY)  
                SPLITKEY('Page No.: ' *IF 10 3 ROWCOL *EQ '1')
```

In this scenario, imagine that the spooled file contains invoices for multiple customers. Every time a new customer invoice is started, the page number is reset to 1. Here, the SPLITKEY parameter is being used to locate the page number in the spooled file: it is the text 3 characters long 10 characters to the right of the words "Page No.". A split will occur every time this area of the spooled file is equal to "1", i.e. every first page of a customer invoice.

#### **Example:**

```
CVTSPLPDF      FROMFILE(SALES)...  
                SPLIT(*KEY)  
                SPLITKEY('Area code: ' *IF 12 5 *ROWCOL *NE *PRV)
```

Here, the SPLITKEY parameter is being used to locate the area code in the spooled file: it is the 5 characters of text 12 characters to the right of the words "Area code:". A split will occur every time this area of the spooled file is different from the previous value at the same position, i.e. every time the area code changes.

## ***SPLITPOS - Split by position options***

Parameter	<b>SPLITPOS</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSPLF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>
Dependent on:	<b>Others: SPLIT(*POS) or SPLIT(*POSKEY)</b>

This parameter allows you to locate an area of the spooled file on the page by means of coordinates down the page from top to bottom and across the page from left to right, or by line number and column position. You can then perform comparisons on the value of the text at the position in the spooled file thus located and use these to control the splitting of the spooled file.

### ***Example:***

```
CVTSPLPDF      FROMFILE(SALES)...  
                SPLIT(*POS)  
                SPLITPOS(1 3 10 *ROWCOL *NE *PRV)
```

Here the sales report is split into separate stream files based on the 10 characters of text that appear at column 3 of line 1 of each page. If this text is not equal to the previous value at this same position, a split will occur.

### **Line number**

The line number of the start of the area of the page to be checked, i.e. the position down the page from top to bottom where the area of the spooled file to be checked is located.

### **Character position**

The character position or column number of the start of the area of the page to be checked, i.e. the position across the spooled file from left to right where the area of the spooled file to be checked is located.

### **Length**

The length of the area to be checked. It is specified in either columns, inches or millimeters, depending on the value of the measurement method option (see fourth element below).

### **Measurement method**

The only option is now:

**\*ROWCOL**

Rows and columns.

Use DSPSPLF as your guide.

### **Comparison**

A comparison operator. This allows you to indicate the type of comparison to be performed on the area of the spooled file identified by the preceding parameters.

Options are:

**\*EQ**

Equal to.

<b>*NE</b>	Not equal to
<b>*GT</b>	Greater than
<b>*LT</b>	Less than
<b>*GE</b>	Greater than or equal to
<b>*LE</b>	Less than or equal to
<b>*CT</b>	“Contained in”, i.e. the comparison string appears somewhere in the area identified
<b>*NC</b>	“Not contained in”, i.e. the comparison string does not appear anywhere in the area identified

### String to compare

The last element is the string against which the area of the spooled file selected should be compared, using the comparison operator specified in the preceding parameter. Any string may be specified, but the default value is the special value \*PRV, which denotes the previous value at the same location.

### Create new workbook or sheet

CVTSPLXL and CVTSPLXLS only.

Whether a split triggered by this rule creates a new workbook (Excel file) or a new worksheet within the Excel file.

Options are:

<b><u>DFT</u></b>	The default action specified on the SPLIT parameter is taken.
<b>*WORKBOOK</b>	A new workbook (Excel file) is created).
<b>*WORKSHEET</b>	A new worksheet is created within the workbook.

Note that it is possible to define multiple split criteria, some specifying \*WORKBOOK and some \*WORKSHEET. This enables multi-level splitting where a top-level split criterion causes multiple workbooks to be created and a secondary split criterion causes separate worksheets within the workbook to be generated.

### Example:

**CVTSPLPDF      FROMFILE(SALES)...  
                     SPLIT(\*POS)  
                     SPLITPOS(2 12 5 \*ROWCOL \*NE \*PRV)**

Here, the SPLITPOS parameter is being used to locate the area code in the spooled file: it is the 5 characters of text at column 12 of line 2. A split will occur every time this area of the spooled file is different from the previous value at the same position, i.e. every time the area code changes.

## ***SPLITPAGE – Split file every n pages***

Parameter	<b>SPLITPAGE</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSPLF, CVTSPLTXT, CVTSPLXL, CVTSPLXLS, CVTSPLXML</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and SPLIT(*PAGE) Others: SPLIT(*PAGE)</b>

When SPLIT(\*PAGE) is specified to indicate that the spooled file should be split into separate output files every so many pages, this parameter allows you to specify the number of pages after which CoolSpools will create a new output file.

### ***Example:***

```
CVTSPLPDF      FROMFILE(INVOICES)...  
                SPLIT(*PAGE)  
                SPLITPAGE(2)
```

Here the invoices spooled file is split into separate stream files every two pages. Each stream file will contain exactly two pages.

```
CVTSPLSPLF     FROMFILE(INVOICES)...  
                SPLIT(*PAGE)  
                SPLITPAGE(10)
```

Here again the invoices spooled file is being split, but this time into different spooled files, each 10 pages long.

## STMFCODPAG – Stream File Code Page

Parameter	<b>STMFCODPAG</b>
Applies to commands:	<b>CVTSPLSTMF, CVTSPLCSV, CVTSPLHTML, CVTSPLRTF, CVTSPLSAV, CVTSPLTIFF, CVTSPLTXT, CVTSPLXML, RTVPCLRSC, RTVSPLDTA, SAVSPLF</b>
Dependent on:	<b>Only shown if F10 pressed</b>

This parameter determines the CCSID attribute that CoolSpools assigns to stream files that it creates. In the case of text-based output formats (e.g. \*TEXT, \*HTML and \*CSV) it also determines the code page used to encode data in the file.

Some of the output formats that CoolSpools supports are **binary** formats. For example, PDF and Excel formats both have their own specific rules that govern how data in those files formats can be represented. Similarly, the output from the CVTSPLSAV, SAVSPLF commands is compressed binary data and does not represent characters. Likewise, the output from RTVSPLDTA is not translated and is retained in its original encoding.

On the other hand, other file formats that CoolSpools can generate are **text** formats and data in the spooled file will typically be converted to an ASCII or Unicode representation when those file types are being created. Examples are the output from the CVTSPLTXT, CVTSPLCSV and CVTSPLHTML commands.

In relation to text formats, the value specified on the STMFCODPAG command will determine the way in which data from the spooled file is translated and re-encoded before being written to the output file. For example, when you are creating a text file from your spooled file so that you can open it on non-system i platform, the value you specify on the STMFCODPAG should correspond to the data format appropriate to that platform, e.g. Windows ASCII, Unicode etc.

In relation to binary formats, the encoding of the data is determined by the requirements of the output format itself.

However, every stream file that CoolSpools creates will be assigned a CCSID attribute that can be viewed when the file attributes are displayed with DSPLNK or WRKLNK. OS/400 uses this CCSID attribute to decide how to handle data in the file when the file is copied or moved.

In relation to text file formats, this CCSID attribute should match the actual encoding of data in the file so that if the file is translated, for example by being sent to another system by FTP when not in binary mode, the translation is performed correctly.

In relation to binary file formats, this CCSID attribute is largely arbitrary because the data in the file is binary not text. If OS/400 attempts any translation of data in the file when it is copied or moved, the file will be corrupted. You should therefore use the STMFCODPAG parameter to assign a CCSID attribute which will minimize the risk of this happening. For example, if you are most likely to access the file from Windows, assign a Windows ASCII CCSID so that the data will not be translated when copied to Windows.

### **Stream file encoding**

Options are:



<b><u>*CALC</u></b>	(Default) CoolSpools selects an appropriate codepage based on the CCSID of the spooled file and the format to which the data is being converted.
<b>*WINDOWS</b>	CoolSpools selects the Windows ASCII codepage corresponding to the CCSID of the spooled file data, e.g. 1252.
<b>*PCASCII</b>	A synonym for *WINDOWS provided for the sake of compatibility with previous releases and consistency with IBM-supplied commands such as CPYTOSTMF.
<b>*IBMASII</b>	CoolSpools selects the IBM PC ASCII codepage corresponding to the CCSID of the spooled file data, e.g. 437.
<b>*STDASCII</b>	A synonym for *IBMASII provided for the sake of compatibility with previous releases and consistency with IBM-supplied commands such as CPYTOSTMF.
<b>*ISOASCII</b>	CoolSpools selects the ISO ASCII codepage corresponding to the CCSID of the spooled file data, e.g. 819.
<b>*STMF</b>	If the stream file exists, the code page of the existing stream file is used, where it is appropriate to the file format being created.
<b>*UNICODE</b>	CoolSpools converts data to Unicode (specifically, UCS-2 bigendian) format, CCSID 13488)
<b>*UCS2</b>	Equivalent to *UNICODE.
<b>*UTF8</b>	CoolSpools converts data to Unicode UTF-8 encoding. (CCSID 1208).
<b>*UTF16</b>	CoolSpools converts data to Unicode UTF-16 encoding (CCSID 1200)
<b>*NOCONV</b>	Data is not converted. It is left in its original encoding. This may give the best results with certain spooled file data, such as Arabic CCSID 420.
<b>CCSID_value</b>	Enter a specific CCSID to be used.

### ***Bigendian or littleendian***

When converting to a format that supports Unicode encoding, and a double-byte Unicode value is selected on the previous element, whether values should be encoded using bigendian or littleendian byte order.

Options are:

<b><u>*LITTLE</u></b>	Littleendian. The byte order used by PCs. The most significant of the two bytes is encoded second.
-----------------------	--

**\*BIG**

Bigendian. The byte order used by the system i.  
The most significant of the two bytes is encoded first.

***Include Unicode marker?***

This option determines whether CoolSpools outputs a marker at the start of a text file which indicates to a reader application whether the byte order is bigendian or littleendian. Some applications such as Windows NotePad check for a marker at the start of the file (hex x'FEFF' or x'FFFE') and use this to identify whether Unicode data is encoded in bigendian or littleendian format.

Options are:

**\*YES**

A byte order marker is output

**\*NO**

No byte order marker is output

***Example:***

***CVTSPLTXT***

***FROMFILE(INVOICES)...***  
***STMFCODPAG(1253)***

Here the CVTSPLTXT command is being applied to create an ASCII text file from a Greek-language spooled file called **INVOICES**. Code page 1253, suitable for Greek-language data, will be used to convert the contents of the spooled file.

## TEXT – Text options

Parameter	<b>TEXT</b>
Applies to commands:	<b>CVTSPLCSV, CVTSPLTXT, CVTSPLHTML, CVTSPLPDF, CVTSPLRTF, CVTSPLSTMF</b>
Dependent on:	<b>CVTSPLSTMF: PMTADLPARM(*YES) and TOFMT(*PDF), TOFMT(*HTML), TOFMT(*HTMLCSS) or TOFMT(*RTF).</b>

The **TEXT** (Text options) parameter allows you to control various options relating to the processing of text in the spooled file.

The format varies slightly from one command to another as explained below.

### *Include overlay text?*

The first element determines whether textual content derived from an overlay or page segment object is included in the output.

Overlays often contain constants, labels and heading text while the spooled file itself contains the variable data associated with those constants, labels and headings. For example, you might have an invoicing application where your overlay contains text such as “Customer name”, “Invoice number” and “Invoice date” and your spooled file supplies the actual customer name, invoice number and invoice date information to be printed alongside those labels.

In some circumstances, for example when creating PDF versions of your spooled file, it may be appropriate to include the overlay text for the sake of clarity. However, in other circumstances, for example when converting the data to CSV format for interfacing into a Data Warehouse application, it might be more appropriate to exclude the overlay text and just process the variable data from the spooled file itself. This parameter allows you to indicate which option you wish to choose.

Values are:

#### **\*OUTPUT**

Overlay text is included in the stream file that is created. However, text from overlays and page segments is ignored when processing text functions such as bookmarks, split triggers and exit program parameters.

This option is the default value for CVTSPLHTML, CVTSPLPDF, CVTSPLRTF and CVTSPLTXT

#### **\*IGNOVLDTA**

All content from overlay and page segment objects (both text and non-text) is ignored and dropped from the processing.

#### **\*TOFMT**

CoolSpools determines whether to include overlay text based on the format of the stream file being created. Overlay text is included if the spooled file is being converted to \*PDF or \*HTMLCSS, otherwise it is excluded.

This option is available only from CVTSPLSTMF, where it is the default value.

<b>*SPLF</b>	CoolSpools determines whether to include overlay text based on the format of the spooled file being converted. Overlay text is excluded if the spooled file is being converted is *SCS, otherwise it is included.
<b>*YES</b>	Overlay text is included.
<b>*NO</b>	Overlay text is excluded.

### ***Include blank lines?***

This second element is available only from the CVTSPLTXT command as it is relevant only to flat ASCII text output.

This parameter allows you to define whether blank lines in the original report should be duplicated in the output.

<b><u>*YES</u></b>	(Default) Blank lines in the original report are reflected in the output. Pages are padded out with blank lines to resemble the printed page.
<b>*NO</b>	Blank lines in the original report are not reflected in the output and are compressed out.
<b>*FF</b>	Blank lines in the original report are reflected in the output. At the end of each page, a form feed character (x'0C') is embedded in the output to force a page throw.

### ***Text line calculation method***

This option controls the way in which CoolSpools calculates line numbers in the report for the purposes of creating text files and for text selection (e.g. bookmarks, split keys, exit program parameters). Where the spooled file contains text with different font sizes, especially proportional fonts, it is not obvious how to calculate the text "line" for a piece of text when text is being selected using the \*ROWCOL method.

Values are:

<b>*ENVVAR</b>	CoolSpools uses the method indicated by environment variable CS_TXT_LINE_METHOD. If this exists, and is set to *NEW, the new method is used (see *NEW below), otherwise the old method is used (see *OLD below). A job-level environment variable overrides a system-level environment variable.  This is the default for all commands except CVTSPLSTMF.
<b>*NEW</b>	CoolSpools calculates text line numbers using the LPI attribute of the spooled file. This is the method used by DSPSPLF.
<b>*OLD</b>	CoolSpools uses the LPI values but the results are slightly different from those given by *NEW and may differ from those shown by DSPSPLF.

This option is provided purely for reasons of backwards compatibility and \*NEW is the recommended value. However, where you have existing applications which rely on CoolSpools returning data from your report based on the previous method, this option can be selected to avoid having to modify the application.

This option was introduced by Version 5 PTF 5CV0028. Earlier versions of CoolSpools and Version 5 without that or a later PTF may calculate line numbers in AFP and other non-SCS spooled files differently from DSPSPLF.

While we believe the new method of calculating line numbers is a significant improvement and will help users to determine the correct parameters to use, it inevitably means that CoolSpools' behavior could change and this has a potential effect on existing applications which depend on text being selected using the \*ROWCOL method. In particular, the parameters passed to exit programs could change, or splitting might no longer work as expected, or incorrect bookmarks could be generated.

These changes affect:

- Exit Program Parameters (EXITPGMPOS parameter)
- Bookmarks (BMARKPOS parameter)
- Splitting (SPLITPOS parameter)
- Lines of text in files output in \*TEXT, \*CSV, \*HTML or \*XLS formats.

In order to minimize the risk of disrupting existing systems:

- For CVTSPLSTMF, the default for this option is \*OLD, which means that line numbers will be calculated as they were previously.
- For the other commands, the default is \*ENVVAR, which means that the method used depends on the setting of environment variable CS\_TXT\_LINE\_METHOD.
- If the value of the environment variable is \*NEW, the new method is used
- If the environment variable does not exist or its value is anything other than \*NEW, the old method is used.
- When the product is installed or reinstalled, and no system environment variable called CS\_TXT\_LINE\_METHOD exists, one is created with a value of \*NEW. This is intended to allow new users to gain immediate benefit from the improved method of calculating line numbers.
- When PTF 5CV0028 or later is applied, and no system-level environment variable called CS\_TXT\_LINE\_METHOD exists, one is created with a value of \*OLD. This is intended to protect existing users who may have applications which depend on CoolSpools calculating line numbers according to the previous method from unexpected changes.
- However, existing users should note that if you install CoolSpools on a new system, you will need to set the value of the environment variable

to match that on your previous system otherwise CoolSpools could behave differently on the new system.

- If the environment variable exists as both a system-level and a job-level environment variable, the job-level environment variable overrides the system-level environment variable.
- You can change the value of the system-level environment variable to enable or disable the new method system-wide, or you can create job-level environment variables to override the behavior for particular jobs. This can be useful if you wish to test the effects of changing the system-level environment variable, e.g.:

#### **ADDENVVAR**

```
ENVVAR('CS_TXT_LINE_METHOD') VALUE('*NEW') LEVEL(*JOB)
```

#### **CHGENVVAR**

```
ENVVAR('CS_TXT_LINE_METHOD') VALUE('*OLD') LEVEL(*SYS)
```

We believe that this approach gives the best possible compromise between protecting existing users, whose applications might be adversely affected by these changes, and making the benefits of these changes available to new users immediately.

### ***CPI value to use for output***

The characters per inch value to assume when calculating columns/character positions.

Options are:

<b><u>*SPLF</u></b>	The value implied by the CPI attribute of the spooled file.
<b>*BESTFIT</b>	CoolSpools calculates a value based on the smallest font size used in the file, intended to avoid text overlaying other text in the output.
<b>CPI_value</b>	Specify an alternative CPI setting. This can be useful when converting spooled files where the CPI value is misleading (e.g. *USERASCII spooled files) or which contain text which uses variable CPI values or variable size fonts. If using the spooled file CPI value results in text overlaying other text of being truncated, because the CPI value or font size associated with that text is different from the spooled file CPI, specify a higher CPI value here.

### ***LPI value to use for output***

The lines per inch value to assume when calculating line numbers.

Options are:

<b><u>*SPLF</u></b>	The value implied by the LPI attribute of the spooled file.
---------------------	---

<b>*BESTFIT</b>	CoolSpools calculates a value based on the smallest font size used in the file, intended to avoid text overlaying other text in the output.
<b>LPI_value</b>	Specify an alternative LPI setting. This can be useful when converting spooled files where the LPI value is misleading (e.g. *USERASCII spooled files) or which contain text which uses variable LPI values or variable size fonts. If using the spooled file LPI value results in text overlaying other text of being truncated, because the LPI value or font size associated with that text is different from the spooled file LPI, specify a higher LPI value here.

## ***TITLE - Title for HTML or PDF***

Parameter	TITLE
Applies to commands:	CVTSPLPDF, CVTSPLHTML, CVTSPLSTMF
Dependent on:	CVTSPLSTMF: TOFMT(*PDF), TOFMT(*HTML), TOFMT(*HTXT), TOFMT(*HTMLCSS) and PMTADLPARM(*YES)

The **TITLE** parameter allows you to define a title for the report in HTML or PDF.

If you are running CVTSPLSTMF, this parameter is displayed during command prompting only if you specified **PMTADLPARM(\*YES)** to prompt additional parameters and if one of the following options is specified for the **TOFMT** parameter: \*PDF, \*HTML, \*HTXT, \*HTMLCSS.

In relation to HTML output, the text you enter for the TITLE parameter will appear in your browser's title bar when the HTML file that CoolSpools creates is opened.

In relation to PDF, the text you enter for the TITLE parameter will appear when you open the file that CoolSpools creates in Adobe Acrobat Reader and display the document properties.

CoolSpools variables may be specified on this parameter element.

Alternatively, you may select one of the special values:

<b><u>*NONE</u></b>	The report has no title.
<b>*STMFILE</b>	The report title is the same as the stream file name specified on the TOSTMF parameter.

### ***Example:***

***CVTSPLPDF      FROMFILE(SALES)...  
                     TIITLE('Sales Statistics April 2010')***

The sales report is converted to PDF format. Users can check the Document Properties in Acrobat Reader to see the title 'Sales Statistics April 2010' to confirm the nature of the report.



## ***TOFILE - To spooled file name***

Parameter	<b>TOFILE</b>
Applies to commands:	<b>CVTSPLSPLF</b>
Dependent on:	<b>None</b>

This parameter specifies the name of the spooled files to be created by the CVTSPLSPLF command. It is roughly equivalent to the TOSTMF parameter.

Options are:

<b><u>*FROMFILE</u></b>	CoolSpools uses the name of the original spooled file and appends a numeric suffix to create a unique name for each spooled file.
<b>*SAME</b>	Each spooled file will have the same name as the original spooled file.
<b>*EXITPGM</b>	The name will be specified at run time by an exit program, which will generate a CS_STM01 option structure.
<b>SpLf_name</b>	Specify the spooled file name to be used. CoolSpools will append a numeric suffix if several spooled files are to be created.

If a suffix separator character is specified on the SPLIT parameter, this will be inserted between the body of the spooled file name and the numeric suffix.

The maximum length of a spooled file name is 10 characters. If the name that results after CoolSpools has interpreted the value specified and added any suffix is longer than 10 characters, an error will occur.

## ***COLUMNOPT – Column creation options***

Parameter	<b>COLUMNOPT</b>
Applies to commands:	<b>CVTSPLXLS</b>
Dependent on:	

This parameter allows you to control the way in which CoolSpools calculates the columns into which the data from the spooled file will be arranged in the Excel file.

### ***Column creation method***

Determines the method that CoolSpools uses to decide how text in the spooled file should be allocated to columns in the spreadsheet CVTSPLXLS creates.

Many spooled files, especially \*SCS spooled files created using program-described printer files and Query output, are just completely “flat” text files, where each line of the spooled file is a single, unformatted block of text. When CVTSPLXLS is converting a spooled file such as this to a spreadsheet, CoolSpools needs to try to split those lines of text up into columns. This process is far from trivial.

By default CVTSPLXLS uses some statistical techniques to identify patterns in the data and will try to allocate text to columns as best it can, but this process can never be 100% reliable. You should also bear in mind that CVTSPLXLS is not guaranteed to produce the same results every time, as the decisions it makes can vary depending on the particular set of data that appears in one version of a spooled file compared to another.

This option controls the method CVTSPLXLS uses to allocate text to columns. If the default \*CALC method does not give good results for you, you should consider one of the following alternatives.

- Try overriding or fine-tuning the results you get using COLUMNOPT(\*CALC) by using COLUMNOPT(\*CALCPOS) instead and using the COLUMNPOS parameter to add in missing columns or remove unwanted columns. This is more work than using COLUMNOPT(\*CALC) but is likely to produce better results.
- Use COLUMNOPT(\*POS) and specify all of the column positions you want on the COLUMNPOS parameter. Again, while this involves some initial work on your part to determine the appropriate column positions, that work will pay off in terms of the results you obtain.
- Better still, use the CVTSPLXL command instead of CVTSPLXLS. CVTSPLXL requires you to create a report definition describing the structure and contents of the spooled file you are converting. It also requires you to create a Report-to-Excel map based on that report definition which tells CoolSpools how you want your Excel file to be structured and where to take the information from. While this can seem like a lot of work, you only need to do it once and, when it's done, you will be able to produce reliable and consistent output tailored to your precise needs.

Options are:

<b>*CALC</b>	CoolSpools will attempt to calculate the column positions automatically.
<b>*POS</b>	You will specify all column positions manually, on the COLUMNPOS parameter.
<b>*CALCPOS</b>	CoolSpools will calculate the column positions automatically, but you will fine-tune the column positions selected using options on the COLUMNPOS parameter.
<b>*TOKEN</b>	<p>CoolSpools will calculate the column positions automatically using a different method which identifies text "tokens" in the spooled file.</p> <p>This method may give better results than COLUMNOPT(*CALC) with spooled files created from externally described printer files.</p>

### ***Column creation threshold***

This option allows you to influence the way in which CoolSpools' algorithm for calculating column positions operates. The algorithm uses statistical techniques to identify character positions in the report where left-aligned alphanumeric or right-aligned numeric columns of data appear. The algorithm will select character positions where such items occur with a frequency which is more than a given number of standard deviations from the norm. The value specified here is the number of standard deviations to use. If the default value (1 standard deviation from the norm) does not give good results, you can try adjusting this to a different value.

Options are:

<b><u>1.00</u></b>	One standard deviation.
<b>0.00-9.99</b>	The number of standard deviations from the mean to use in the column selection algorithm.

### ***Ignore char position (\*TOKEN)***

Specifies whether CoolSpools should taken any notice of character position when implementing the COLUMNOPT(\*TOKEN) method.

Options are:

<b><u>*NO</u></b>	CoolSpools will identify text "tokens" in the spooled file and allocate those tokens to columns based on the character position across the page where the token is found. If blanks are being used as the token delimiter, and the value in a column is blank, this will help ensure that columns align correctly.
<b>*YES</b>	CoolSpools will identify text "tokens" in the spooled file and allocate those tokens to columns based on sequential number of the token (i.e. the first such token is allocated to column A, the second to column B etc.)

## COLUMNPOS –Column positions

Parameter	<b>COLUMNPOS</b>
Applies to commands:	<b>CVTSPLXLS</b>
Dependent on:	

This parameter allows you to specify the column positions to be used in the Excel file (in conjunction with COLUMNOPT(\*POS)) or to override the column positions determined automatically by CoolSpools (in conjunction with COLUMNOPT(\*CALCPOS)).

If you are having difficulty obtaining the results you want using CVTSPLXLS, we recommend you try using CVTSPLXL command instead. CVTSPLXL requires you to create a report definition describing the structure and contents of the spooled file you are converting. It also requires you to create a Report-to-Excel map based on that report definition which tells CoolSpools how you want your Excel file to be structured and where to take the information from. While this can seem like a lot of work, you only need to do it once and, when it's done, you will be able to produce reliable and consistent output tailored to your precise needs.

Note that when CVTSPLXLS is run, CoolSpools will send messages to the joblog regarding the column positions selected. These messages can be helpful in identifying the column positions you need to modify on the COLUMNPOS parameter.

Single option:

### **\*CALC**

No column positions are defined.  
COLUMNOPT(\*CALC) must be specified.

Other values (up to 100 repetitions):

### ***Spooled file column position***

Options are:

1-999

The character position in the report where the column should be added or removed.

Please note that for a **left-aligned column** (\*LEFT specified for the third parameter element below), the position indicated should be the **start** position of the column (left-most character position) but for a **right-aligned column** (\*RIGHT specified for the third parameter element below), the position indicated should be the **end** position of the column (right-most character position, taking into account any possible trailing minus sign).

### ***Action at this position:***

Single option:

### **\*RMV**

The column created by CoolSpools at the specified character position will be removed.

Use this option to remove unwanted columns when COLUMNOPT(\*CALCPOS) has been specified.

Other options:

### ***Column action***

#### **\*ADD**

A column will be created at the specified character position in the report.

Use this option to add extra columns when COLUMNOPT(\*CALCPOS) has been specified, or to define all column positions required, when COLUMNOPT(\*POS) was specified.

### ***Left or right column?***

Options are:

#### **\*LEFT**

The column is left-aligned. The column position specified above indicates the start of the column (left-most position).

#### **\*RIGHT**

The column will be right-aligned. The column position specified above indicates the end of the column (right-most position).

### ***Alpha or numeric column?***

Options are:

#### **\*ALPHA**

The column contains alphanumeric data.

#### **\*NUMERIC**

The column contains numeric data.

### ***Column width***

Options are:

#### **\*CALC**

The column width will be calculated from the position of adjacent columns.

#### **width**

The column width in characters, measuring to the right for a left-aligned column and to the left for a right-aligned column. For the best results, it is recommended that the column width be specified manually using this option.

## LINTYPES –Line types

Parameter	<b>LINTYPES</b>
Applies to commands:	<b>CVTSPLXLS</b>
Dependent on:	

This parameter allows you to tell CoolSpools explicitly what type of line each line in your report is, to influence whether lines are treated as unwanted headings or not, and to override the column structure used for the main body of the spreadsheet in relation to certain lines.

Note that this parameter of the CVTSPLXLS command does not allow anywhere near the same degree of control and precision that the CVTSPLXL command does.

If you are having difficulty obtaining the results you want using CVTSPLXLS, we recommend you try using CVTSPLXL command instead. CVTSPLXL requires you to create a report definition describing the structure and contents of the spooled file you are converting. It also requires you to create a Report-to-Excel map based on that report definition which tells CoolSpools how you want your Excel file to be structured and where to take the information from. While this can seem like a lot of work, you only need to do it once and, when it's done, you will be able to produce reliable and consistent output tailored to your precise needs.

Single values:

**\*NONE**

No line types are defined. CoolSpools will attempt to identify the type of line using statistical and positional criteria.

Other values (up to 100 repetitions):

### ***From page number***

Options are:

**\*FIRST**

The first page.

**\*LAST**

The last page.

**from-page**

Specify the first page number to which the definition relates.

A negative number is interpreted as counting from the end of the report. For example, -1 is the last page, -2 the penultimate page etc.

### ***To page number***

Options are:

**\*LAST**

The last page.

**\*FIRST**

The first page.

**to-page**

Specify the last page number to which the definition relates.

A negative number is interpreted as counting from the end of the report. For example, -1 is the last page, -2 the penultimate page etc.

### ***From line number***

Options are:

<b><u>*FIRST</u></b>	The first line.
<b>*LAST</b>	The last line.
<b>from-line</b>	Specify the first line number to which the definition relates.

### ***To line number***

Options are:

<b><u>*LAST</u></b>	The last line.
<b>*FIRST</b>	The first line.
<b>to-line</b>	Specify the last line number to which the definition relates.

### ***Line type***

What type of line this is.

Options are:

<b><u>*PAGHDG</u></b>	The line is a page heading. The lines selected will be treated as page headings for the purposes of suppressing page headings (see option on EXCEL parameter).
<b>*COLHDG</b>	The line is a column heading. The lines selected will be treated as page headings for the purposes of suppressing column headings (see option on EXCEL parameter).
<b>*OTHER</b>	The line is not a detail line, page heading or column heading (e.g. summary line, total etc.). It will not be subject to either the option for suppressing page headings or that for suppressing column headings (see options on EXCEL parameter).

### ***Conversion method***

How the data for this line should be handled.

Options are:

<b><u>*COLUMNS</u></b>	CoolSpools attempts to assign the data to the column structure in an appropriate way. If this does not give good results for you, use CVTSPLXL instead of CVTSPLXLS.
<b>*LINE</b>	The data is treated as a single line of text and not broken up into columns.

### ***Keep line?***

If and when lines of this type are retained in the output or dropped.

Options are:

#### **\*DFT**

Default action:

- If the line is identified as a page heading (\*PAGHDG specified above), the option from the EXCEL parameter controlling the dropping of page headings applies
- If the line is identified as a column heading (\*COLHDG specified above), the option from the EXCEL parameter controlling the dropping of column headings applies
- If the line is identified as another type (\*OTHER specified above), duplicate lines after the first are dropped.

#### **\*NONE**

All lines identified by this parameter are dropped.

#### **\*ALL**

All lines identified by this parameter are retained.

#### **\*FIRST**

All duplicate lines identified by this parameter after the first are dropped.

### ***Key string to check for***

Specifies a key string to be checked. Only lines which match the page and line criteria above and which also contain this (case-sensitive) key string will be selected by this rule.

Options are:

#### **\*NONE**

No key string is checked. Just the page and line criteria apply.

#### **key**

Specify the key string to be checked for.

### ***Line type name***

Specifies a name for the line type. This name can be used to identify the line type on the APYSTYLES and CNDFMTGRP parameters.

Options are:

#### **\*NONE**

No name is assigned to the line type

#### **line\_type\_name**

Specify a name for the line type. The name can be up to 20 characters in length but must otherwise conform to the normal rules for OS/400 names.



## ***XLSADJUST – Adjust pages to***

Parameter	<b>XLSADJUST</b>
Description	<b>Specifies the percentage scaling when XLSPRINT(*ADJUST...) is requested.</b>
Applies to commands:	<b>CVTSPLXL</b>
Dependent on:	<b>XLSPRINT(*ADJUST)</b>

Specifies the percentage scaling when XLSPRINT(\*ADJUST...) is used.

Options are:

**100**

Scale by 100% (no change).

**0-400**

Specify the percentage scaling.

## ***XLSFIT – Fit pages to***

Parameter	<b>XLSFIT</b>
Description	<b>Specifies the number of pages to fit the output to when XLSPRINT(*FITPAGE...) is requested.</b>
Applies to commands:	<b>CVTSPLXL</b>
Dependent on:	<b>XLSPRINT(*FIT)</b>

Specifies the number of pages to which the output is fitted when XLSPRINT(\*FIT...) is used.

There are two elements:

### **The number of pages wide (horizontal).**

Options are:

<b><u>*AUTO</u></b>	Excel will calculate the number of pages required automatically.
0-65535	Specify the number of pages to which the data should be fitted horizontally.

### **The number of pages tall (vertical).**

Options are:

<b><u>*AUTO</u></b>	Excel will calculate the number of pages required automatically.
0-65535	Specify the number of pages to which the data should be fitted vertically.

## ***XLSCOLUMNS – Excel columns***

Parameter	<b>XLSCOLUMNS</b>
Applies to commands:	<b>CVTSPLXLS</b>
Dependent on:	

This parameter allows you to fine-tune the decisions made by CoolSpools with regard to the allocation and formatting of data in columns in an Excel spreadsheet.

Up to 100 actions may be specified.

The single value \*NONE indicates that there are no Excel column actions defined.

The options are

<b>*DROP</b>	Drop the column and the data it contains from the output.
<b>*MRGLFT</b>	Merge the column and the data it contains with the column to the left.
<b>*MRGRGT</b>	Merge the column and the data it contains with the column to the right.
<b>*ALNLFT</b>	Align the column to the left.
<b>*ALNRGT</b>	Align the column to the right.
<b>*CVTLBL</b>	Create a label not a numeric cell.

## ***XLSPRINT – Excel print setup***

Parameter	<b>XLSPRINT</b>
Description	<b>Specifies Excel print options</b>
Applies to commands:	<b>CVTSPLXL</b>

Specifies print options for Excel spreadsheets.

### ***Scaling***

How the data is enlarged or reduced when you print so that it fits the required number of pages.

Specify \*FIT and a number of pages wide and tall on the XLSFITPAGES parameter to fit the data to the required number of pages.

Specify \*ADJUST and a percentage on the XLSADJUST parameter to scale the data by that percentage.

Options are:

<b><u>*FIT</u></b>	Fit the data to a number of pages wide and a number of pages tall. The number of pages wide and tall are specified on the dependent parameter XLSFIT.
<b>*ADJUST</b>	Adjust the data by applying a percentage scaling. The percentage by which the data is scaled is specified on the dependent parameter XLSADJUST.

### ***Print gridlines***

Whether gridlines should be printed or not.

Options are:

<b><u>*NO</u></b>	Gridlines are not printed.
<b>*YES</b>	Gridlines are printed.

### ***Printer header rows on each page***

Whether header rows should be printed on each page. The header rows in this context are the rows that were specified as being frozen at the top of the page on the EXCEL parameter.

Options are:

<b><u>*NO</u></b>	The header rows, if there are any, are printed only on the first page.
<b>*YES</b>	The header rows, if there are any, are printed on each page.

### ***Page breaks***

Whether CoolSpools should insert page breaks in the Excel file at the end of each page in the original spooled file.

Options are:

**\*NO**

No page breaks will be inserted.

**\*YES**

A page break will be inserted in the Excel file after the last row of each page in the original spooled file.

### ***Unit of measure***

The unit of measure in which margins are defined (see below)

Options are:

**\*INCH**

Inches

**\*MM**

Millimeters

**\*CM**

Centimeters

### ***Left margin***

The left page margin measured in the units specified (see Unit of Measure above).

### ***Right margin***

The right page margin measured in the units specified (see Unit of Measure above).

### ***Top margin***

The top page margin measured in the units specified (see Unit of Measure above).

### ***Bottom margin***

The bottom page margin measured in the units specified (see Unit of Measure above).

### ***Page header left section***

The text to appear in the left section of the page header.

CoolSpools Spool Converter variables and Excel placeholders are supported on this parameter.

### ***Page header center section***

The text to appear in the center section of the page header.

CoolSpools Spool Converter variables and Excel placeholders are supported on this parameter.

### ***Page header right section***

The text to appear in the right section of the page header.

CoolSpools Spool Converter variables and Excel placeholders are supported on this parameter.

### ***Page footer left section***

The text to appear in the left section of the page footer.

CoolSpools Spool Converter variables and Excel placeholders are supported on this parameter.

### ***Page footer center section***

The text to appear in the center section of the page footer.

CoolSpools Spool Converter variables and Excel placeholders are supported on this parameter.

***Page footer right section***

The text to appear in the right section of the page footer.

CoolSpools Spool Converter variables and Excel placeholders are supported on this parameter.

## ***XLSPROTECT – Excel worksheet protection***

Parameter	<b>XLSPROTECT</b>
Applies to commands:	<b>CVTSPLXLS, CVTSPLXL</b>
Migration notes	Note that CVTSPLXL has two extra elements compared with CVTSPLXLS. Note also that columns can now be optionally left unlocked when a worksheet is protected by using a named style with the locking attribute set appropriately.

Specifies protection options for the spreadsheet.

The default is the single value:

**\*NO**

No protection options are specified.

### ***Protect worksheet***

Options are:

**\*YES**

The worksheet(s) created will be protected so that they cannot be modified.

### ***Worksheet protection password***

**\*NONE**

No password will be required to unprotect the worksheet. The user who opens the file will be able to unprotect the worksheet simply by selecting the appropriate menu option.

**password**

Specify the password that must be entered to unprotect the worksheet.

### ***Encrypted password supplied***

The element does not exist for CVTSPLXLS.

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

**\*NO**

The password supplied on the previous element is in plain text format and not scrambled.

**\*YES**

The password supplied on the previous is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used.

### ***Allow actions***

Defines the actions that can be applied to locked items on a protected worksheet.

Single options are:

<b><u>*DFT</u></b>	(Default) The actions allowed by Excel by default when a worksheet is protected are permitted. Both locked and unlocked cells may be selected, and objects and scenarios may be edited.
<b>*NONE</b>	No actions are permitted on locked cells.

Alternatively, specify the actions to be permitted from the following list:

<b>*DLTCOLS</b>	Deletion of columns
<b>*DLTROWS</b>	Deletion of rows
<b>*AUTOFILTER</b>	Applying autofilters
<b>*EDTOBJ</b>	Editing objects
<b>*EDTSCN</b>	Editing scenarios
<b>*FMTCELLS</b>	Changing the formatting of cells
<b>*FMTCOLS</b>	Changing the formatting of columns
<b>*FMTRROWS</b>	Changing the formatting of rows
<b>*INSCOLS</b>	Inserting columns
<b>*INSROWS</b>	Inserting rows
<b>*INSLINKS</b>	Inserting hyperlinks
<b>*PIVOTTABLE</b>	Applying pivot tables
<b>*SLTUNLOCKED</b>	Selecting unlocked cells
<b>*SLTLOCKED</b>	Selecting locked cells
<b>*SORT</b>	Sorting rows



## ***XLSPRPRTY – Document properties***

Parameter	<b>XLSPRPRTY</b>
Description	<b>Specifies document properties for Excel files.</b>
Applies to commands:	<b>CVTSPLXL</b>
Dependent on:	<b>None</b>
Migration notes	<b>The EXCEL parameter of CVTSPLXL has been considerably simplified compared with the equivalent parameter of CVTSPLXLS through the creation of this separate XLSPRPRTY parameter where Excel file properties are now defined.</b>

This parameter allows you to define file properties for documentation and audit purposes.

The information defined here appears in Excel 2007 when you select:

**Office button -> Prepare -> Properties**

### **Title**

**\*NONE** (Default) The file will have no title.  
**Title\_text** Up to 32 characters of title text.

### **Subject**

**\*NONE** (Default) The file will have no subject.  
**Subject\_text** Up to 32 characters of subject text.

### **Author**

A number of special values are available to help you use this field to document the origin of the file.

**\*NONE** (Default) The file will have no title.  
**\*USRPRF** The user id of the user that created the file, e.g. SALESUSER.  
**\*JOB** The name of the job that created the file, e.g. SALESJOB,  
**\*QUALJOB** The qualified name of the job that created the file, e.g. 123456/SALESUSER/SALESJOB.  
**Author\_text** Up to 32 characters of author text.

### **Manager**

**\*NONE** (Default) The file will have no manager.  
**Manager\_text** Up to 32 characters of manager text.

### **Company**

**\*NONE** (Default) The file will have no company.  
**Company\_text** Up to 32 characters of company text.

### **Category**

**\*NONE** (Default) The file will have no category.

**Category\_text** Up to 32 characters of category text.

## Keywords

**\*NONE** (Default) The file will have no keywords.  
**Keywords\_text** Up to 128 characters of keywords text.

## Comments

**\*NONE** (Default) The file will have no comments.  
**Comments\_text** Up to 256 characters of comments text.

## Document content status

**\*NONE** (Default) The file will have no document content status  
**Status\_text** Up to 32 characters of text describing the status of the document content (e.g. "Draft", "Final", "Approved" etc.)

## **RSTSPLF Command**

The RSTSPLF (Restore Spooled File) command complements SAVSPLF command and allows spooled files saved by that command to be restored.

The command has just three parameters.

### **FROMSTMF – From stream File**

The **FROMSTMF** (From Stream File) parameter specifies the name of the stream file archive from which you wish to restore a spooled file.

You may specify the name of the stream file in either of two ways.

The first option is to enter a full path name on this parameter, that is the complete directory path and the name of the file to be created or replaced.

The second option is to enter just the name of the file itself. You will then need to specify the directory path in which that file will be saved on the **FROMDIR** (From Directory) parameter.

This file **MUST** be a stream file previously created using the CVTSPLSTMF...TOFMT(\*SAV) option, the CVTSPLSAV command or the SAVSPLF command.

A generic path name may be specified. All stream files that match the pattern specified will be processed.

### **NEWOWN – New owner**

The **NEWOWN** (New Owner) parameter specifies the user profile who should be assigned ownership of the spooled file when it is restored.

Options are:

<b>*SPLF</b>	(Default) The owner of the restored spooled file will be the same as the owner of the original spooled file. However, if the user profile which owned the original spooled file does not exist on the system to which the spooled file is being restored, an error will occur.
<b>*CURRENT</b>	The owner of the restored spooled file will be the user running this command.
<b>User_profile</b>	Specify the user who should own the restored spooled file.

### **Example:**

```
RSTSPLF          FROMSTMF(/spools/2001/june sales.sav)  
NEWOWN(PETE)
```

Here the RSTSPLF command is used to restore a previously saved spooled file from a stream file called **sales.sav** in a directory called **/spools/2001/june**. The owner of the restored spooled file will be user PETE.

## ***OUTPTY – Output priority***

The **OUTPTY** (Output Priority) parameter allows you to override the output priority attribute of the spooled file when it is restored.

Options are:

<b>*SPLF</b>	(Default) The output priority of the original spooled file will be used. However, if this output priority exceeds the maximum output priority allowed for the user who is restoring the spooled file, the restore operation will fail. This error can be avoided by specifying a different (lower) output priority on this parameter.
<b>Output_pty</b>	Specify the priority to be used (1-9).

## **MRGPDF**

The MRGPDF (Merge PDF) command lets you merge two or more PDF files to create a composite PDF.

The PDF files can be files created with CoolSpools or any other application that generates PDF files.

Parameters are as follows:

### ***FROMPDF - PDF files to merge***

Specifies the files that are to be merged together to create the new file. You may specify a minimum of 2 and a maximum of 16 files to be combined.

The order in which the files are listed on this parameter is significant: the files will be combined in the order in which they appear on this parameter. Each file is appended to the file(s) which precede it in the list.

Each item on the list consists of three elements:

- **path name of the file to be processed.**
- **password**
- **page rotation**

#### **Path name of the file to be processed**

Specify the absolute or relative path of the PDF file in the IFS.

Refer to "[Understanding IFS Path Names](#)" above for a discussion of how to specify the path name where the file should be saved. Further information on path names is also available at <http://publib.boulder.ibm.com/system/i/v5r2/ic2924/info/rbam6/rbam6pathnames.htm>.

#### **Password**

Options are:

<b><u>*NONE</u></b>	The PDF file does not need a password to be opened.
<b>Password</b>	If the file has been secured with a password, you will need to enter a password on this parameter otherwise it cannot be processed. If the file has been secured in such a way that its contents cannot be copied or modified without supplying the owner password, you will be required to enter the owner password to process the file with MRGPDF.

#### **Page rotation**

The rotation angle to be applied to each page in the included file.

Where pages in the various input files have different page orientations, it may be convenient to apply a rotation to pages on one or more files in order to bring them into a single, consistent orientation.

Options are:

<b><u>*FROMPDF</u></b>	The orientation is the same as that in the original file.
<b>0</b>	No rotation is applied.
<b>90</b>	A 90-degree rotation is applied.
<b>180</b>	A 180-degree rotation is applied.
<b>270</b>	A 270-degree rotation is applied.
<b>360</b>	A 360-degree rotation is applied.

### ***TOPDF - Merged PDF file***

Specify the full IFS path name of the file you wish to create.

Refer to "[Understanding IFS Path Names](#)" above for a discussion of how to specify the path name where the file should be saved. Further information on path names is also available at <http://publib.boulder.ibm.com/system/i/v5r2/ic2924/info/rbam6/rbam6pathnames.htm>.

Options are:

<b><u>*FROMPDF</u></b>	The name of the file to be created will be the same as the first file listed on the FROMPDF parameter.
<b>path_name</b>	Specify the path for the new file.

### ***REPLACE - Replace existing PDF***

This parameter determines whether the file specified on the TOPDF parameter will be replaced if it already exists.

Options are:.

<b>*NO</b>	The file is not replaced. An error occurs if the file already exists.
<b>*YES</b>	The file is replaced.

### ***PASSWORD - Merged PDF file security***

This parameter determines the security applied to the merged file which is created.

There are five elements to this parameter:

- **Password protect merged file?**
- **User password**
- **Owner password**
- **Allow printing?**
- **Allow modifications?**
- **Allow copying of text?**
- **Allow annotation?**

## Password protect merged file?

This indicates what passwords, if any, the merged file should have

Options are:

<b><u>*FROMPDF</u></b>	The security will be the same as that of the first file specified on the FROMPDF parameter. If that file has no passwords, neither will the merged file. If that file has passwords or other security restrictions, the merged file will have the same passwords.
<b>*NO</b>	The file will have no passwords or security restrictions.
<b>*YES</b>	The file will have either one or two passwords.
<b>*RESTRICT</b>	The merged file will have no passwords but the operations that can be performed on it will be restricted.

## User password

The second element is the User password. This is where you define the user password for the merged file. The user password will open the file, but only those operations which are allowed by the later elements of this parameter can be performed.

If you leave this field blank, the file will have no user password. When the file is opened, the user will not be prompted to enter a password, but operations will be restricted to those that are permitted on the later elements of this parameter.

## Owner password

The third element is the Owner password. This is where you define the owner password for the merged file. The owner password will open the file and allow all operations to be performed, irrespective of restrictions indicate on the later elements of this parameter.

If you leave this field blank, the file will have no owner password and there will be no way of performing restricted operations on the file.

The remaining parameters control which operations can be performed on the file when the file has not been opened with the owner password.

## Allow printing?

Options are:

<b><u>*YES</u></b>	Printing is permitted.
<b>*NO</b>	Printing is not permitted.

## Allow modifications?

Options are:

<b><u>*YES</u></b>	Modifications are permitted.
<b>*NO</b>	Modifications are not permitted.

## Allow copying of text?

Options are:

**\*YES**  
**\*NO**

Copying of text is permitted.  
Copying of text is not permitted.

## Allow annotation?

Options are:

**\*YES**  
**\*NO**

Annotation is permitted.  
Annotation is not permitted.

## ***NOTFOUND - File not found action***

This parameter controls how MRGPdf behaves when one or more of the files listed on the FROMPDF parameter cannot be found.

Options are:

**\*STOP**

If one or more of the files listed on the FROMPDF parameter cannot be found, processing will stop and an error is returned.

**\*CONTINUE**

If one or more of the files listed on the FROMPDF parameter cannot be found, the file is skipped and processing continues with the next file.

Note that MRGPdf requires that at least two of the files listed on the FROMPDF parameter must exist before it can do any processing. Also, if TOPDF(\*FROMPDF) is specified, the first file listed on the FROMPDF parameter must exist, since otherwise MRGPdf cannot determine the name to be given to the merged output file.

## ***AUT - Public data authority***

The AUT (Public Data Authority) parameter allows you to define the data rights given to \*PUBLIC for the merged file.

Note that the owner of the output file will be the user running this command. If the first file on the FROMPDF parameter is owned by a different user, the ownership will change.

Note also that authorities and authorization lists associated with the directory in which the output file resides will be inherited automatically by the output file, but that any private authorities associated with the first file on the FROMPDF parameter will be lost.

Options are:

**\*FROMPDF**

Public data rights are copied from the first file on the FROMPDF parameter.

**\*R**

Read only

**\*W**

Write only

**\*X**

Execute only

**\*RW**

Read and write



<b>*RX</b>	Read and execute
<b>*WX</b>	Write and execute
<b>*RWX</b>	Read, write and execute (all)
<b>*NONE</b>	No authority

## **ADDPDFSGN**

The ADDPDFSGN (Add PDF Signature) command adds a digital signature to an existing PDF file.

Parameters are as follows:

### ***PDFFILE - PDF file***

Specifies the file to which the digital signature should be added.

### ***PDFPWD – Owner password***

If the PDF file is password-protected, specify the owner password needed to modify the file.

The default is the single option:

**\*NONE**                      The file is not password-protected.

Other options:

#### **Password**

Specify the owner's password.

#### **Encrypted password supplied**

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

<b><u>*NO</u></b>	The password supplied on the previous element is in plain text format and not scrambled.
<b>*YES</b>	The password supplied on the previous element is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used.

### ***CTFFILE - Certificate file***

Specifies the path to the stream file containing a PKCS12 digital certificate key.

### ***CTFPWD – Certificate password***

Specifies the password for the certificate file.

## Password

Specify the file password.

### Encrypted password supplied

Whether or not the password supplied on the previous element is supplied in the encrypted form returned by CoolSpools' DSPENCPWD (Display Encrypted Password) command. See the discussion of [encrypted passwords](#) above.

DSPENCPWD applies an encryption algorithm to a password and returns a scrambled version of that password to you. If you specify the scrambled password on the previous element, and specify \*YES here, CoolSpools Spool Converter will unscramble the password for you before using it. The main purpose of this facility is to avoid the need to hold passwords in plain text form in source code.

Options are:

<b><u>*NO</u></b>	The password supplied on the previous element is in plain text format and not scrambled.
<b>*YES</b>	The password supplied on the previous is in the scrambled form returned by DSPENCPWD. It will be automatically unscrambled before being used.

### REASON – Reason for signing

Allows you to describe the reason why the document is being signed.

Options are:

<b><u>*NONE</u></b>	No reason is specified.
<b>reason_text</b>	Free format text describing the reason for signing.

### LOCATION - Location

Allows you to describe the location where the document is being signed.

Options are:

<b><u>*NONE</u></b>	No location is specified.
<b>location_text</b>	Free format text describing the location of signing.

### CONTACT - Signing contact information

Allows you to specify a contact for enquiries relating to the signature.

Options are:

<b><u>*NONE</u></b>	No contact is specified.
<b>contact_text</b>	Free format text specifying a contact.

### VISIBLE - Visible signature?

Whether the signature will have some visible representation in the file.

Options are:

<b><u>*YES</u></b>	The signature will be visible.
--------------------	--------------------------------

**\*NO** The signature will be invisible.

### ***PAGNBR - Show on page number***

Which page of the PDF the signature should appear on. Ignored if the signature is not visible.

Options are:

<b>*<u>FIRST</u></b>	The signature will appear on the first page.
<b>*LAST</b>	The signature will appear on the last page.
<b>page_number</b>	Specify the page number where the signature should appear.

### ***IMAGE - Display image file***

Specifies the path to an image file (e.g. a JPEG) which will be used to provide a pictorial representation of the signature.

Options are:

<b>*<u>NONE</u></b>	There is no pictorial presentation of the signature.
<b>image_path</b>	Specify the path to the image file.

### ***XCOORD - X coordinate***

Specifies the horizontal coordinate of the graphical representation of the signature.

Options are:

<b>*<u>LEFT</u></b>	At the left margin of the page
<b>*RIGHT</b>	At the right margin of the page
<b>*CENTER</b>	In the center of the page
<b>X_coordinate</b>	Specify the X coordinate in the units defined below.

### ***YCOORD - Y coordinate***

Specifies the vertical coordinate of the graphical representation of the signature.

Options are:

<b>*<u>TOP</u></b>	At the top margin of the page
<b>*BOTTOM</b>	At the bottom margin of the page
<b>*CENTER</b>	In the center of the page
<b>Y_coordinate</b>	Specify the Y coordinate in the units defined below.

### ***WIDTH - Width***

Specifies the horizontal dimension of the graphical representation of the signature.

Options are:

**\*DFT**

The actual width of the image as defined in the image properties.

**width**

Specify the width of the image in the units defined below.

***HEIGHT - Height***

Specifies the vertical dimension of the graphical representation of the signature.

Options are:

**\*DFT**

The actual height of the image as defined in the image properties.

**height**

Specify the height of the image in the units defined below.

**UOM - Unit of measure**

Defines the units used to specify the dimensions and coordinates.

Options are:

**\*MM**

Millimeters

**\*CM**

Centimeters

**\*INCH**

Inches

## **Report Definitions and Report Maps**

When converting a spooled file to formats such as PDF, the focus is very much on the appearance of the document that is created and we are concerned with things like font typefaces and size and the positioning on the page of text and graphical items. The spooled file itself contains all of the information CoolSpools needs in order to generate a PDF that looks just like the paper document created when the spooled file is printed.

However, when converting a spooled file to other file formats, notably Excel and more particularly XML, we are less concerned about the appearance of the document than we are in its semantic content, about the information it contains and how that information is structured. Unfortunately, spooled files normally contain little or no metadata to assist CoolSpools. This information can be obtained reliably through user input.

Therefore, in order to produce better Excel output and in order to allow the creation of XML, CoolSpools now supports the creation of **report definitions** and **report maps**.

A **report definition** defines the structure and content of a spooled file.

Thus, it describes the **input** to the conversion process and consists of:

- **Report lines** that describe the different lines that occur in the report
- **Report items** that specify the location and format of data items
- **Report sections** that define the relationship between lines in the report

A **report map** defines the structure and content of a stream file produced by CoolSpools and maps the items defined in a report definition to their required place in the file.

Thus, it describes the **output** from the conversion process. There are currently two types of report map:

- **Report-to-Excel maps** describing Excel output
- **Report-to-XML maps** describing XML output

The CVTSPLXL (Convert Spooled File to Excel) and CVTSPLXML (Convert Spooled File to XML) commands require the use of an **report map** of the appropriate type which specifies the structure of the Excel and XML file you wish to create from your spooled file.

Refer to the worked example below for a detailed description of the process of creating and using report definitions and report maps.

## **Commands related to Report Definitions**

### **CRRPTDFN – Create Report Definition**

The CRRPTDFN (Create Report Definition) command creates a report definition describing the semantic content and structure of a spooled file.

Once you have created your report definition you need to specify the different line types, data items and sections it comprises. See the DSNRPTDFN (Design Report Definition) command below for details of how to specify a report definition interactively using a sample copy of the spooled file. Before you can use DSNRPTDFN, however, you must first create the report definition with CRRPTDFN (or by pressing F6 from the WRKRPTDFN screen) so that you can specify some basic attributes.

#### ***REPORTNAME – Report definition name***

Specify the name you wish to give to the report definition.

Report definition names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

#### ***LPI – Lines per inch***

In order to process the spooled file that this report definition describes, CoolSpools needs to be able to format the content of the spooled file as text in such a way that positions on the page can be referred to by means of rows and columns. While this is fairly simple with basic \*SCS spooled files, it can be a lot more complex for spooled file types such as \*AFPDS and \*USERASCII, which may use a variety of font sizes, including proportional fonts.

The method CoolSpools uses to convert spooled files to text and determine rows and columns is the same as that used by IBM's DSPSPLF command, namely that a single LPI (Lines Per Inch) and CPI (Characters Per Inch) value is assumed throughout the spooled file, irrespective of whether different font sizes are being used.

This method has the advantage of simplicity, but problems can arise where the actual font size used for a piece of text is at odds with the assumed LPI or CPI values; specifically, text may be truncated or overwritten by other pieces of text.

Where this occurs, you should choose higher LPI and CPI values until settings are identified that render all text in the spooled file in a satisfactory manner so that no text content is lost.

Bear in mind that if you specify an LPI or CPI value for a report definition other than that implied by the spooled file attributes, the coordinates you give when defining report items need to be consistent with the LPI or CPI value of the report definition, not the spooled file, and will differ from those suggested when you view the spooled file in DSPSPLF.

Options are:

#### **\*SPLF**

The LPI value implied by the spooled file attributes is used. Note that this may not have been set appropriately

when the printer file from which the spooled file was generated was created and could therefore be misleading. This is especially true of \*USERASCII spooled files

**LPI**

Specify an LPI value between 0.01 and 99.9.

### ***CPI – Characters per inch***

See the LPI above for a discussion of the use of this parameter.

Options are:

**\*SPLF**

The CPI value implied by the spooled file attributes is used. Note that this may not have been set appropriately when the printer file from which the spooled file was generated was created and could therefore be misleading. This is especially true of \*USERASCII spooled files

**CPI**

Specify a CPI value between 0.01 and 99.9.

### ***DFTUSEAUT - Default use authority***

The default authority to use this report definition.

Individual user authorities to the report definition can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a report definition is

ARIADNE\_RPT\_DFN\_nnnnnnnnnn\_USE

where nnnnnnnnnn is the internal report definition ID, which is displayed by DSPRPTDFN.

Options are:

**\*ALLOWED**

By default, users other than the user creating the definition are permitted to use it.

**\*DENIED**

By default, users other than the user creating the definition are not permitted to use it.

### ***DFTCHGAUT - Default change authority***

The default authority to change or delete this report definition.

Individual user authorities to the report definition can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a report definition is

ARIADNE\_RPT\_DFN\_nnnnnnnnnn\_CHG

where nnnnnnnnnn is the internal report definition ID, which is displayed by DSPRPTDFN.

Options are:

**\*DENIED**

By default, users other than the user creating the report definition are not permitted to change, delete or manage it.



**\*ALLOWED**

By default, users other than the user creating the report definition are permitted to change, delete or manage it.

***TEXT 'description'***

Specify up to 50 characters of free-format descriptive text to help you identify the report definition.

Options are:

**\*BLANK**

No text is specified.

**Text**

Specify the text 'description'.

***CCSID - Spooled file CCSID***

Specify the CCSID to assume when processing converting the spooled files using this report definition.

Options are:

**\*SPLF**

The CCSID of the data in the spooled file is taken from the spooled file content and attributes, or, if those do not give an indication of the encoding, the job CCSID is used.

**\*JOB**

The CCSID of the job in which the report definition is being used is taken.

**\*SYSVAL**

The CCSID of the system using where the report definition is being used is taken.

**\*USER**

The CCSID of the user profile using the report definition is taken.

**CCSID**

Specify the CCSID of data in the spooled file.

***DATFMT - Spooled file date format***

Specify the date format to assume when processing dates in spooled files using this report definition.

Options are:

**\*JOB**

The DATFMT job attribute is used to determine the format of dates in the spooled file.

**\*SYSVAL**

The QDATFMT system value is used to determine the format of dates in the spooled file.

**\*DMY**

Day Month Year format is assumed.

**\*MDY**

Month Day Year format is assumed.

**\*YMD**

Year Month Day format is assumed.

***DATSEP - Spooled file date separator***

Specify the date separator to assume when processing dates in spooled files using this report definition.

Options are:

<b><u>*JOB</u></b>	The DATSEP job attribute is used to determine the format of dates in the spooled file.
<b>*SYSVAL</b>	The QDATSEP system value is used to determine the format of dates in the spooled file.
<b>*NONE</b>	No date separator is used.
<b>Separator</b>	Specify the separator character to assume.

### ***CURSYM - Spooled file currency symbol***

Specify the currency symbol to assume when interpreting numeric values in spooled files using this report definition.

Options are:

<b>*SYSVAL</b>	The QCURSYM system value is used to interpret numeric values in the spooled file.
<b>Cur_sym</b>	Specify the currency symbol to assume.

### ***DECPOINT - Spooled file decimal point***

Specify the decimal point character to assume when interpreting numeric values in spooled files using this report definition.

Options are:

<b><u>*JOB</u></b>	The DECFMT job attribute is used to determine the decimal point character used when interpreting numeric values in the spooled file.
<b>*SYSVAL</b>	The QDECFMT system value is used to determine the decimal point character used when interpreting numeric values in the spooled file.
<b>Dec_point</b>	Specify the decimal point character to assume.

### ***THOUSANDS - Spooled file 1000s separator***

Specify the thousands separator character to assume when interpreting numeric values in spooled files using this report definition.

Options are:

<b><u>*JOB</u></b>	The DECFMT job attribute is used to determine the thousands separator character used when interpreting numeric values in the spooled file.
<b>*SYSVAL</b>	The QDECFMT system value is used to determine the thousands separator character used when interpreting numeric values in the spooled file.
<b>*NONE</b>	Assume that no thousands separator is used.
<b>1000s_sep</b>	Specify the thousands separator character to assume.

The following commands also operate on report definitions. Parameters are only described where they differ significantly from those of the CRTRPTDFN command described above.

## **CHGRPTDFN – Change Report Definition**

The CHGRPTDFN (Change Report Definition) command modifies an existing report definition.

See CRTRPTDFN above for a discussion of the various parameters.

## **CPYRPTDFN – Copy Report Definition**

The CPYRPTDFN (Copy Report Definition) command copies a report definition and its associated report lines, items and sections.

### ***FROMREPORT – From report definition name***

Specify the name of the report definition you wish to copy.

### ***TOREPORT – To report definition name***

Specify the name of the report definition you wish to create, based on the report definition being copied.

The remaining parameters allow attributes to be modified while the report is being copied. See CRTRPTDFN above for a discussion of these parameters.

## **DLTRPTDFN – Delete Report Definition**

The DLTRPTDFN (Delete Report Definition) command deletes a report definition.

See CRTRPTDFN above for a discussion of the various parameters.

### ***REPORTNAME – Report definition name***

Specify the name of the report definition you wish to delete.

### ***CHKDEPMAP – Check dependent maps***

Whether the system should check for report maps that depend on this report definition.

Options are:

#### **\*YES**

The system will check for report maps that depend on this report definition. If any such maps are found, the system will issue an error and deletion of the report definition will fail.

#### **\*NO**

No check for dependent maps will be made. If any such maps exist, they will become unusable and an error will occur if you try to use them.

## **DSPRPTDFN – Display Report Definition**

The DSPRPTDFN (Display Report Definition) command displays details of a report definition.

## **RNMRPTDFN – Rename Report Definition**

The RNMRPTDFN (Rename Report Definition) command renames a report definition.

### ***REPORTNAME – Report definition name***

Specify the name of the report definition you wish to rename.

### ***NEWREPORT – New report definition name***

Specify the new name for the report definition.

## **RTVRPTDFN – Retrieve Report Definition**

The RTVRPTDFN (Retrieve Report Definition) command retrieves CL source for creating a report definition and all its associated report lines, report items and report sections. This provides a convenient way of saving and distributing a report definition to other systems. The source that is retrieved can be easily converted to a program which can be run to create the report definition.

### ***REPORTNAME – Report definition name***

Specify the name of the report definition for which you wish to retrieve source.

### ***SRCFILE – Source file***

Specify the qualified name of the source file into which the source should be retrieved. The file and library must already exist.

### ***SRCMBR – Source member***

Specify the name of the source member into which the source should be retrieved. If the member does not already exist, it will be created.

### ***MBROPT – Source member***

Whether an existing member is replaced or appended to.

Options are:

<b><u>*REPLACE</u></b>	If the member already exists, it will be replaced.
<b>*ADD</b>	If the member already exists, the retrieved source will be appended to it.

## **SAVRPTDFN – Save Report Definition**

The SAVRPTDFN (Save Report Definition) command saves one or more report definitions into a stream file. The RSTRPTDFN (Restore Report Definition) command can be used to restore report definitions from the stream file. This facility provides a means of backing up report definitions and also of distributing them to other systems.

The stream file that is created is a zipped XML document.

## ***REPORTNAME – Report definition name***

Specify the name of one or more report definitions you wish to save or specify **\*ALL** to save all report definitions.

## ***TOSTMF – To stream file***

Specify a path for the stream file in which the report definitions are saved.

If only a single name was specified on the REPORTNAME parameter, you can use the special default value **\*RPTDFN**, in which case CoolSpools will save the report definition in the current directory as **report\_definition\_name.rpt**, where report\_definition\_name is the name specified on the REPORTNAME parameter.

## ***REPLACE - Replace existing stream file***

Whether the stream file is replaced if it already exists.

Options are:

<b><u>*NO</u></b>	If the stream file exists, an error will occur and it will not be replaced.
<b>*YES</b>	If the stream file exists, it will be replaced.

## ***AUT- Public data authority***

The public data authority to assign to the stream file when it is created.

Options are:

<b><u>*R</u></b>	Read only
<b>*W</b>	Write only
<b>*X</b>	Execute only
<b>*RW</b>	Read and write
<b>*RX</b>	Read and execute
<b>*WX</b>	Write and execute
<b>*RWX</b>	Read, write and execute (all)
<b>*NONE</b>	No authority
<b>autl_name</b>	Alternatively, specify the name of an authorization list. This authorization list will be associated with the stream file and authorities for *PUBLIC assigned from the authorization list.

## ***RSTRPTDFN – Restore Report Definition***

The RSTRPTDFN (Restore Report Definition) command restores one or more report definitions from a stream file containing report definitions saved by the SAVRPTDFN (Save Report Definition) command.

## ***REPORTNAME – Report definition name***

Specify the name of one or more report definitions you wish to restore or specify **\*ALL** to restore all report definitions saved in the stream file.

## ***FROMSTMF – From stream file***

Specify a path for the stream file in which the report definitions were saved.

If only a single name was specified on the REPORTNAME parameter, you can use the special default value \*RPTDFN, in which case CoolSpools will look in the current directory for a stream file called **report\_definition\_name.rpt**, where report\_definition\_name is the name specified on the REPORTNAME parameter.

## ***REPLACE - Replace existing report***

Whether reports definitions are replaced if they already exist.

Options are:

<b><u>*NO</u></b>	If a report definition already exists on the system, an error will occur and it will not be replaced.
<b>*YES</b>	If a report definition already exists on the system, it will be replaced.

## **WRKRPTDFN – Work with Report Definition**

The WRKRPTDFN (Work with Report Definition) command lets you work with a list of report definitions.

## **DSNRPTDFN – Design Report Definition**

The DSNRPTDFN (Design Report Definition) command lets you design a report definition, that is, specify report lines, report data items and report sections by reference to a sample spooled file displayed on screen.

You must first create the report definition with CRTTRPTDFN.

Refer to the Worked Example below for a description of how to use this facility.

## ***REPORTNAME – Report definition name***

Specify the report definition you wish to design.

## ***SPLFNAME - Based on spooled file***

Specify the name of a spooled file which will be selected and displayed on screen as a template for you to work with in designing the report definition.

Options are:

<b><u>*SELECT</u></b>	You will be prompted to select the spooled file from a list displayed on screen.
-----------------------	--

**\*RPTDFN**                      The spooled file has the same name as the report definition.

**splf\_name**                      Specify the name of the spooled file

### ***JOB-Spoiled file job***

Specify the qualified name of the job in which the spooled file was created.

This parameter is ignored if SPLFNAME(\*SELECT) is specified.

Options are:

**\***                                      The spooled file is associated with the current job  
**\_**  
**job\_name**                      Specify the name of the job.

### ***SPLNBR-Spoiled file number***

Specify the number of the spooled file.

This parameter is ignored if SPLFNAME(\*SELECT) is specified.

Options are:

**\*ONLY**                              There is only one spooled file of the specified name in the specified job.  
**\*LAST**                              Select the last spooled file of the specified name in the specified job.  
**spl\_nbr**                              Specify the spooled file number.

The following commands allow you to define the content and structure of a report by adding report lines, report data items and report sections to a report definition.

These commands can also be invoked by taking options from the WRKRPTDFN screen.

See the DSNRPTDFN (Design Report Definition) command or details of an alternative method of specifying a report definition interactively using a sample copy of the spooled file.

## **ADDRPTLIN – Add Report Line**

The ADDRPTLIN (Add Report Line) command adds a report line to a report definition.

A report line describes a line in a report in the sense of a line of text that conforms to a particular model or pattern. This could be a heading line, a summary line, a detail line etc. In the Worked Example below, the sample Customer Orders report comprises a dozen or so different line types, such as:

- report headings
- region headings
- state headings
- customer headings
- order details
- customer totals
- state totals

- region totals etc.

When CoolSpools is analyzing a spooled file in order to convert it to text and extract information from it, its first task is to try to match each line of the spooled file against a report line from the report definition it is using.

For every different type of line in your report from which you wish to extract information and which defines the start or end of a section, you should add a report line to your report definition.

In order for CoolSpools to analyze your spooled file successfully and extract information from it, it is vital that CoolSpools can correctly identify, for each line in the spooled file, which report line it corresponds to. Spooled file lines which cannot be matched to a report line will not be processed. Spooled file lines which are matched to the wrong report line will produce garbage.

Parameters are as follows. Many of these specify rules which CoolSpools will use to determine whether lines of text in the spooled file it is processing should be matched to this report line or not.

### ***REPORTNAME – Report definition name***

Specify the name of the existing report definition to which you wish to add the line type.

Report definition names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

### ***LINENAME – Report line name***

Specify the name you wish to give to the line type.

Report line names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

### ***LINETYPE – Line type***

Specify the type of line you are describing. This information has no function other than to serve as documentation currently but may be used in other ways in future versions of CoolSpools.

Options are:

<b><u>*DETAIL</u></b>	A detail line.
<b>*PAGHDG</b>	A page heading line
<b>*COLHDG</b>	A column heading line
<b>*SUMMARY</b>	A summary line
<b>*OTHER</b>	Some other kind of line

### ***PAGERANGE – Can occur on page numbers***

Specify the earliest and latest page numbers in the report which can include the report line you are adding. For example, often report headings can appear only on the first page of a report and grand totals only on the last.



The page range is defined in terms of a pair of page numbers with optional associated page offsets.

### From page number

The earliest possible page on which this line type can occur.

Options are:

<b><u>*FIRST</u></b>	The first page in the spooled file.
<b>*LAST</b>	The last page in the spooled file.
<b>page_number</b>	Specify the earliest page number.

### Offset

A number which is added to the page number above to give the actual page number.

Options are:

<b><u>*NONE</u></b>	No offset is applied.
<b>page_offset</b>	Specify the page offset to apply.

This is most commonly used in the form of a negative page offset together with the option \*LAST for the page number above. For example, the pair of values:

page number:	*LAST
page offset:	-1

denotes the penultimate page of the spooled file (last page but one).

### To page number

The latest possible page on which this line type can occur.

Options are:

<b><u>*LAST</u></b>	The last page in the spooled file.
<b>*FIRST</b>	The first page in the spooled file.
<b>page_number</b>	Specify the latest page number.

### Offset

A number which is added to the page number above to give the actual page number.

Options are:

<b><u>*NONE</u></b>	No offset is applied.
<b>page_offset</b>	Specify the page offset to apply.

This is most commonly used in the form of a negative page offset together with the option \*LAST for the page number above. For example, the pair of values:

page number:	*LAST
page offset:	-1

denotes the penultimate page of the spooled file (last page but one).

## ***LINERANGE – Can occur on line numbers***

Specify the earliest and latest line numbers on the page on which the report line you are adding can occur. For example, often page headings can appear in the first few lines of a page and footings on the last few lines.

The line range is defined in terms of a pair of line numbers. Note that these will be calculated using the LPI value specified for the report definition when it was created.

### **From line number**

The earliest possible line on which this report line can occur.

Options are:

<b><u>*FIRST</u></b>	The first line of the page.
<b>*LAST</b>	The last line of the page.
<b>line_number</b>	Specify the earliest line number.

### **To line number**

The latest possible line on which this report line can occur.

Options are:

<b><u>*LAST</u></b>	The last line of the page.
<b>*FIRST</b>	The first line of the page.
<b>line_number</b>	Specify the latest line number.

## ***RULETYPE – Rule type***

Specify how CoolSpools will identify lines of this type.

The options are:

<b><u>*LINNBR</u></b>	The line type can be identified by its line number alone. Any line on the range of pages specified which falls in the line range specified will be selected and assigned to the line type being defined.
<b>*REPEAT</b>	<p>The line type can be identified by its line number alone, but is part of a repeat group. A repeat group is a group of related lines that occur together on the page and are repeated down the page, something like this:</p> <p>Line 1 Line 2 Line 3 Line 4 Line 1 Line 2 Line 3</p>

Line 4  
Line 1  
Line 2  
Line 3  
Line 4

...

The from- and to-line numbers specify the earliest and latest lines on the page between which this particular line type can occur. You must also specify a repeat group depth on the REPEAT parameter which identifies how many lines there are in the group.

Any line on the range of pages specified which fulfils the line number criteria specified will be selected and assigned to the line type being defined.

For example, taking the repeat group above, the repeat group depth is 4. If the repeat group starts on line 21 and ends on line 50, then each line would be specified as follows:

Line type	From line number	To line number	Possible lines on which this line type can occur
Line 1	21	47	21, 25, 29, 33...
Line 2	22	48	22, 26, 30, 34...
Line 3	23	49	23, 27, 31, 35...
Line 4	24	50	24, 28, 32, 36...

**\*DEFAULT**

This value identifies the line type as the default, i.e. the line type that is assigned if no other line type is selected.

**Use this option with caution. Misuse of this facility can result in garbage being produced if lines are assigned to this report line incorrectly.**

**\*RULE**

This value indicates that you wish to apply one or more tests to the line to determine its type. The

tests are specified on the LINERULES parameter below.

## ***RULEPTY - Rule evaluation priority***

The rule evaluation priority is a number between 1 and 999 which specifies the order in which the report lines are checked against each spooled file line. Since the first matching report line will be selected, you can use the rule evaluation priority to prioritize one report line before another. Typically, you would prioritize the most specific tests first and the most general tests later, with perhaps a default or “catch-all” report line last of all.

For example, imagine you have a report with two summary lines which contain the following labels that you propose to use to identify them:

**Customers who purchased this month**  
**Customers who purchased this month last year**

If you define a rule to check for the text “Customers who purchased this month”, it will potentially match **both** lines and select the wrong line the for the line where the text is “Customers who purchased this month last year”. In order to select the right line types, you could set the rule priorities like this:

**Customers who purchased this month**                      **Priority = 100**  
**Customers who purchased this month last year**      **Priority = 050**

thus ensuring that the second rule is tested first (because its rule priority setting is lower) and that rule will select just the line where the text is “Customers who purchased this month last year” leaving the other rule to select the line where the text is “Customers who purchased this month”.

## ***SECTION – Section name***

Allows you indicate that this report line is part of a report section.

Report lines can be, and usually are, part of a report section. However, in order to define a section, you need to specify the report lines with which it starts and ends and the report lines it includes.

There is consequently a mutual dependency between report lines and report sections and you cannot define report sections until the report lines have been defined. Therefore, you must define things in the following sequence:

- Add the report lines (with SECTION(\*NONE) specified
- Add report items to those report lines
- Add report sections, referencing those line and items

Normally, therefore, when you first define a report line, you will need to specify SECTION(\*NONE) here initially. Then, when you have defined the report sections, you can define a report line as part of a section either by:

- including the report line in the list of lines included in the section (LINENAMES parameter of ADDRPTSCT)
- specifying the section name on this parameter of the CHGRPTLIN command

## **TEXT – Text ‘description’**

Specify up to 50 characters of free-format descriptive text to help you identify the report line.

Options are:

<b><u>*BLANK</u></b>	No text is specified.
<b>Text</b>	Specify the text ‘description’.

## **REPEAT –Depth of repeat group**

The number of lines in the repeat group. Used in conjunction with the RULETYPE(\*REPEAT) option described above.

Options are:

<b><u>*NONE</u></b>	This line is not part of a repeat group. Invalid if RULETYPE(*REPEAT) was specified.
<b>repeat_depth</b>	Specify the number of lines in the repeat group.

## **LINERULES – Line rules**

When RULETYPE(\*RULE) is specified, this parameter lets you define one or more tests to be applied to data on the page this line is on.

The piece of data is identified by a line number and character position on the page. It cannot be identified by a report item name because the identification of report items is dependent on the identification of report lines, and to do so would therefore create a circular definition.

### **Relationship**

Specifies the logical relationship between one test and the next

Options are:

<b><u>*IF</u></b>	Mandatory for the first test and allowed only on the first test.
<b>*AND</b>	Indicates that this test is part of the same logical group as the previous line. The rule will be true only if the result of all tests since and including the previous test starting with *IF or *OR test are true.
<b>*OR</b>	Indicates that this test starts a new OR group. The rule will be true if the combined result of this group or any other group starting with *IF or OR is true.

### **Line number**

Specifies the line number on which the piece of data to be tested can be found on the current page.

Note that this will be calculated using the LPI value specified for the report definition when it was created.

Options are:

<b><u>*CURRENT</u></b>	The test applies to the current line, i.e. the line which is being analyzed to determine its type.
------------------------	--

<b>*FIRST</b>	The test applies to the first line on the page.
<b>*LAST</b>	The test applies to the last line on the page.
<b>Line number</b>	Specify the absolute line number on the page of the line to be tested, e.g. a value of 1 here would cause the test to be applied to Line 1 on the page.

## Offset

Specifies the offset from the line number specified on the previous element to the actual line number to be tested.

Options are:

<b><u>*NONE</u></b>	No offset is applied. The line number alone identifies the line to be tested.
<b>Offset</b>	<p>Specify a number that is added to the line number to obtain the actual line number to be tested.</p> <p>Typically, this is used in conjunction with a line number of *CURRENT or *LAST to specify a line number relative to the current line or the last line on the page.</p> <p>For example:</p> <p style="padding-left: 40px;">Line: *CURRENT    Offset: 1</p> <p>denotes the line following the line being analyzed, while:</p> <p style="padding-left: 40px;">Line: *LAST        Offset: -1</p> <p>denotes the last but one line on the page.</p>

## Char Position

Identifies the position on the line to be tested.

Options are:

<b>Character position</b>	<p>Specify the starting position on the line of the first character to be tested or compared to the comparison value.</p> <p>Note that this will be calculated using the CPI value specified for the report definition when it was created.</p>
<b>*TYPE</b>	<p>Indicates that the comparison value contains the name of line type. This allows you to test the type of other lines on the page and identify the type of this line by reference to those lines.</p> <p><b>You must guard against circular definitions, i.e. one report line referring to another which refers back to the first report line.</b></p>

## Cmp (Comparison type)

Specifies the type of comparison to be applied.

Options are:

<b><u>*EQ</u></b>	The rule is true if the value at the specified position on the line to be tested is equal to the comparison value.
<b>*NE</b>	The rule is true if the value at the specified position on the line to be tested is not equal to the comparison value.
<b>*GT</b>	The rule is true if the value at the specified position on the line to be tested is greater than to the comparison value.
<b>*LT</b>	The rule is true if the value at the specified position on the line to be tested is less than to the comparison value.
<b>*GE</b>	The rule is true if the value at the specified position on the line to be tested is greater than or equal to the comparison value.
<b>*LE</b>	The rule is true if the value at the specified position on the line to be tested is less than or equal to the comparison value.
<b>*CT</b>	The rule is true if the value at the specified position on the line to be tested contains the comparison value.
<b>*NC</b>	The rule is true if the value at the specified position on the line to be tested does not contain the comparison value.

Only \*EQ and \*NE are valid if \*TYPE was specified for the comparison type.

### Value (comparison value)

Specifies the value against which the test occurs.

In addition to specifying simple constant strings on this field, one powerful way of testing for a report line is to use the built-in functions \$\$PATTERN or \$\$REGEX to test the text at the specified position using a pattern string or a regular expression.

A pattern string is a simplified form of regular expression which defines a sequence of characters or character types (e.g. numeric digits, alphabetic characters etc.) and makes it easy to check for things like strings, numeric values and dates at particular positions. See the discussion of the \$\$PATTERN CoolSpools function and the Worked Example below for further details.

A regular expression is a more complex but extremely powerful technique for testing for patterns in text. See the discussion of the \$\$REGEX CoolSpools function and the Worked Example below for further details.

Examples:

Rel	Line	Off	Pos	Cmp	Value	Description of test
*IF	*CURRENT	*NONE	1	*EQ	Region:	The test will evaluate to true if the line starts with the value

						"Region:"
*IF	*CURRENT	1	3	*NE	` `	The test will evaluate to true if the third character of the next line is not a blank.  Note that it is necessary to enclose the value in apostrophes if it consists of all blanks.
*IF	*LAST	-1	*TYPE	*EQ	END_OF_REPORT	The test will evaluate to true if the last but one line of the current page is of type END_OF_REPORT.
*IF	*CURRENT	*NONE	1	*EQ	\$\$PATTERN(####.##)	This test checks if the current line, starting at position 1, matches the pattern string specified. Patterns are explained below.
*IF	*CURRENT	*NONE	1	*EQ	\$\$REGEX(regular_expression_string)	This test checks if the current line, starting at position 1, matches the regular expression string specified. Regular expressions are explained below.

The following commands also operate on report lines. Parameters are only described where they differ significantly from those of the ADDRPTLIN command described above.

### **CHGRPTLIN – Change Report Line**

The CHGRPTLIN (Change Report Line) command modifies an existing report line. See ADDRPTLIN above for a discussion of the various parameters.

### **CPYRPTLIN – Copy Report Line**

The CPYRPTLIN (Copy Report Line) command copies a report line. Note that this command does **not** copy the report items associated with a line. Those must be copied individually later using CPYRPTITM.

### **RMVRPTLIN – Remove Report Line**

The RMVRPTLIN (Remove Report Line) command removes a report line from a report definition.



## **DSPRPTLIN – Display Report Line**

The DSPRPTLIN (Display Report Line) command displays details of a report line.

## **RNMRPTLIN – Rename Report Line**

The RNMRPTLIN (Rename Report Line) command renames a report line.

## **WRKRPTLIN – Work with Report Lines**

The WRKRPTLIN (Work with Report Lines) command lets you work with a list of report lines.

### ***REPORTNAME – Report definition name***

Specify the name of the report definition for which you wish to display list of report lines.

### ***SECTION – Report section name***

Specify the name of the section within the above report definition for which you wish to display list of report lines.

Options are:

<b>*ALL</b>	Display lines for all sections
<b>section_name</b>	Just display lines for the specified section.

## **ADDRPTITM – Add Report Item**

The ADDRPTITM (Add Report Item) command adds a report item to a report definition.

A report item describes a data item in a report that is a data field or text constant.

When CoolSpools is analyzing a spooled file in order to convert it to text and extract information from it, its first task is to try to match each line of the spooled file against a report line from the report definition it is using. Once it knows what report line describes a particular line of text in the spooled file, it can then chop that line of text up into the various report items it contains and use them.

### ***REPORTNAME – Report definition name***

Specify the name of the existing report definition to which you wish to add the report item.

Report definition names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

### ***ITEMNAME – Report item name***

Specify the name you wish to give to the report item.

Report item names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long. The name must be unique within the report definition.

### ***LINENAME – Report line name***

Specify the name of the existing report line of which this report item is part,

Report line names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

<b><u>*SELECT</u></b>	Select the line from a list of lines for the report definition.
<b>line_name</b>	Specify the line name

### ***SECTION – Report section name***

Specify the name of the existing report section of which this report item is part,

Options are:

<b><u>*LINE</u></b>	The report item is part of the section associated with the report line to which this report item belongs. This is the most common situation.
<b>*SELECT</b>	Select the section from a list of sections for the report definition.
<b>section_name</b>	<p>Specify the name of the existing section to which this data item belongs.</p> <p>Occasionally, a report line might comprise data items that need to be defined as belonging to different sections. For example, referring to the Customer Orders Report used in the Worked Example below, it would be possible to combine the region heading and state heading lines into one line and specify the region code, region name, state code and state name all on one line. If that were the case, then the region code and region data items name would be logically associated with the REGION section, while the state code and state name data items would belong to the STATE section. The one heading line would then start both the REGION and STATE sections, but dependent on different section rules (REGION would start on change of region code, while STATE would start on change of state code).</p>

### ***CHARPOS – Character position***

Specify the character position (column) on the line at which this data item starts.

Note that unlike the COLUMNPOS parameter of the CVTSPLXLS command, this character position always denotes the start (left-most) character of the data item, irrespective of its data type.

The character position is calculated using the CPI (Characters Per Inch) value specified when the report definition was created.

### ***CHARLEN – Character length***

Specify the number of characters over which this data item extends, starting with and including the character position specified on CHARPOS above.

When you are defining a numeric data item, remember to include all possible positions that can be occupied by the field, including any trailing minus sign.

### ***ITEMTYPE – Item type***

Specify the type of item being defined. This attribute does not play a significant part in processing at this stage but may be used by future features.

Options are:

<b><u>*VAR</u></b>	The item being defined in a variable (data field).
<b>*LABEL</b>	The item being defined is a piece of constant text that labels (describes) a variable (data field).
<b>*CONST</b>	The item being defined is a piece of constant text not associated with a variable (data field).

### ***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the report item.

Options are:

<b><u>*BLANK</u></b>	No text is specified.
<b>Text</b>	Specify the text ‘description’.

### ***DATATYPE – Data type***

Specify the data type of the item being defined. This information is used to determine the default type of Excel cells derived from this data item.

This parameter is ignored unless ITEMTYPE(\*VAR) was specified.

Options are:

<b><u>*ALPHA</u></b>	An alphanumeric field
<b>*NUMERIC</b>	A numeric field
<b>*DATE</b>	A date field.

### ***NULLDTAOPT – Blank data option***

Where the value of a data item changes from one occurrence of the line type to the next, this option determines whether or not that change is ignored if the new data value is blanks and the previous data value was non-blanks.

This option can be helpful where (for example in Query/400 output) the value for a column of data is output once only and blanks are output on subsequent lines. In those circumstances you may wish to specify \*IGNORE for this attribute so that the

first non-blank value is retained and not overwritten by blank values from subsequent lines.

Options are:

<b><u>*NONE</u></b>	Changes of value for this data item are always taken into account, even where blank data is replacing non-blank data.
<b>*IGNORE</b>	Changes of value for this data item are ignored where blank data would overwrite a previous non-blank data value.

### ***DATFMT – Date format***

Defines the format of the date this data item describes. This parameter is ignored unless DATATYPE(\*DATE) was specified.

Options are:

<b><u>*RPTDFN</u></b>	The default date format for the report definition that was defined when you created it.
<b><u>*JOB</u></b>	The DATFMT job attribute is used to determine the format of dates in the spooled file.
<b>*SYSVAL</b>	The QDATFMT system value is used to determine the format of dates in the spooled file.
<b>*DMY</b>	Day Month Year format is assumed.
<b>*MDY</b>	Month Day Year format is assumed.
<b>*YMD</b>	Year Month Day format is assumed.

### ***DATSEP - Spooled file date separator***

Defines the separator used for dates this data item describes. This parameter is ignored unless DATATYPE(\*DATE) was specified.

Options are:

<b><u>*RPTDFN</u></b>	The default date separator for the report definition that was defined when you created it.
<b>*JOB</b>	The DATSEP job attribute is used to determine the format of dates in the spooled file.
<b>*SYSVAL</b>	The QDATSEP system value is used to determine the format of dates in the spooled file.
<b>*NONE</b>	No date separator is used.

The following commands also operate on report items. Parameters are only described where they differ significantly from those of the ADDRPTITM command described above.

### **CHGRPTITM – Change Report Item**

The CHGRPTITM (Change Report Item) command modifies an existing report item. See ADDRPTITM above for a discussion of the various parameters.

## **CPYRPTITM – Copy Report Item**

The CPYRPTITM (Copy Report Item) command copies a report item.

## **RMVRPTITM – Remove Report Item**

The RMVRPTITM (Remove Report Item) command removes a report item from a report definition.

## **DSPRPTITM – Display Report Item**

The DSPRPTITM (Display Report Item) command displays details of a report item.

## **RNMRPTITM – Rename Report Item**

The RNMRPTITM (Rename Report Item) command renames a report item.

## **WRKRPTITM – Work with Report Items**

The WRKRPTITM (Work with Report Items) command lets you work with a list of report items.

### ***REPORTNAME – Report definition name***

Specify the name of the report definition for which you wish to display list of report items.

### ***LINENAME – Report line name***

Specify the name of the report line within the above report definition for which you wish to display list of report items.

Options are:

<b>*ALL</b>	Display items for all lines.
<b>line_name</b>	Just display items for the specified line.

## **ADDRPTSCT – Add Report Section**

The ADDRPTSCT (Add Report Section) command adds a report section to a report definition.

When report lines and the items they comprise have been defined, you then need to define the section structure of the report.

Most reports have some kind of section structure, and it is important that CoolSpools knows about this structure in order to be able to create meaningful output from your report, for example XML documents where elements are nested correctly.

For example, in the case of the demo Customer Order Report DM\_ORDRPT1 (see Worked Example below), the report lists orders for a given date range by customer, within US state, by region of the USA. There are therefore 3 sections that CoolSpools needs to know about and these form the following section hierarchy:



The region section of the report comprises one or more states and each state shown in the report comprises one or more customers.

If, for example, you want CoolSpools to be able to build an XML document from the report which takes the following form:

```

<region>
  <state>
    <customer/>
    <customer/>
    <customer/>
  </state>
  <state>
    <customer/>
    <customer/>
  </state>
</region>

```

...

etc.

then CoolSpools needs to know how sections relate to one another and when each section starts and ends.

Sections define the relationship between the various lines in a report. In general, where two lines in the report are related to one another, and may need to be handled as a unit, they should be defined as being part of the same section, or as being part of different sections that are themselves related in terms of a section hierarchy. For example, if the order information for each order in the Customer Order Report were to cover two lines, those lines should be defined as making up an order section. CoolSpools can then treat the two lines as a single entity and correctly process the data for a single order from both lines together (e.g. outputting that data to a single XML element or Excel row).

## ***REPORTNAME – Report definition name***

Specify the name of the existing report definition to which you wish to add the report section.

Report definition names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

## **SECTION – Report section name**

Specify the name you wish to give to the report section.

Report section names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long. The name must be unique within the report definition.

## **PARENT– Parent section name**

Specify the name of the existing report section of which this report section is a child section.

As described above, sections can form part of a section hierarchy where one section is the “parent” of one or more other sections (its “children”).

Options are:

<b><u>*NONE</u></b>	The section is either not part of a section hierarchy or is the top-level section and has no parent, only children.
<b>*SELECT</b>	Select the section from a list of sections in the report definition.
<b>section_name</b>	Specify the name of the existing section of which this section is a child.

## **STARTLINE - Section start line name**

Specify the name of the report line, which, when it is encountered during the analysis of a spooled file, indicates the start of a new instance of this report section

Note that rules can be specified on STARTRULES below to further qualify the conditions under which a new instance of this report section starts.

For example, in the Customer Orders report used in the Worked Example below, the REGION section starts when a REGION\_HEADER line is encountered, but only if the REGION\_CODE field has changed from the previous REGION\_HEADER line.

Options are:

<b><u>*SELECT</u></b>	Select the line from a list of report lines in the report definition.
<b>*REPORT</b>	The section starts at the beginning of the report.
<b>line_name</b>	Specify the name of the existing report line which starts the section.

## **STARTRULES – Section start rules**

Defines the conditions under which a new instance of this report section starts when a spooled file is being analyzed.

## **Relationship**

Specifies the logical relationship between one test and the next

Options are:

<b><u>*IF</u></b>	Mandatory for the first test and allowed only on the first test.
<b>*AND</b>	Indicates that this test is part of the same logical group as the previous line. The rule will be true only if the result of all tests since and including the previous test starting with *IF or *OR test are true.
<b>*OR</b>	Indicates that this test starts a new OR group. The rule will be true if the combined result of this group or any other group starting with *IF or OR is true.

## Report item name

Specify a report item to be tested.

Options are:

<b><u>*SELECT</u></b>	Select the item from a list of report items in the report definition.
<b>item_name</b>	Specify the name of the item to test.

## Comparison

Specifies the type of comparison to be applied.

Options are:

<b><u>*EQ</u></b>	The rule is true if the value at the specified position on the line to be tested is equal to the comparison value.
<b>*NE</b>	The rule is true if the value at the specified position on the line to be tested is not equal to the comparison value.
<b>*GT</b>	The rule is true if the value at the specified position on the line to be tested is greater than to the comparison value.
<b>*LT</b>	The rule is true if the value at the specified position on the line to be tested is less than to the comparison value.
<b>*GE</b>	The rule is true if the value at the specified position on the line to be tested is greater than or equal to the comparison value.
<b>*LE</b>	The rule is true if the value at the specified position on the line to be tested is less than or equal to the comparison value.

## Value

Specifies the value against which the test occurs.

You can also use the special value **\*PRV** to denote the previous value of the item on the previous occurrence of the report line. One frequent use of this option is to test for changes to report item values. For example, in the Customer Orders Report used



in the Worked Example below, the REGION section has a start rule that states that a new instance of the REGION section begins

**\*IF REGION\_CODE \*NE \*PRV**

that is, when the region code changes.

## ***ENDLINE - Section end line name***

Specify the name of the report line, which, when it is encountered during the analysis of a spooled file, indicates the end of an instance of this report section.

Note that rules can be specified on ENDRULES below to further qualify the conditions under which an instance of this report section ends.

For example, in the Customer Orders report used in the Worked Example below, the REGION section always ends when a REGION\_TOTALS line is encountered.

However, in other reports, it could be necessary to apply additional tests to determine if a section should end or not.

There are two elements to this parameter, the default for which is the single value:

### **\*STARTLINE**

An instance of this section ends when an occurrence of the line specified on the STARTLINE parameter is encountered and when the rules specified on STARTRULES above, if any, are true.

This option is intended to handle the situation where there is no convenient line which marks the end of a section and the only way to tell when an instance of a section has ended is the occurrence of the line which starts a new instance of that section.

## **Line name**

Options are:

### **\*SELECT**

Select the line from a list of report lines in the report definition.

### **\*REPORT**

The section ends at the end of the report.

### **line\_name**

Specify the name of the existing report line which ends the section.

## **Part of this section**

Options are:

### **\*YES**

The end line specified above is part of the instance of the section it is ending.

### **\*NO**

The end line specified above is not part of the instance of the section it is ending (i.e. it is part of a different section or a new instance of this section, but serves as a convenient marker for the end of the section).

## ***ENDRULES – Section end rules***

Defines the conditions under which an instance of this report section ends when a spooled file is being analyzed.

Options are the same as for STARTRULES above but you can also use the single values:

<b><u>*STARTRULES</u></b>	The same rules apply when testing the end of a section as when testing for the start.
<b>*NONE</b>	No end rules are needed. The occurrence of the ENDLINE specified above is sufficient to mark the end of an instance of this section.

## ***LINENAMES – Lines included in section***

Specifies the report lines this section comprises.

Note that the start and end lines specified above are not automatically included as they are not necessarily part of the section. They must be listed here if required to be included.

<b><u>*NONE</u></b>	No lines are included.
<b>*SELECT</b>	Select one or more lines from a list of lines in the report definition.
<b>line_name</b>	Specify between 1 and 100 line names.

## **CHGRPTSCT – Change Report Section**

The CHGRPTSCT (Change Report Section) command modifies an existing report section.

See ADDRPTSCT above for a discussion of the various parameters.

## **CPYRPTSCT – Copy Report Section**

The CPYRPTSCT (Copy Report Section) command copies a report section.

## **RMVRPTSCT – Remove Report Section**

The RMVRPTSCT (Remove Report Section) command removes a report section from a report definition.

## **DSPRPTSCT – Display Report Section**

The DSPRPTSCT (Display Report Section) command displays details of a report section.

## **RNMRPTSCT – Rename Report Section**

The RNMRPTSCT (Rename Report Section) command renames a report section.

## **WRKRPTSCT – Work with Report Sections**

The WRKRPTSCT (Work with Report Sections) command lets you work with a list of report sections.

### ***REPORTNAME – Report definition name***

Specify the name of the report definition for which you wish to display list of report sections.

### ***PARENT– Parent section name***

Specify the name of the report section within the above report definition for which you wish to display a list of child sections.

Options are:

<b>*ALL</b>	Display all sections irrespective of the parent.
<b>parent_name</b>	Just display items for the specified parent section.

## **Commands related to Report Maps**

### **CRRPTXL – Create Report-to-Excel Map**

The CRRPTXL (Create Report-to-Excel Map) command creates a report-to-Excel map definition describing the content and structure of an Excel file to be created from a spooled file.

Once you have created your report-to-Excel map you need to specify the different row groups and cells it comprises. See the ADDRPTXLR (Add Report-to-Excel Map Row Group) and ADDRPTXLC (Add Report-to-Excel Map Cell) commands for details of how to do that.

#### **MAPNAME – Report-to-Excel map name**

Specify the name you wish to give to the report-to-Excel map.

Report map names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

#### **REPORTNAME – Report definition name**

Specify the name of an existing report definition that will define the input to the conversion process. The report map defines the output from the conversion process.

You must be authorized to use the report definition.

#### **DFTUSEAUT - Default use authority**

The default authority to use this report map.

Individual user authorities to the map can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a report-to-Excel map is

ARIADNE\_XLS\_MAP\_nnnnnnnnnn\_USE

where nnnnnnnnn is the internal map identifier, which is displayed by DSPRPTXL.

Options are:

##### **\*ALLOWED**

By default, users other than the user creating the map are permitted to use it when converting a spooled file to Excel.

##### **\*DENIED**

By default, users other than the user creating the map are not permitted to use it when converting a spooled file to Excel.

#### **DFTCHGAUT - Default change authority**

The default authority to change or delete this report map.

Individual user authorities to the map can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a report-to-Excel map is

ARIADNE\_XLS\_MAP\_nnnnnnnnnn\_CHG

where nnnnnnnnn is the internal map identifier, which is displayed by DSPRPTXL.

Options are:

- |                       |  |
|-----------------------|--|
| <b><u>*DENIED</u></b> | By default, users other than the user creating the map are not permitted to change, delete or manage it. |
| <b>*ALLOWED</b>       | By default, users other than the user creating the map are permitted to change, delete or manage it.     |

### ***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the report map.

Options are:

- |                      |                                 |
|----------------------|---------------------------------|
| <b><u>*BLANK</u></b> | No text is specified.           |
| <b>Text</b>          | Specify the text ‘description’. |

### ***GRPSEQOPT - Row group sequence option***

Determines the order in which row groups are output.

Options are:

- |                    |  |
|--------------------|--|
| <b><u>*MAP</u></b> | The order of rows in the Excel file is determined by the way row groups are organized in the report-to-Excel map.<br><br>Rows will be written to the Excel worksheet based on the hierarchy of row groups in the report-to-Excel map, taking account of parent-child relationships between row groups and the sequence number of child row groups within the parent row group. |
| <b>*SPLF</b>       | The order of rows in the Excel file is determined by the order of data in the spooled file.  |

The following commands also operate on report-to-Excel maps. Parameters are only described where they differ significantly from those of the CRTRPTXL command described above.

### **CHGRPTXL – Change Report-to-Excel map**

The CHGRPTXL (Change Report-to-Excel map) command modifies an existing Report-to-Excel map.

See CRTRPTXL above for a discussion of the various parameters.

### **CPYRPTXL – Copy Report-to-Excel map**

The CPYRPTXL (Copy Report-to-Excel map) command copies a Report-to-Excel map and its associated row groups and cells.

## ***FROMMAP – From Report-Excel map name***

Specify the name of the Report-to-Excel map you wish to copy.

## ***TOMAP – To Report-Excel map***

Specify the name of the Report-to-Excel map you wish to create, based on the Report-to-Excel map being copied.

The remaining parameters allow attributes to be modified while the map is being copied. See CRTRPTXL above for a discussion of these parameters.

## ***DLTRPTXL – Delete Report-to-Excel map***

The DLTRPTXL (Delete Report-to-Excel map) command deletes a Report-to-Excel map.

## ***DSPRPTXL – Display Report-to-Excel map***

The DSPRPTXL (Display Report-to-Excel map) command displays details of a report definition.

## ***RNMRPTXL – Rename Report-to-Excel map***

The RNMRPTXL (Rename Report-to-Excel map) command renames a Report-to-Excel map.

## ***MAPNAME – Report Report-to-Excel map***

Specify the name of the Report-to-Excel map you wish to rename.

## ***NEWMAP – New Report-to-Excel map***

Specify the new name for the Report-to-Excel map.

## ***RTVRPTXL – Retrieve Report-to-Excel map***

The RTVRPTXL (Retrieve Report-to-Excel map) command retrieves CL source for creating a Report-to-Excel map and all its associated row groups and cells. This provides a convenient way of saving and distributing a Report-to-Excel map to other systems. The source that is retrieved can be easily converted to a program which can be run to create the Report-to-Excel map.

## ***MAPNAME – Report-to-Excel map***

Specify the name of the Report-to-Excel map for which you wish to retrieve source.

## ***SRCFILE – Source file***

Specify the qualified name of the source file into which the source should be retrieved. The file and library must already exist.

## ***SRCMBR – Source member***

Specify the name of the source member into which the source should be retrieved. If the member does not already exist, it will be created.

## ***MBROPT – Source member***

Whether an existing member is replaced or appended to.

Options are:

<b><u>*REPLACE</u></b>	If the member already exists, it will be replaced.
<b>*ADD</b>	If the member already exists, the retrieved source will be appended to it.

## **WRKRPTXL – Work with Report-to-Excel maps**

The WRKRPTXL (Work with Report-to-Excel maps) command displays a list of existing Report-to-Excel maps and lets you operate on them or create new maps.

The following commands allow you to define the content and structure of a report by adding report lines, report data items and report sections to a report definition.

These commands can also be invoked by taking options from the WRKRPTDFN screen.

See the DSNRPTDFN (Design Report Definition) command below for details of an alternative method of specifying a report definition interactively using a sample copy of the spooled file.

## **ADDRPTXLR – Add Report-to-Excel Map Row Group**

The ADDRPTXLR (Add Report-to-Excel Map Row Group) command adds a row group to a Report-to-Excel map.

A row group is a set of one or more related rows that are output to an Excel worksheet as a group when CoolSpools is converting a spooled file to Excel using a Report-to-Excel map. In the Customer Orders Report Worked Example below, there are different row groups for:

- report headings
- region headings
- state headings
- customer headings
- order detail
- customer totals
- state totals
- region totals
- report totals

Each row group has its own set of cells defined. Those cells specify the content of the row group, in terms of variables derived from report items or static constant text.

Each row group can have different styling applied to other row groups.

It is important to grasp the concept that a row *group* can encompass more than one row written to the Excel worksheet. A row group can define any number of related rows that are always written together. Thus, one line in your report can generate multiple rows in your Excel file.

If you wish to have empty rows, in your Excel file, this can easily be achieved by defining a row of empty, merged cells.

The parameters of the ADDRPTXLR command are as follows.

### ***MAPNAME – Report-to-Excel map name***

Specify the name of the existing Report-to-Excel map to which you wish to add the row group.

Report-to-Excel map names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

### ***ROWGRPNAME – Row group name***

Specify the name you wish to give to the row group.

Row group names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

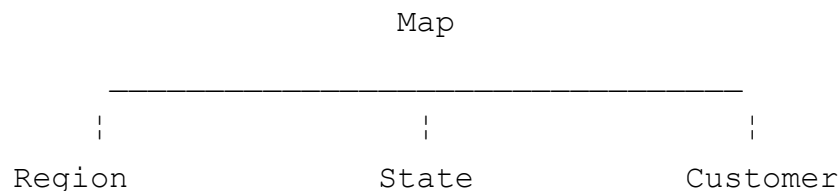
### ***PARENT– Parent row group name***

Specify the name of the existing row group which is the parent of this row group.

Like XML elements, Excel row groups can be nested inside one another in a parent-child relationship. It is important to define these relationships correctly in order to obtain the right results. The Excel file is built by using the Excel map as a template to generate a tree structure from the report data and the structure of the map you define is crucial in determining the structure of the Excel files you create.

Specifically, rows in the Excel file are output in the sequence of the corresponding row groups at a particular level in the hierarchy of row groups in the Excel map.

For example, if you were to define an Excel map for the Customer Orders Report that had row groups corresponding to the region, state and customer sections at the same level, like this:



the resultant Excel file would have rows for all of regions first, then rows for all states, then rows for all customers, thus:

Region = NORTHEAST

Region = SOUTH

Region = WEST

...



State = MASSACHUSETTS

State = NEW JERSEY

State = NEW YORK

...

Customer = TRULY TASTY TURNIPS

Customer = PRAIRIE TREE AND SEED

Customer = EVERGREEN & HARWOOD SEEDS

...

whereas defining the correct hierarchy thus:



would give properly nested results, thus:

Region = NORTHEAST

State = MASSACHUSETTS

Customer = TRULY TASTY TURNIPS

...

State = NEW JERSEY

Customer = PRAIRIE TREE AND SEED

...

State = NEW YORK

Customer = EVERGREEN & HARWOOD SEEDS

...

Region = SOUTH

State = FLORIDA

Customer = SARAH'S SAFARI FRUITS

Customer = ABUNDANT FRUIT & FLOWERS

...

A Report-to-Excel map does not require a named root row group in the same way an XML map requires a named root element. The Excel workbook itself operates as an implied root row group, and you can define more than one row group with PARENT(\*NONE), indicating that they are all children of the implied root.

Options are:

<b><u>*NONE</u></b>	The row group has no parent row group, i.e. it is an immediate child of the implied root row group, which is the Excel file itself.
<b>*SELECT</b>	Select the parent from a list of row groups in the Report-to-Excel map.
<b>row_group</b>	Specify the name of the parent row group.

### ***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the row group.

Options are:

<b><u>*BLANK</u></b>	No text is specified.
<b>Text</b>	Specify the text ‘description’.

### ***NEWGRPOPT - New row group option***

This option determines the logic which controls the creation of new row groups of this kind.

Options are:

<b><u>*SECTION</u></b>	This option ties the row group to a section in the report definition on which the Report-to-Excel map is dependent. A new row group of this kind will be created every time a new instance of the section of the type specified on the SECTION parameter is encountered in the spooled file being converted.
<b>*LINE</b>	This option ties the row group to a line in the report definition on which the Report-to-Excel map is dependent. A new row group of this kind will be created every time a new line of the type specified on the LINENAME parameter is encountered in the spooled file being converted.
<b>*NEVER</b>	A new row group is never created. A single row group will be created within each occurrence of the parent row group. If there is no parent row group, then there will be a single occurrence of this row group in the file. This option is normally appropriate for top-level row groups such as page headings, or row groups which need to occur once in relation to some other row group, such as column headings.

### ***SECTION – Report section name***

Specify the name of the section to which this row group is linked, when NEWGRPTOPT(\*SECTION) is used.

Options are:

<b><u>*NONE</u></b>	This row group is not linked to a section. Invalid if NEWGRPTOPT(*SECTION) was specified.
<b>*SELECT</b>	Select the section from a list of sections in the report definition.
<b>section_name</b>	Specify the name of the linked section.

### **LINENAME – Report line name**

Specify the name of the report line to which this row group is linked, when NEWGRPTOPT(\*LINE) is used.

Options are:

<b><u>*NONE</u></b>	This row group is not linked to a report line. Invalid if NEWGRPTOPT(*LINE) was specified.
<b>*SELECT</b>	Select the line from a list of lines in the report definition.
<b>line_name</b>	Specify the name of the linked line.

The following commands also operate on row groups. Parameters are only described where they differ significantly from those of the ADDRPTXLR command described above.

### **CHGRPTXLR – Change Report-to-Excel Map Row Group**

The CHGRPTXLR (Change Report-to-Excel Map Row Group) command modifies an existing Report-to-Excel Map Row Group.

See ADDRPTXLR above for a discussion of the various parameters.

### **CPYRPTXLR – Copy Report-to-Excel Map Row Group**

The CPYRPTXLR (Copy Report-to-Excel Map Row Group) command copies a Report-to-Excel Map row group and its related cells.

### **RMVRPTXLR – Remove Report-to-Excel Map Row Group**

The RMVRPTXLR (Remove Report-to-Excel Map Row Group) command removes a Report-to-Excel Map Row Group from a report definition.

### **DSPRPTXLR – Display Report-to-Excel Map Row Group**

The DSPRPTXLR (Display Report-to-Excel Map Row Group) command displays details of a Report-to-Excel Map Row Group.

### **RNMRPTXLR – Rename Report-to-Excel Map Row Group**

The RNMRPTXLR (Rename Report-to-Excel Map Row Group) command renames a Report-to-Excel Map Row Group.

## **WRKRPTXLR – Work with Report-to-Excel Map Row Groups**

The WRKRPTXLR (Work with Report-to-Excel Map Row Groups) command lets you work with a list of Report-to-Excel Map Row Groups.

### ***MAPNAME – Report-to-Excel map name***

Specify the name of the Report-to-Excel map for which you wish to display list of row groups.

### ***PARENT– Parent row group name***

Specify the name of the row group within the above Report-to-Excel map for which you wish to display list of child row groups.

Options are:

- \*ALL** Display all row groups in the map.
- row\_group\_name** Just display row groups that are children of the specified row group.

## **ADDRPTXLC – Add Report-to-Excel Map Cell**

The ADDRPTXLC (Add Report-to-Excel Map Cell command adds a cell to a Report-to-Excel map.

A Report-to-Excel map cell determines the content of a particular cell (row/column intersection) in the Excel file that CoolSpools creates from your spooled file using your Report-to-Excel map.

Each cell belongs to a particular row group. This allows you to define different cell content for different row groups. The cell is identified by means of:

- the row group to which it belongs
- the row number within that row group on which it appears
- the Excel column reference identifying the vertical column on which it appears

### ***MAPNAME – Report-to-Excel map name***

Specify the name of the existing Report-to-Excel map to which you wish to add the cell.

Report-to-Excel map names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

### ***ROWGRPNAME – Row group name***

Specify the name of the existing Report-to-Excel row group to which you wish to add the cell.

Row group names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

## **ROWNR-Row number**

Specify the number of the row within the row group on which this cell should appear.

Note that this is not the row number in the Excel worksheet itself. It indicates the relative row number within the row group. For example, ROWNR(1) indicates the first row in the row group, ROWNR(2) the second etc. If the row group is output to the Excel worksheet twice, starting at Excel row 101 and row number 201, then ROWNR(1) will correspond to Excel row numbers 101 and 201, ROWNR(2) to Excel row numbers 102 and 202 etc.

## **COLUMN-Column letter**

Specify the Excel column reference identifying the vertical column in the spreadsheet at which the cell will appear, e.g. A = first column, Z= 26<sup>th</sup> column, AA=27<sup>th</sup> column etc.

## **CONTENT-Cell content**

Determines what type of content the cell should have.

Options are:

<b><u>*ITEM</u></b>	The cell content is derived from the value of a report item defined in the report definition describing the spooled file that is used as input to the conversion. You specify the name of the report item on the CELLITEM parameter below. The current value of the item is written to the cell.
<b>*TEXT</b>	The cell content will be a piece of static, constant text which you will supply on the CELLTEXT parameter below. This option is useful for headings, cell labels etc.
<b>Text</b>	Specify the text 'description'.

## **CELLITEM-Report item**

The report item from which the cell content will be derived when CONTENT(\*ITEM) is specified.

### **Name**

The name of the report item.

Options are:

<b>*SELECT</b>	Select the item from a list of report items in the report definition.
<b>item_name</b>	Specify the name of the existing report item.

### **Data type**

The type of cell to create.

Options are:

<b>*ITEM</b>	The data type is determined by the data type of the report item.
<b>*ALPHA</b>	The cell will be a text label, even if the report item from which it is derived is a numeric or date item.
<b>*NUMERIC</b>	The cell will be a number, even if the report item from which it is derived is an alphanumeric or date item. Note that non-numeric data in the item may result in garbage being written to the cell if this option is taken.
<b>*DATE</b>	The cell will be a date, even if the report item from which it is derived is an alphanumeric or numeric item. Note that non-date data in the item may result in garbage being written to the cell if this option is taken.

### **CELLTEXT – Cell text**

The constant text which will be written to the cell content when CONTENT(\*TEXT) is specified.

Note that the text can contain CoolSpools variables. For example:

**CELLTEXT('<:CURMONTH:>/<:CURDAY:>/<:CURYEAR:>')**

would cause the current date in MM/DD/YYYY format to be output.

### **MRGCELLS – Merge to cell**

Specify the row and column to which this cell will be merged.

The default is the single value **\*NONE**, indicating that the cell is not merged with any neighboring cells and simply occupies the intersection point of the row and column identified by ROWNBR and COLUMN above.

Alternatively, specify a row number within the row group and column letter. This will then specify a block of merged cells extending from the row number and column identified by ROWNBR and COLUMN above to the row number and column specified below.

#### **Merge to row number**

The number within the row group of the row to which the block of merged cells extends.

Options are:

<b>*ROWNBR</b>	The block of merged cells ends on the same row identified by the ROWNBR parameter above. In other words, the block is all on one row.
<b>row_number</b>	Specify the number of the row in the row group to which the block of merged cells will extend.

#### **Merge to column letter**

Options are:

**\*COLUMN**

This block of merged cells ends on the same column identified by the COLUMN parameter above.

**column\_letter**

Specify the Excel column letter/reference of the column to which the block of merged cells will extend.

The following commands also operate on Report-to-Excel map cells. Parameters are only described where they differ significantly from those of the ADDRPTXLC command described above.

**CHGRPTXL – Change Report-to-Excel Map Cell**

The CHGRPTXLC (Change Report-to-Excel Map Cell) command modifies an existing Report-to-Excel Map Cell.

See ADDRPTXLC above for a discussion of the various parameters.

**CPYRPTXLC – Copy Report-to-Excel Map Cell**

The CPYRPTXLC (Copy Report-to-Excel Map Cell) command copies a Report-to-Excel Map cell.

**RMVRPTXLC – Remove Report-to-Excel Map Cell**

The RMVRPTXLC (Remove Report-to-Excel Map Cell) command removes a Report-to-Excel Map Cell from a report definition.

**DSPRPTXLC – Display Report-to-Excel Map Cell**

The DSPRPTXLC (Display Report-to-Excel Map Cell) command displays details of a Report-to-Excel Map Cell.

**RNMRPTXLC – Rename Report-to-Excel Map Cell**

The RNMRPTXLC (Rename Report-to-Excel Map Cell) command renames a Report-to-Excel Map Cell.

**WRKRPTXLC – Work with Report-to-Excel Map Cells**

The WRKRPTXLC (Work with Report-to-Excel Map Cells) command lets you work with a list of Report-to-Excel Map Cells.

**MAPNAME – Report-to-Excel map name**

Specify the name of the Report-to-Excel map in which the row group named below exists.

**ROWGRPNAME – Row group name**

Specify the name of the row group within the above Report-to-Excel map for which you wish to display list of cells.

## **ADDRPTXMLE (Add Report-to-XML Map Element)** **command**

The ADDRPTXMLE (Add Report-to-XML Map Element) command adds an element to a Report-to-XML map. A Report-to-XML element controls the creation of nodes within an XML document.

### ***MAPNAME – Report-to-XML Map Name***

Specify the name of the existing Report-to-XML map to which you wish to add the row group.

### ***ELEMENT–Element name***

Specify the name of the XML element you wish to add.

Report-to-XML element names can be up to 50 characters in length and, like all things XML, are case-sensitive. They must conform to the rules for XML names.

### ***PARENT– Parent element name***

Specify the name of the parent element, if any. Use of this parameter allows the creating of a nested tree of XML elements.

One and only one element in a Report-to-XML map must be designated the root by specifying PARENT(\*NONE) when it is created. All other elements must be descendants of this element.

Options are:

<b><u>*NONE</u></b>	The element will be the document root element. Only one root element can exist for each Report-to-XML map.
<b>*SELECT</b>	Select the parent element from a list of existing elements in the Report-to-XML map.
<b>Parent_name</b>	Specify the name of the parent element, which could be the root or a descendant of the root.

### ***SEQNBR – Sequence number***

A number determining the order in which elements are output within their parent elements. Elements are output in the order of their sequence number within their parent element.

Options for the library name are:

<b><u>*NEXT</u></b>	The next highest available sequence number in the parent element is assigned.
<b>seq_number</b>	Specify the sequence number.

### ***ITEMNAME- Report item***

Specify the name of a report item from which the value of the text node for this element will be derived.

Single options are:



**\*NONE**

The element has no text node.

**\*SELECT**

You will be prompted to select the report item from a list of report items in the report definition. The data type will be taken from the data type of the selected item.

**Name**

Specify the name of the report item.

**Data type**

How to handle the data.

Options are:

**\*ITEM**

The data type is determined by the data type of the report item.

**\*ALPHA**

The data will be treated as alphanumeric, even if the report item from which it is derived is a numeric or date item.

**\*NUMERIC**

The data will be treated as numeric, even if the report item from which it is derived is an alphanumeric or date item. Note that non-numeric data in the item may result in garbage being written if this option is taken.

**\*DATE**

The data will be treated as a date, even if the report item from which it is derived is an alphanumeric or numeric item. Note that non-date data in the item may result in garbage being written if this option is taken.

***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the element.

Options are:

**\*BLANK**

No text is specified.

**Text**

Specify the text ‘description’.

***NEWELMOPT - New element option***

This option determines the logic which controls the creation of new elements of this kind.

Options are:

**\*SECTION**

This option ties the element to a section in the report definition on which the Report-to-Excel map is dependent. A new element of this kind will be created every time a new instance of the section of the type specified on the SECTION parameter is encountered in the spooled file being converted.

<b>*LINE</b>	This option ties the element to a line in the report definition on which the Report-to-Excel map is dependent. A new element of this kind will be created every time a new line of the type specified on the LINENAME parameter is encountered in the spooled file being converted.
<b>*NEVER</b>	A new element is never created. A single element will be created within each occurrence of the parent element. If there is no parent element, then there will be a single occurrence of this row group in the file. This option is normally appropriate for top-level elements such as page headings, or elements which need to occur once in relation to some other element, such as column headings.

### ***SECTION – Report section name***

Specify the name of the section to which this element is linked, when NEWGRPTOPT(\*SECTION) is used.

Options are:

<b><u>*NONE</u></b>	This element is not linked to a section. Invalid if NEWGRPTOPT(*SECTION) was specified.
<b>*SELECT</b>	Select the section from a list of sections in the report definition.
<b>section_name</b>	Specify the name of the linked section.

### ***LINENAME – Report line name***

Specify the name of the report line to which this element is linked, when NEWGRPTOPT(\*LINE) is used.

Options are:

<b><u>*NONE</u></b>	This element is not linked to a report line. Invalid if NEWGRPTOPT(*LINE) was specified.
<b>*SELECT</b>	Select the line from a list of line in the report definition.
<b>line_name</b>	Specify the name of the linked line.

The following commands also operate on elements. Parameters are only described where they differ significantly from those of the ADDRPTXMLE command described above.

### **CHGRPTXMLE – Change Report-to-Excel Map Element**

The CHGRPTXMLE (Change Report-to-Excel Map Element) command modifies an existing Report-to-Excel Map Element.

See ADDRPTXMLE above for a discussion of the various parameters.

## **CPYRPTXMLE – Copy Report-to-Excel Map Element**

The CPYRPTXMLE (Copy Report-to-Excel Map Element) command copies a Report-to-Excel Map element and its related attributes.

## **RMVRPTXMLE – Remove Report-to-Excel Map Element**

The RMVRPTXMLE (Remove Report-to-Excel Map Element) command removes a Report-to-Excel Map Element from a report definition.

## **DSPRPTXMLE – Display Report-to-Excel Map Element**

The DSPRPTXMLE (Display Report-to-Excel Map Element) command displays details of a Report-to-Excel Map Element.

## **RNMRPTXMLE – Rename Report-to-Excel Map Element**

The RNMRPTXMLE (Rename Report-to-Excel Map Element) command renames a Report-to-Excel Map Element.

## **WRKRPTXMLE – Work with Report-to-Excel Map Elements**

The WRKRPTXMLE (Work with Report-to-Excel Map Elements) command lets you work with a list of Report-to-Excel Map Elements.

### ***MAPNAME – Report-to-Excel map name***

Specify the name of the Report-to-Excel map for which you wish to display list of elements.

### ***PARENT– Parent element name***

Specify the name of the element within the above Report-to-Excel map for which you wish to display list of child elements.

Options are:

<b>*ALL</b>	Display all elements in the map.
<b>element_name</b>	Just display elements s that are children of the specified element.

## **ADDRPTXMLA (Add Report-to-XML Map Attribute) command**

The ADDRPTXMLA (Add Report-to-XML Map Attribute) command adds an attribute to an element in a Report-to-XML map.

### ***MAPNAME –Report-to-XML Map Name***

Specify the name of the existing Report-to-XML map to which you wish to add the row group.

### ***ELEMENT–Element name***

Specify the name of the existing XML element to which you wish to add an attribute.

Report-to-XML element names can be up to 50 characters in length and, like all things XML, are case-sensitive. They must conform to the rules for XML names.

### ***ATTRIBUTE–Attribute name***

Specify the name of the XML attribute you wish to add to the element.

Report-to-XML attribute names can be up to 50 characters in length and, like all things XML, are case-sensitive. They must conform to the rules for XML names.

### ***SEQNBR – Sequence number***

A number determining the order in which attributes are output on their associated elements. Attributes are output in the order of their sequence number specified.

Options for the library name are:

#### **\*NEXT**

The next highest available sequence number in the element is assigned.

#### **seq\_number**

Specify the sequence number.

### ***ITEMNAME- Report item***

Specify the name of a report item from which the value of the attribute will be derived.

Single options are:

#### **\*SELECT**

You will be prompted to select the report item from a list of report items in the report definition. The data type will be taken from the data type of the selected item.

### ***Name***

Specify the name of the report item.

### ***Data type***

How to handle the data.

Options are:

#### **\*ITEM**

The data type is determined by the data type of the report item.

<b>*ALPHA</b>	The data will be treated as alphanumeric, even if the report item from which it is derived is a numeric or date item.
<b>*NUMERIC</b>	The data will be treated as numeric, even if the report item from which it is derived is an alphanumeric or date item. Note that non-numeric data in the item may result in garbage being written if this option is taken.
<b>*DATE</b>	The data will be treated as a date, even if the report item from which it is derived is an alphanumeric or numeric item. Note that non-date data in the item may result in garbage being written if this option is taken.

### ***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the element.

Options are:

<b><u>*BLANK</u></b>	No text is specified.
<b>Text</b>	Specify the text ‘description’.

The following commands also operate on attributes. Parameters are only described where they differ significantly from those of the ADDRPTXMLA command described above.

### **CHGRPTXMLA – Change Report-to-Excel Map Attribute**

The CHGRPTXMLA (Change Report-to-Excel Map Attribute) command modifies an existing Report-to-Excel Map Attribute.

See ADDRPTXMLA above for a discussion of the various parameters.

### **CPYRPTXMLA – Copy Report-to-Excel Map Attribute**

The CPYRPTXMLA (Copy Report-to-Excel Map Attribute) command copies a Report-to-Excel Map element and its related attributes.

### **RMVRPTXMLA – Remove Report-to-Excel Map Attribute**

The RMVRPTXMLA (Remove Report-to-Excel Map Attribute) command removes a Report-to-Excel Map Attribute from a report definition.

### **DSPRPTXMLA – Display Report-to-Excel Map Attribute**

The DSPRPTXMLA (Display Report-to-Excel Map Attribute) command displays details of a Report-to-Excel Map Attribute.

## **RNMRPTXMLA – Rename Report-to-Excel Map Attribute**

The RNMRPTXMLA (Rename Report-to-Excel Map Attribute) command renames a Report-to-Excel Map Attribute.

## **WRKRPTXMLA – Work with Report-to-Excel Map Attributes**

The WRKRPTXMLA (Work with Report-to-Excel Map Attributes) command lets you work with a list of Report-to-Excel Map Attributes.

### ***MAPNAME – Report-to-Excel map name***

Specify the name of the Report-to-Excel map for which you wish to display list of attributes.

### ***ELEMENT–Element name***

Specify the name of the element within the above Report-to-Excel map for which you wish to display a list of attributes.

## **CRRPTXML – Create Report-to-XML Map**

The CRRPTXML (Create Report-to-XML Map) command creates a report-to-XML map definition describing the content and structure of an XML file to be created from a spooled file.

Once you have created your report-to- XML map you need to specify the different elements and attributes it comprises. See the ADDRPTXML (Add Report-to-XML Map Element) and ADDRPTXMLA (Add Report-to-XML Map Attribute) commands for details of how to do that.

### ***MAPNAME – Report-to- XML map name***

Specify the name you wish to give to the report-to-XML map.

Report map names conform to the normal rules for OS/400 object names, except that they can be up to 20 characters long.

### ***REPORTNAME – Report definition name***

Specify the name of an existing report definition that will define the input to the conversion process. The report map defines the output from the conversion process.

You must be authorized to use the report definition.

### ***DFTUSEAUT - Default use authority***

The default authority to use this report map.

Individual user authorities to the map can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a report-to-XML map is

ARIADNE\_XML\_MAP\_nnnnnnnnnn\_USE

where nnnnnnnnn is the internal map identifier, which is displayed by DSPRPTXML.

Options are:

#### **\*ALLOWED**

By default, users other than the user creating the map are permitted to use it when converting a spooled file to XML.

#### **\*DENIED**

By default, users other than the user creating the map are not permitted to use it when converting a spooled file to XML.

### ***DFTCHGAUT - Default change authority***

The default authority to change or delete this report map.

Individual user authorities to the map can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a report-to-XML map is

ARIADNE\_XML\_MAP\_nnnnnnnnnn\_CHG

where nnnnnnnnn is the internal map identifier, which is displayed by DSPRPTXML.

Options are:

**\*DENIED**

By default, users other than the user creating the map are not permitted to change, delete or manage it.

**\*ALLOWED**

By default, users other than the user creating the map are permitted to change, delete or manage it.

***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the report map.

Options are:

**\*BLANK**

No text is specified.

**Text**

Specify the text ‘description’.

***ELMSEQOPT - Element sequence option***

Determines the order in which row groups are output.

Options are:

**\*MAP**

The order of rows in the XML file is determined by the way row groups are organized in the report-to-XML map.

Rows will be written to the XML worksheet based on the hierarchy of row groups in the report-to-XML map, taking account of parent-child relationships between row groups and the sequence number of child row groups within the parent row group.

**\*SPLF**

The order of rows in the XML file is determined by the order of data in the spooled file.

The following commands also operate on report-to-XML maps. Parameters are only described where they differ significantly from those of the CRTRPTXML command described above.

***CHGRPTXML – Change Report-to-XML map***

The CHGRPTXML (Change Report-to-XML map) command modifies an existing Report-to-XML map.

See CRTRPTXML above for a discussion of the various parameters.

***CPYRPTXML – Copy Report-to-XML map***

The CPYRPTXML (Copy Report-to-XML map) command copies a Report-to-XML map and its associated row groups and cells.

***FROMMAP – From Report-XML map name***

Specify the name of the Report-to-XML map you wish to copy.



## ***TOMAP – To Report-XML map***

Specify the name of the Report-to-XML map you wish to create, based on the Report-to-XML map being copied.

The remaining parameters allow attributes to be modified while the map is being copied. See CRTRPTXML above for a discussion of these parameters.

## ***DLTRPTXML – Delete Report-to-XML map***

The DLTRPTXML (Delete Report-to-XML map) command deletes a Report-to-XML map.

## ***DSPRPTXML – Display Report-to-XML map***

The DSPRPTXML (Display Report-to-XML map) command displays details of a report definition.

## ***RNMRPTXML – Rename Report-to-XML map***

The RNMRPTXML (Rename Report-to-XML map) command renames a Report-to-XML map.

## ***MAPNAME – Report Report-to-XML map***

Specify the name of the Report-to-XML map you wish to rename.

## ***NEWMAP – New Report-to-XML map***

Specify the new name for the Report-to-XML map.

## ***RTVRPTXML – Retrieve Report-to-XML map***

The RTVRPTXML (Retrieve Report-to-XML map) command retrieves CL source for creating a Report-to-XML map and all its associated row groups and cells. This provides a convenient way of saving and distributing a Report-to-XML map to other systems. The source that is retrieved can be easily converted to a program which can be run to create the Report-to-XML map.

## ***MAPNAME – Report-to-XML map***

Specify the name of the Report-to-XML map for which you wish to retrieve source.

## ***SRCFILE – Source file***

Specify the qualified name of the source file into which the source should be retrieved. The file and library must already exist.

## ***SRCMBR – Source member***

Specify the name of the source member into which the source should be retrieved. If the member does not already exist, it will be created.

## ***MBROPT – Source member***

Whether an existing member is replaced or appended to.

Options are:

**\*REPLACE**

If the member already exists, it will be replaced.

**\*ADD**

If the member already exists, the retrieved source will be appended to it.

## **WRKRPTXML – Work with Report-to-XML maps**

The WRKRPTXML (Work with Report-to-XML maps) command displays a list of existing Report-to-XML maps and lets you operate on them or create new maps.

## **Worked Example: Using report definitions and maps**

### **SUPER SUN SEEDS – Customer Order Report**

This worked example used the dummy SUPER SUN SEEDS company and the Customer Order Report DM\_ORDRPT1. This demo report is supplied with CoolSpools as a stream file and it located in the IFS at

**/ariadne/CoolSpoolsV6R1/samples/DM\_ORDRPT1.SPL**

It can be restored to you system as a spooled file by running:

**COOLSPV6R1/RSTSPLF**

**FROMSTMF('/ariadne/CoolSpoolsV6R1/samples/DM\_ORDRPT1.SPL')  
NEWOWN(\*CURRENT)**

A sample report definition called DM\_ORDRPT1 corresponding to this dummy report is supplied with CoolSpools.

The steps below explain how to create a report definition which describes the structure and semantic content of a report so that report can be used to generate meaningful XML and complex Excel files. We suggest you follow the steps below either against the demo DM\_ORDRPT1 Customer Order Report or a report of your own. You can check and compare what you do against the DM\_ORDRPT1 report definition supplied with CoolSpools.

#### **1. Create the Report Definition**

Run WRKRPTDFN (Work with Report Definitions) and press F6=Create or use the CRTRPTDFN (Create Report Definition) command, e.g.

**CRTRPTDFN**

**REPORTNAME(CUSTOMER\_ORDERS)  
TEXT('Customer Order Report')  
DATFMT(\*MDY)  
DATSEP('/')  
CURSYM('\$')  
DECPOINT('.')  
THOUSANDS(',')**

This creates a new report definition called CUSTOMER\_ORDERS. You can now specify the layout of the report either by adding lines to it with ADDRPTLIN or by specifying lines on screen using DSNRPTDFN (Design Report Definition). We'll do things on screen in this example.

#### **2. Design the Report Definition**

Run WRKRPTDFN (Work with Report Definitions) and take option 11 against report definition DM\_ORDRPT1 or use the DSNRPTDFN (Design Report Definition) command e.g.

**DSNTRPTDFN**

**REPORTNAME(CUSTOMER\_ORDERS)**

You now need to select a sample spooled file or the right type to use as the base for specifying the report layout. By default, you will be prompted with a list of spooled

files from your current job. If the spooled file you want to use is in another job, press F14 and change the selection criteria to select a different list of spooled files.

When you've found the spooled file you want to use, select it with option 1=Select.

If your display is capable of running in 27 lines by 132 columns mode, you'll now see something like this:

```
-----
Spooled file: DM_ORDRPT1 Job: QCTLDEV001/ARIADNE/549496 Spl Nbr: 11 Page: 1 of 24
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9...+...0...+...1
001 DM_ORDRPT1 SUPER SUN SEEDS Date: 11/15/09
002 Page: 1 Customer Order Report for 10/01/09 to 10/31/09 Time: 12:57:53
003 Region: NE NORTHEAST
004 State: MA MASSACHUSETTS
005 =====
006 Customer: 000114 TRULY TASTY TURNIPS
007
008 Order Order Total Total Total
009 Number Date Cost Amount Profit
010 000180 10/23/09 615.57 734.67 119.10
011
012 Totals for 000114 TRULY TASTY TURNIPS 615.57 734.67 119.10
013 -----
014
015 Total for: MA MASSACHUSETTS 615.57 734.67 119.10
016
017
018
019
020
More..
F3=Exit F7=Previous page F8=Next page F10=Define line F11=Define section F12=Cancel F13=Show line F14=Show section
F16=Work with lines F17=Work with sections F18=Undefine line F19=Window left F24=More keys
-----
```

Otherwise, you'll see things in 24 lines x 80 columns mode like this:

```
-----
DM_ORDRPT1 QCTLDEV001/ARIADNE/549496 11 1 / 24
*...+...1...+...2...+...3...+...4...+...5...+...6...+...
001 DM_ORDRPT1 SUPER SUN SEEDS
002 Page: 1 Customer Order Report for 10/01/09 to
003 Region: NE NORTHEAST
004 State: MA MASSACHUSETTS
005 =====
006 Customer: 000114 TRULY TASTY TURNIPS
007
008 Order Order Total
009 Number Date Cost
010 000180 10/23/09 615.57
011
012 Totals for 000114 TRULY TASTY TURNIPS 615.57
013 -----
014
015 Total for: MA MASSACHUSETTS 615.57
016
017
More...
F3=Exit F7=Prev page F8=Next page F10=Define line F11=Define section
F12=Cancel F13=Show line F14=Show section F24=More keys
-----
```

You can press F2 to switch between modes if your display supports both modes.

Use the page up/page down keys to scroll up and down the page displayed.

Use the F7 and F8 keys to move to earlier or later pages, or key a page number into the page number box top right to move to a specific page.

### 3. Define the different line types

The next step should be to define the different types of line that occur in the report.

For each type of line which contains data you want to be able to extract, locate an example in the report and press F10 on that line to define it. If that line of the report

is already associated with a line type, you will be able to change the definition of that line type, otherwise you will be prompted to create a new line type.

You should see something like the screen below.

At the top of the screen you will see the line from the report against which you pressed F10 with a “ruler” above it to aid with column identification.

If you position the cursor on a field that has a limited number of possible values and press F4, you will be prompted with a list of possible options.

Specify the following basic information for the line:

➤ **Line name**

Specify a name for the report line. Report line names may be up to 20 characters long but otherwise confirm to the normal system standards for object naming.

If you leave any spaces inside the name, they will automatically be converted to underscores.

➤ **Text 'description'**

Give the line some descriptive text. If no text is specified, it is automatically derived from the line name. Specify \*BLANK if you want the text left blank.

➤ **Line type**

Specify what kind of line it is, e.g.:

- \*DETAIL      A report detail line
- \*PAGHDG    A page heading
- \*COLHDG    A column heading
- \*SUMMARY   A summary line (e.g. totals)
- \*OTHER      Some other type of line

---

### Add Report Line

\*...+....1....+....2....+....3....+....4....+....5....+....6....+....  
002 Page: 1 Customer Order Report for 10/01/09 to

Line name . . . . .  
Text 'description' . . .  
Line type . . . . . \*DETAIL  
Can occur from page . . . \*FIRST Offset . . . . . \*NONE  
Can occur to page . . . \*LAST Offset . . . . . \*NONE  
Can occur from line . . . 2 Can occur to line . . . . 2  
Part of section . . . . \*NONE  
Rule type . . . . . \*LINNBR  
100

F3=Exit F4=List F9=Save F10=Define item F11=Items F19=Left F20=Right

---

## Identifying the line type

Now you need to tell CoolSpools how it can identify a line of this type. For each line type you define, you need to specify a set of criteria which will allow CoolSpools to determine, for any line of text read from the spooled file, whether it is a match for this line type.

There are several possible methods to do this, all of which can be used together.

### ➤ Rule type

You must specify how the line will be identified. The options are:

- **\*LINNBR**

The line type can be identified by its line number alone. Any line on the range of pages specified which falls in the line range specified will be selected and assigned to the line type being defined.

- **\*REPEAT**

The line type can be identified by its line number alone, but is part of a repeat group. A repeat group is a group of related lines that occur together on the page and are repeated down the page, something like this:

```
Line 1
Line 2
Line 3
Line 4
Line 1
Line 2
Line 3
Line 4
Line 1
Line 2
Line 3
```

Line 4

...

The from- and to-line numbers specify the earliest and latest lines on the page between which this particular line type can occur. You must also specify a **repeat group depth** which identifies how many lines there are in the group.

Any line on the range of pages specified which fulfils the line number criteria specified will be selected and assigned to the line type being defined.

For example, taking the repeat group above, the repeat group depth is 4. If the repeat group starts on line 21 and ends on line 50, then each line would be specified as follows:

Line type	From line number	To line number	Possible lines on which this line type can occur
Line 1	21	47	21, 25, 29, 33...
Line 2	22	48	22, 26, 30, 34...
Line 3	23	49	23, 27, 31, 35...
Line 4	24	50	24, 28, 32, 36...

- **\*DEFAULT**

This value identifies the line type as the default, i.e. the line type that is assigned if no other line type is selected.

- **\*RULE**

This value indicates that you wish to apply one or more tests to the line to determine its type. Enter \*RULE for the rule type and press Enter and you will be prompted to enter the tests to be applied.

See below for details.

## ➤ Page range

You can specify the earliest and latest page numbers in the report which can include a line of the type being specified. The page range is defined in terms of a pair of page numbers:

- **Can occur from page**

The earliest possible page on which this line type can occur. Specify a page number or \*FIRST for the first page or \*LAST for the last page.

- **Can occur to page**

The latest possible page on which this line type can occur. Specify a page number or \*FIRST for the first page or \*LAST for the last page.

For example, some lines might appear only on the first page, in which case you could define the range of pages as:

<b>Can occur from page</b>	*FIRST
<b>Can occur to page</b>	*FIRST

When you press F10 on a line, the page range will default to \*FIRST \*LAST.

For each of the from- and to- page numbers, you can also define an associated page offset. The page offset is added to the page number specified to calculate the actual page number. This can be particularly useful, for example, where you need to select a page number relative to the last page, for example the last page but one in the report which is defined using:

<b>Can occur to page</b>	*FIRST
<b>Offset</b>	-1

➤ **Line range**

You can specify the earliest and latest line numbers on each page of the report which can correspond to the type being specified. The line range is defined in terms of a pair of line numbers:

- **Can occur from line**

The earliest possible line on which this line type can occur. Specify a line number or \*FIRST for the first line or \*LAST for the last line.

- **Can occur to line**

The latest possible line on which this line type can occur. Specify a line number or \*FIRST for the first line or \*LAST for the last line.

For example, a line type which can only occur on line 4 of the report might be defined like this:

<b>Can occur from line</b>	4
<b>Can occur to line</b>	4

***When you press F10 on a line, the from- and to- lines both default to the line number on which F10 was pressed. Be sure to modify this if the line type can occur on lines other than just this line number.***

➤ **Rule**

If page range and line number range are not sufficient to identify a line type, you can specify a line rule. Line rules allow you to define a set of tests that will uniquely identify the line. These are explained in detail below.



## ➤ Rule evaluation priority

The rule evaluation priority is a number between 1 and 999 which specifies the order in which the rules associated with different line types are evaluated. Since the first line type where the rule set evaluates to true will be selected, you can use the rule evaluation priority to prioritize one line type before another. Typically, you would prioritize a default or “catch-all” rule last, so that other rules are given an opportunity to select a line first.

For example, imagine you have a report with two summary lines which contain the following labels that you propose to use to identify them:

**Customers who purchased this month**  
**Customers who purchased this month last year**

If you define a rule to check for the text “Customers who purchased this month”, it will potentially match **both** lines and select the wrong line the for the line where the text is “Customers who purchased this month last year”. In order to select the right line types, you could set the rule priorities like this:

<b>Customers who purchased this month</b>	<b>Priority = 100</b>
<b>Customers who purchased this month last year</b>	<b>Priority = 050</b>

thus ensuring that the second rule is tested first (because its rule priority setting is lower) and that rule will select just the line where the text is “Customers who purchased this month last year” leaving the other rule to select the line where the text is “Customers who purchased this month”.

## Specifying line rules

If you enter \*RULE for the rule type and press Enter, the screen will change to allow entry of tests to be applied to the line.

```

-----
                          Add Report Line

      *...+....1....+....2....+....3....+....4....+....5....+....6....+....
003 Region:      NE      NORTHEAST

Line name . . . . . REGION_HEADER
Text 'description' . . . Region Header
Line type . . . . . *DETAIL
Can occur from page . . . *FIRST      Offset . . . . . *NONE
Can occur to page . . . *LAST      Offset . . . . . *NONE
Can occur from line . . . 3      Can occur to line . . . 3
Part of section . . . . *NONE
Rule type . . . . . *RULE
Rule eval priority . . . 100

Rel  Line      Off  Pos  Cmp Value
*IF  *CURRENT *NONE 1    *EQ

More...
F3=Exit  F4=List  F9=Save  F10=Define item  F11=Items  F19=Left  F20=Right
-----

```

The tests consist of the following:

### ➤ “Rel” (Relationship)

This identifies the relationship between each test. Options are:

- \*IF                      This value is mandatory on the first line and is possible only on the first line.
- \*AND                    Indicates that this test is part of an AND group with the previous line. The rule will be true only if the result of all tests in an AND group is true.
- \*OR                     Indicates that this test starts a new OR group. The rule will be true if the combined result of any OR group is true.

### ➤ Line (line number)

Specifies the number of the line to be tested. Options are:

- \*CURRENT              The test applies to the current line, i.e. the line which is being analyzed to determine its type.
- \*FIRST                The test applies to the first line on the page.
- \*LAST                 The test applies to the last line on the page.
- Line number          Specify the absolute line number on the page of the line to be tested, e.g. a value of 1 here would cause the test to be applied to Line 1 on the page.

### ➤ Offset

Specifies the offset from the line number specified on the previous element to the actual line number to be tested. Options are:

- \*NONE                 No offset is applied. The line number alone identifies the line to be tested.

- **Offset** Specify a number that is added to the line number to obtain the actual line number to be tested.

Typically, this is used in conjunction with a line number of \*CURRENT or \*LAST to specify a line number relative to the current line or the last line on the page.

For example:

Line: \*CURRENT Offset: 1

denotes the line following the line being analyzed, while:

Line: \*LAST Offset: -1

denotes the last but one line on the page.

### ➤ **Pos (character position)**

Identifies the position on the line to be tested. Options are:

- **Position** Specify the starting position on the line of the first character to be tested or compared to the comparison value.
- **\*TYPE** Indicates that the comparison value contains the name of line type. This allows you to test the type of other lines on the page and identify the type of this line by reference to those lines.

### ➤ **Cmp (Comparison type)**

Specifies the type of comparison to be applied. Options are:

- **\*EQ** The rule is true if the value at the specified position on the line to be tested is equal to the comparison value.
- **\*NE** The rule is true if the value at the specified position on the line to be tested is not equal to the comparison value.
- **\*GT** The rule is true if the value at the specified position on the line to be tested is greater than to the comparison value.
- **\*LT** The rule is true if the value at the specified position on the line to be tested is less than to the comparison value.
- **\*GE** The rule is true if the value at the specified position on the line to be tested is greater than or equal to the comparison value.
- **\*LE** The rule is true if the value at the specified position on the line to be tested is less than or equal to the comparison value.

Only \*EQ and \*NE are valid if \*TYPE was specified for the comparison type.

### ➤ **Value (comparison value)**

Specifies the value against which the test occurs.

In addition to specifying simple constant strings on this field, you can also use regular expressions and patterns. These are explained below.

#### Examples:

Rel	Line	Off	Pos	Cmp	Value	Description of test
*IF	*CURRENT	*NONE	1	*EQ	Region:	The test will evaluate to true if the line starts with the value "Region:"
*IF	*CURRENT	1	3	*NE	' '	The test will evaluate to true if the third character of the next line is not a blank.  Note that it is necessary to enclose the value in apostrophes if it consists of all blanks.
*IF	*LAST	-1	*TYPE	*EQ	END_OF_REPORT	The test will evaluate to true if the last but one line of the current page is of type END_OF_REPORT.
*IF	*CURRENT	*NONE	1	*EQ	\$\$PATTERN(####.##)	This test checks if the current line, starting at position 1, matches the pattern string specified. Patterns are explained below.
*IF	*CURRENT	*NONE	1	*EQ	\$\$REGEX(regular_expression_string)	This test checks if the current line, starting at position 1, matches the regular expression string specified. Regular expressions are explained below.

### Patterns and regular expressions

One powerful and useful technique for defining rules for identifying line types is to use the \$\$PATTERN and/or \$\$REGEX CoolSpools functions.

\$\$PATTERN tests the value at a specified position on a line of text against a given pattern string. A pattern string consists of a series of characters which denote actual characters or sets of character. These are explained in the table below.

Patterns are implemented by converting them to a regular expression string and using regular expression processing. Hence they are highly efficient like regular expressions but easier to define and understand if you are unfamiliar with regular expressions.

The pattern string must follow \$\$PATTERN and be enclosed in parentheses (). The string can also optionally be enclosed in single quotes '.

Pattern Symbol	Denotes	Corresponding regular expression element	Comments
. (period)	A character including space	.	
X	Any character except space	[^ ]	
A	Any character except space or a digit (0-9)	[^ 0-9]	
#	A digit (0-9), thousands separator, currency symbol, minus sign or space	[0-9,\$- ]	The thousands separator and currency symbol are those associated with the report definition.  Useful for referring to areas of the page which contain edited numeric values.
9	A digit (0-9)	[0-9]	Useful for referring to areas of the page that contain unedited numbers. Use # instead of 9 if the number is edited (has zero suppression, thousands separators, minus signs or currency symbols).
\X	An X	[X]	Where a character is used as a pattern symbol, precede that character by a backslash \ to denote the actual character not the pattern symbol.
\.	A period	[.]	
\A	An A	[A]	
\#	A hash	[#]	
\9	A 9	[9]	
Any other	The character specified	[char]	Any other character just denotes that

			character itself.
--	--	--	-------------------

Examples:

Pattern string	Denotes
\$\$PATTERN(XXX 999)	Three non-space characters followed by 2 spaces and then 3 numeric digits.
\$\$PATTERN(Totals for: 999999)	The string "Totals for:" followed by a space and then 6 numeric digits.
\$\$PATTERN(999999 #9/99/99 #####\.)	Six numeric digits followed by 6 spaces then a date (allowing for zero suppression on the first digit) then 4 spaces then an edited number. Note the use of # rather than 9 to allow for zero suppression and the presence of thousands separators and currency symbols. Also note the backslash before the period to indicate that the pattern is checking for an actual period at that position.

If you are familiar with regular expressions, you can use the \$\$REGEX CoolSpools function to define even more powerful rules.

\$\$REGEX tests the value at a specified position on a line of text against a given regular expression.

Refer to <http://www.regular-expressions.info/> for information on regular expressions.

\$\$REGEX regular expressions are case-sensitive and support

### Line section

You can also define this line as belonging to a section. Since sections cannot be defined until their associated lines have been defined, you will need to do this later. Leave the section name as \*NONE at this time.

### Saving the line definition

When you have finished defining the line, press F9 to save your changes.

You will be returned to the main screen. Any text lines on the page that now match a line definition will be colored pink. Those that do not match a line definition will remain their original color.

Press F13 and the name of the matching line definition will be displayed.

If for some reason a line of the report has been associated with the wrong line type (for example, if you defined a line rule too broadly and lines were included that should not have been), press F18 to undefine the line, i.e. disassociate it from the line type. You can then press F10 on that line and define a new line type for it.

Alternatively, press F15 to work with report lines. You can then rename or modify the existing line definitions or add new ones.

Repeat the above steps until all lines in the report for which you wish to process data have been correctly defined to and identified by CoolSpools. You do not need to define blank lines and other lines that contain no data (e.g. banners and separators).

### Command-line alternative

You can also add lines to a report definition using the ADDRPTLIN command. Report lines can be changed with CHGRPTLIN, removed with RMVRPTLIN, copied with CPYRPTLIN, displayed with DSPRPTLIN and renamed with RNMRPTLIN.

## 4. Define the report items for each line

Now that you have defined the lines, you need to define the data items those lines comprise.

Position your cursor on a line which has already been defined and has a correctly associated line type. Press F10 to work with that line.

You will see something like this. At the top of the screen there is part of the text line on which you pressed F10. At the bottom are the details of the currently associated line definition.

```
-----
                                Change Report Line

      *...+....1....+....2....+....3....+....4....+....5....+....6....+....
006 Customer:    000151  EVERGREEN & HARWOOD SEEDS

Line name . . . . . CUSTOMER_HEADER
Text 'description' . . . Customer heading line
Line type . . . . . *DETAIL
Can occur from page . . . *FIRST      Offset . . . . . *NONE
Can occur to page . . . *LAST      Offset . . . . . *NONE
Can occur from line . . . 6        Can occur to line . . . . 50
Part of section . . . . *NONE
Rule type . . . . . *RULE
Rule eval priority . . . 100

Rel  Line      Off  Pos  Cmp Value
*IF  *CURRENT *NONE 1    *EQ Customer:

More...
F3=Exit  F4=List  F9=Save  F10=Define item  F11=Items  F19=Left  F20=Right
-----
```

Position your cursor on the text line at the top of the screen at the beginning of the data item to be defined. If the data item is not shown because it is to the left or right of the portion of the text line shown, use the F19=Left and F20=Right keys to window the display left and right. When your cursor is at the beginning of the item, press F10 to define it. You will see something like this.

```

-----
                                Add Report Item

      *...+....1....+....2....+....3....+....4....+....5....+....6....+....
006 Customer:    000151  EVERGREEN & HARWOOD SEEDS
                ^^^^^^

Report item name . . . .
Text 'description' . . .

Part of section . . . . . *LINE
Text 'description' . . .

Character position . . . 13          Character length . . . . 6

Item type . . . . . *VAR          Data type . . . . . *NUMERIC

F3=Exit   F4=List   F9=Save   F12=Cancel
-----

```

The item definition consists of the following. If you position the cursor on a field that has a limited number of possible values and press F4, you will be prompted with a list of possible options.

#### ➤ **Report item name**

Specify a name for the report item. Report item names may be up to 20 characters long but otherwise confirm to the normal system i standards for object naming.

If you leave any spaces inside the name, they will automatically be converted to underscores.

#### ➤ **Text 'description'**

Give the item some descriptive text. If no text is specified, it is automatically derived from the item name. Specify \*BLANK if you want the text left blank.

#### ➤ **Part of section**

Identifies the section to which the item belongs.

This defaults to \*LINE indicating that the item is part of the section associated with the line to which the item belongs, but can be changed to a different section name where appropriate (if a line contains items belonging to more than one section). We will define sections later, so for now leave this as \*LINE.

#### ➤ **Character position**

The character position (column) at which the data item starts on the line. This will automatically have been defaulted by CoolSpools to the cursor position at which you pressed F10.



## ➤ Character length

The number of characters the data item includes starting from and including the character position specified above.

Note that where the item that is being defined is a number, be careful to include the character positions occupied by:

- digits at the beginning of the number which might not be apparent on the text line you selected because of zero suppression but which might be present on other lines where the value of the data item is larger.
- trailing minus signs at the end of the number which might not be apparent on the text line you selected because it is positive but which might be present on other lines where the value of the data item is negative.

CoolSpools will default the length to a value calculated by counting characters to the right from the cursor position at which you pressed F10, stopping at and excluding the first space following the first block of non-space characters. Make sure you check that this length is correct, which it will not be, for example, if the data item you are defining includes embedded blanks.

The characters currently selected for the data item are shown by means of a set of ^ symbols under them.

## ➤ Item type

Specify the type of item being defined:

- \*VAR            The item is a variable
- \*CONST        The item is a constant
- \*LABEL        The item is a label (text associated with a variable).

CoolSpools will default this to \*VAR.

The distinction between these types is of no great significance at this time but may be used by future features.

## ➤ Data type

Specify the type of data the item consists of:

- \*ALPHA        Alphanumeric
- \*NUMERIC     Numeric
- \*DATE         Date

CoolSpools will default this to a value derived from the value of the item on the current text line at the position identified when you pressed F10. Check that this

is correct, which it might not be, for example, for an alphanumeric variable where the current value happens to consist of only numeric characters.

### ➤ **Date format**

Where \*DATE was specified for the data type, specify the format of the date this data item consists of:

- \*RPTDFN      Date format associated with the report definition.
- \*JOB          Current job date format
- \*SYSVAL      Date format identified by the QDATFMT system value.
- \*YMD          Year-month-day
- \*MDY          Month-day-year
- \*DMY          Day-month-year

CoolSpools defaults this field to \*RPTDFN.

### ➤ **Date separator**

Where \*DATE was specified for the data type, specify the separator character used to edit the date this data item consists of:

- \*RPTDFN      Date separator associated with the report definition.
- \*JOB          Current job date format
- \*SYSVAL      Date format identified by the QDATFMT system value.
- \*NONE        No separator character
- sep\_char      Specify the separator character used.

CoolSpools defaults this field to \*RPTDFN.

## **Saving the item definition**

When you have finished defining the item, press F9 to save your changes.

You will be returned to the main screen.

Repeat the above steps until all items on all lines in the report for which you wish to process data have been correctly defined to and identified by CoolSpools. You do not need to define blank areas of the page or other items that contain no data (e.g. banners and separators) or which contain data which you do not require to be included in files created from this report definition.

## **Command-line alternative**

You can also add items to a report definition using the ADDRPTITM command. Report items can be changed with CHGRPTITM, removed with RMVRPTITM, copied with CPYRPTITM, displayed with DSPRPTITM and renamed with RNMRPTITM.

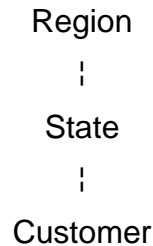
## **5. Define the report sections**

### **What is a report section?**

Now that you have defined the lines and the items they comprise, you need to define the section structure of the report.

Most reports have some kind of section structure, and it is important that CoolSpools knows about this structure in order to be able to create meaningful output from your report, for example XML documents where elements are nested correctly.

For example, in the case of the demo Customer Order Report DM\_ORDRPT1 (see above), the report lists orders for a given date range by customer, within US state, by region of the USA. There are therefore 3 sections that CoolSpools needs to know about and these form the following section hierarchy:



The region section of the report comprises one or more states and each state shown in the report comprises one or more customers.

If, for example, you want CoolSpools to be able to build an XML document from the report which takes the following form:

```
<region>
  <state>
    <customer/>
    <customer/>
    <customer/>
  </state>
  <state>
    <customer/>
    <customer/>
  </state>
</region>
```

...

etc.

then CoolSpools needs to know how sections relate to one another and when each section starts and ends.

Sections define the relationship between the various lines in a report. In general, where two lines in the report are related to one another, and may need to be handled as a unit, they should be defined as being part of the same section, or as being part of different sections that are themselves related in terms of a section hierarchy. For example, if the order information for each order in the Customer Order Report were to cover two lines, those lines should be defined as making up an order section. CoolSpools can then treat the two lines as a single entity and correctly process the data for a single order from both lines together (e.g. outputting that data to a single XML element or Excel row).

## Defining report sections

To define a section, press F11 anywhere on the report.

You will see something like this.

```
-----
                                Add Section

Section name  . .
Text 'description'
Parent section . . *NONE                Section type  . . *DETAIL
Start line name . .                    End line name  . . *NEXTSTART
End line in sect . *YES                End rules same . . *NO
----- Section Start Rules -----
Rel  Item name          Cmp Value
*IF

More...

----- Section End Rules -----
Rel  Item name          Cmp Value
*IF

More...

----- Included Lines -----

More...

F3=Exit  F4=List  F9=Save  F11=Work with lines  F12=Cancel
-----
```

If you position the cursor on a field that has a limited number of possible values and press F4, you will be prompted with a list of possible options.

Specify the following basic information for the line:

### ➤ Section name

Specify a name for the report section. Report section names may be up to 20 characters long but otherwise conform to the normal system standards for object naming.

If you leave any spaces inside the name, they will automatically be converted to underscores.

### ➤ Text 'description'

Give the section some descriptive text. If no text is specified, it is automatically derived from the section name. Specify \*BLANK if you want the text left blank.

### ➤ Parent section

Specify what kind of line it is, e.g.:

### ➤ Parent section

This field allows the creation of a section hierarchy. If you specify the name of another section here, the section you are defining will be a child section of that parent section.

The default is \*NONE which indicates a top-level section with no parent.

### ➤ **Parent section**

What type of section this is:

- \*DETAIL A report detail section
- \*PAGHDG A page heading section
- \*COLHDG A column heading section
- \*SUMMARY A summary section (e.g. totals)
- \*REPORT A report-level section (comprising the entire report)
- \*OTHER Some other type of section

This item is not important at this time but may be used by future features.

### ➤ **Start line name**

Specify the name of the line which starts the section. For example, in the case of the CUSTOMER section in the Customer Order Report, it is the line type called CUSTOMER\_HEADER which marks the start of a group of lines for a new customer.

### ➤ **End line name**

Specify the name of the line which ends the section. For example, in the case of the CUSTOMER section in the Customer Order Report, it is the line type called CUSTOMER\_TOTAL which marks the end of a group of lines for a customer.

The default is \*NEXTSTART, which indicates that the end a section of this type can only be determined by the occurrence of the next start line (as defined above). For example, the CUSTOMER section could equally (but less elegantly and accurately) be defined as starting with CUSTOMER\_HEADER and ending with the next CUSTOMER\_HEADER.

### ➤ **End line in section**

Whether the line specified as the end line for the section should be included in the section or not:

- \*YES The end line is part of the section it ends. For example, the CUSTOMER\_TOTAL line is part of the CUSTOMER section it ends.
- \*NO The end line is not part of the section it ends. For example, where \*NEXTSTART is defined for a section, the end line is not part of the section, but rather the start of the next section of the same type.

### ➤ **End start rules same**

Whether, when section rules need to be defined, the same rules apply both to the ending of the section as to the starting of the section, or whether different rules need to be defined for each.

- \*YES Only one set of rules needs to be defined.
- \*NO Two sets of rules will be defined

## Section rules

Sometimes, start and end line types on their own are not enough to identify the beginning or end of a section.

For example, in the sample Customer Order Report, the REGION section can span many pages and each page starts with a region header reiterating the current region code and name. While the REGION\_HEADER line is indeed the start line for the REGION section, not every REGION\_HEADER line starts a new region section. We need to use a section rule to identify the true start of a new section. Specifically, a new region section starts with a REGION\_HEADER line where the region code is different from the previous region code.

Section rules are similar to line rules and consist of one or more tests which comprise the options.

### ➤ “Rel” (Relationship)

This identifies the relationship between each test. Options are:

- \*IF This value is mandatory on the first line and is possible only on the first line.
- \*AND Indicates that this test is part of an AND group with the previous line. The rule will be true only if the result of all tests in an AND group is true.
- \*OR Indicates that this test starts a new OR group. The rule will be true if the combined result of any OR group is true.

### ➤ Item name

Specifies the name of a report item to be tested.

Press F4 to select the item from a list of items defined for the report.

### ➤ Cmp (Comparison type)

Specifies the type of comparison to be applied. Options are:

- \*EQ The rule is true if the value of the report item to be tested is equal to the comparison value.
- \*NE The rule is true if the value of the report item to be tested is not equal to the comparison value.
- \*GT The rule is true if the value report item to be tested is greater than to the comparison value.
- \*LT The rule is true if the value report item to be tested is less than to the comparison value.
- \*GE The rule is true if the value report item to be tested is greater than or equal to the comparison value.
- \*LE The rule is true if the value report item to be tested is less than or equal to the comparison value.

### ➤ Value (comparison value)

Specifies the value against which the test occurs. The value of the data item is compared to the value specified here.

Often the special \*PRV value will be needed. This denotes the previous value of the data item. For example, in the Customer Order Report, a new section starts when the region code changes, which is specified as:

**\*IF REGION\_CODE \*NE \*PRV**

### Included lines

List the line types which form part of this section.

Press F4 to select from a list. Multiple selections can be made by inputting 1 against each line type to be included in the section.

### Saving the item definition

When you have finished defining the section press F9 to save your changes.

You will be returned to the main screen.

Repeat the above steps until all sections in the report for which you wish to process data have been correctly defined to and identified by CoolSpools.

Press F14 to display the section associated with a line.

### Command-line alternative

You can also add sections to a report definition using the ADDRPTSCT command. Report sections can be changed with CHGRPTSCT, removed with RMRPTSCT, copied with CPYRPTSCT, displayed with DSPRPTSCT and renamed with RNMRPTSCT.

## 6. Create a Report-to-XML Map

Now that you have created a report definition, CoolSpools knows how the information in your report is structured, so when it processes your report as the input to a conversion, it knows where to find the information the report contains and how the various parts of the report relate to one another. Now you can use your report as the input to the CVTSPLXML (Convert Spooled File to XML) and CVTSPLXL (Convert Spooled File to Excel) commands in order to generate complex XML and Excel files, but first you must tell CoolSpools how you want to structure the output you create, i.e. the XML documents and Excel spreadsheets you create.

First, we'll create a report-to-XML map to create some XML. A sample Report-to-XML map for the demo Customer Order Report DM\_ORDRPT1 is supplied with CoolSpools. A report-to-XML map tells CoolSpools how to map between a report and a definition.

**Source code for all commands shown below can be found in demo source file DM\_CLSRC supplied with CoolSpools.**

Run WRKRPTXML (Work with Report-to-XML Maps) and press F6=Create or use the CRTRPTXML (Create Report-to-XML Map) command to create a new map, e.g.

#### CRTRPTXML

**MAPNAME(CUSTOMER\_ORDERS)  
REPORTNAME(CUSTOMER\_ORDERS)  
DFTUSEAUT(\*ALLOWED)**

**DFTCHGAUT(\*DENIED)**  
**TEXT('Example Report-to-XML map: Customer**  
**Order Report')**  
**ELMSEQOPT(\*MAP)**

This creates a new report-to-XML map called CUSTOMER\_ORDERS. On the REPORTNAME parameter, specify the name of the report definition you created earlier. This tells CoolSpools where to look for the definition of data items, sections etc that you will reference while specifying the map.

The XML document we're going to create will consist of a hierarchy or tree of elements something like this:

```
customerOrders
  |
  region
    |
    state
      |
      customer
        |
        order
```

## 1. Add elements to a Report-to-XML Map

Run WRKRPTXML (Work with Report-to-XML Maps), select 8=Elements against the report-to-XML map you just created and press F6=Create, or use the ADDRPTXMLE (Add Report-to-XML Map Element) command to add a new element to the map.

First add the root element customerOrders, something like this:

**ADDRPTXMLE**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(customerOrders)**  
**PARENT(\*NONE)**  
**SEQNBR(\*NONE)**  
**ITEMNAME(\*NONE)**  
**TEXT('Root Element for Customer Orders Report')**  
**NEWELMOPT(\*NEVER)**

Note that element names are case-sensitive (like most things XML) and can be up to 50 characters long.

This will be the root element of the XML document because it has no parent element (PARENT(\*NONE) was specified. You must also specify NEWELMOPT(\*NEVER) for the root element, indicating that a new element is never created: there will be just the one node of this type in the entire document and it will be the parent or ancestor of all other nodes.

ITEMNAME(\*NONE) is specified because this element will have no text node: it will consist entirely of child elements and attributes, which will be defined later.



Now define a child element of the root called region corresponding to the REGION section of the report. A new region element will be started every time a new REGION section starts:

```
ADDRPTXMLE  
  MAPNAME(CUSTOMER_ORDERS)  
  ELEMENT(region)  
  PARENT(customerOrders)  
  SEQNBR(1)  
  ITEMNAME(*NONE)  
  Customer Orders Report')  
  NEWELMOPT(*SECTION)  
  SECTION(REGION)
```

Now define a child element of region called state corresponding to the STATE section of the report.

```
ADDRPTXMLE  
  MAPNAME(CUSTOMER_ORDERS)  
  ELEMENT(state)  
  PARENT(region)  
  SEQNBR(1)  
  ITEMNAME(*NONE)  
  TEXT('State element for Customer Orders Report')  
  NEWELMOPT(*SECTION)  
  SECTION(STATE)
```

Next define a child element of state called customer corresponding to the CUSTOMER section of the report.

```
ADDRPTXMLE  
  MAPNAME(CUSTOMER_ORDERS)  
  ELEMENT(customer)  
  PARENT(state)  
  SEQNBR(1)  
  ITEMNAME(*NONE)  
  TEXT('Customer element for Customer Orders Report')  
  NEWELMOPT(*SECTION)  
  SECTION(CUSTOMER)
```

Finally, define a child element of customer called order corresponding to the ORDER\_LINE line type of the report. A new order node will be created every time a new ORDER\_LINE line is encountered.

```
ADDRPTXMLE  
  MAPNAME(CUSTOMER_ORDERS)  
  ELEMENT(order)  
  PARENT(customer)  
  SEQNBR(1)  
  ITEMNAME(*NONE)  
  TEXT('Order element for Customer Orders Report')  
  NEWELMOPT(*LINE)  
  LINENAME(ORDER_LINE)
```

## 7. Add attributes to Report-to-XML Map Elements

Run WRKRPTXML (Work with Report-to-XML Maps), select 8=Elements against the report-to-XML map you created earlier, then select 9=Attributes against the root element customerOrders and press F6=Create, or else use the ADDRPTXMLA (Add Report-to-XML Map Attribute) command to add a new attribute to the root element customerOrders.

First we'll add attributes corresponding to the report-level data items, namely the from- and to-dates and the report totals.

### **ADDRPTXMLA**

```
MAPNAME(CUSTOMER_ORDERS)  
ELEMENT(customerOrders)  
ATTRIBUTE(fromDate)  
SEQNBR(1)  
ITEMNAME(FROM_DATE *DATE)  
TEXT('From date')
```

Note that attribute and element names are both case-sensitive (as per XML) and can be up to 50 characters long.

ITEMNAME(FROM\_DATE) is specified here to tell CoolSpools to derive the value of this attribute from the current value of the report data item called FROM\_DATE you defined earlier.

Now define the other attributes for the root element:

### **ADDRPTXMLA**

```
MAPNAME(CUSTOMER_ORDERS)  
ELEMENT(customerOrders)  
ATTRIBUTE(toDate)  
SEQNBR(2)  
ITEMNAME(TO_DATE *DATE)  
TEXT('To date')
```

### **ADDRPTXMLA**

```
MAPNAME(CUSTOMER_ORDERS)  
ELEMENT(customerOrders)  
ATTRIBUTE(cost)  
SEQNBR(3)  
ITEMNAME(REPORT_TOTAL_COST *NUMERIC)  
TEXT('Report total cost')
```

### **ADDRPTXMLA**

```
MAPNAME(CUSTOMER_ORDERS)  
ELEMENT(customerOrders)  
ATTRIBUTE(value)  
SEQNBR(4)  
ITEMNAME(REPORT_TOTAL_VALUE *NUMERIC)  
TEXT('Report total value')
```

### **ADDRPTXMLA**

```
MAPNAME(CUSTOMER_ORDERS)  
ELEMENT(customerOrders)
```

```
ATTRIBUTE(profit)
SEQNBR(5)
ITEMNAME(REPORT_TOTAL_PROFIT *NUMERIC)
TEXT('Report total profit')
```

Now define the attributes for the other elements:

```
/* Region element */
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ATTRIBUTE(code)
  SEQNBR(1)
  ITEMNAME(REGION_CODE *ALPHA)
  TEXT('Region code')
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(region)
  ATTRIBUTE(cost)
  SEQNBR(3)
  ITEMNAME(REGION_TOTAL_COST *NUMERIC)
  TEXT('Region total cost')
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(region)
  ATTRIBUTE(name)
  SEQNBR(2)
  ITEMNAME(REGION_NAME *ALPHA)
  TEXT('Region name')
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(region)
  ATTRIBUTE(profit)
  SEQNBR(5)
  ITEMNAME(REGION_TOTAL_PROFIT *NUMERIC)
  TEXT('Region total profit')
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(region)
  ATTRIBUTE(value)
  SEQNBR(4)
  ITEMNAME(REGION_TOTAL_VALUE *NUMERIC)
  TEXT('Region total value')
```

```
/* State element */
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(state)
```

**ATTRIBUTE(code)**  
**SEQNBR(1)**  
**ITEMNAME(STATE\_CODE \*ALPHA)**  
**TEXT('State code')**

**ADDRPTXMLA**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(state)**  
**ATTRIBUTE(cost)**  
**SEQNBR(3)**  
**ITEMNAME(STATE\_TOTAL\_COST \*NUMERIC)**  
**TEXT('State total cost')**

**ADDRPTXMLA**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(state)**  
**ATTRIBUTE(name)**  
**SEQNBR(2)**  
**ITEMNAME(STATE\_NAME \*ALPHA)**  
**TEXT('State name')**

**ADDRPTXMLA**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(state)**  
**ATTRIBUTE(profit)**  
**SEQNBR(5)**  
**ITEMNAME(STATE\_TOTAL\_PROFIT \*NUMERIC)**  
**TEXT('State total profit')**

**ADDRPTXMLA**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(state)**  
**ATTRIBUTE(value)**  
**SEQNBR(4)**  
**ITEMNAME(STATE\_TOTAL\_VALUE \*NUMERIC)**  
**TEXT('State total value')**

***/\* Customer element \*/***

**ADDRPTXMLA**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(customer)**  
**ATTRIBUTE(cost)**  
**SEQNBR(3)**  
**ITEMNAME(CUST\_TOTAL\_COST \*NUMERIC)**  
**TEXT('Customer total cost')**

**ADDRPTXMLA**  
**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(customer)**  
**ATTRIBUTE(name)**  
**SEQNBR(2)**

**ITEMNAME(CUSTOMER\_NAME \*ALPHA)**  
**TEXT('Customer name')**

**ADDRPTXMLA**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(customer)**  
**ATTRIBUTE(number)**  
**SEQNBR(1)**  
**ITEMNAME(CUSTOMER\_NUMBER \*NUMERIC)**  
**TEXT('Customer number')**

**ADDRPTXMLA**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(customer)**  
**ATTRIBUTE(profit)**  
**SEQNBR(5)**  
**ITEMNAME(CUST\_TOTAL\_PROFIT \*NUMERIC)**  
**TEXT('Customer total profit')**

**ADDRPTXMLA**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(customer)**  
**ATTRIBUTE(value)**  
**SEQNBR(4)**  
**ITEMNAME(CUST\_TOTAL\_VALUE \*NUMERIC)**  
**TEXT('Customer total value')**

**/\* Order element \*/**

**ADDRPTXMLA**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(order)**  
**ATTRIBUTE(cost)**  
**SEQNBR(3)**  
**ITEMNAME(ORDER\_COST \*NUMERIC)**  
**TEXT('Order cost')**

**ADDRPTXMLA**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(order)**  
**ATTRIBUTE(date)**  
**SEQNBR(2)**  
**ITEMNAME(ORDER\_DATE \*DATE)**  
**TEXT('Order date')**

**ADDRPTXMLA**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ELEMENT(order)**  
**ATTRIBUTE(number)**  
**SEQNBR(1)**  
**ITEMNAME(ORDER\_NUMBER \*NUMERIC)**  
**TEXT('Order number')**

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(order)
  ATTRIBUTE(profit)
  SEQNBR(5)
  ITEMNAME(ORDER_PROFIT *NUMERIC)
  TEXT('Order profit')
```

```
ADDRPTXMLA
  MAPNAME(CUSTOMER_ORDERS)
  ELEMENT(order)
  ATTRIBUTE(value)
  SEQNBR(4)
  ITEMNAME(ORDER_VALUE *NUMERIC)
  TEXT('Order value')
```

## **8. Use a Report-to-XML Map to generate an XML document**

Now you have defined your report-to-XML map, you can use it to convert a spooled file of the right type to an XML document. Run something like this:

```
CVTSPLXML
  MAPNAME(CUSTOMER_ORDERS)
  FROMFILE(DM_ORDRPT1)
  TOSTMF(DM_ORDRPT1.xml)
  SPLNBR(*LAST)
  STMFOPT(*REPLACE)
```

and the resultant XML document should look like the sample file supplied with CoolSpools and stored in the IFS as

**/ariadne/CoolSpoolsV6R1/samples/dm\_ordrpt1.xml**

## **9. Create a Report-to-Excel Map**

The process of creating a Report-to-Excel map is very similar to that of creating a report-to-XML map, except that you will work in terms of row groups rather than elements and cells rather than attributes.

**Source code for all commands shown below can be found in demo source file DM\_CLSRC supplied with CoolSpools.**

A row group is a set of one or more related rows that are output to the Excel worksheet as a group.

A cell is a single cell (row/column intersection) within a row group, with a value which is either a text constant or a variable derived from a report data item.

Run WRKRPTXL (Work with Report-to-Excel Maps) and press F6=Create or use the CRTRPTXL (Create Report-to-Excel Map) command to create a new map, e.g.

```
CRTRPTXL
  MAPNAME(CUSTOMER_ORDERS)
  REPORTNAME(CUSTOMER_ORDERS)
  DFTUSEAUT(*ALLOWED)
```

**DFTCHGAUT(\*DENIED)**  
**TEXT('Report-to-Excel map for Customer Order Report')**  
**GRPSEQOPT(\*MAP))**

This creates a new report-to-Excel map called CUSTOMER\_ORDERS. On the REPORTNAME parameter, specify the name of the report definition you created earlier. This tells CoolSpools where to look for the definition of data items, sections etc that you will reference while specifying the map.

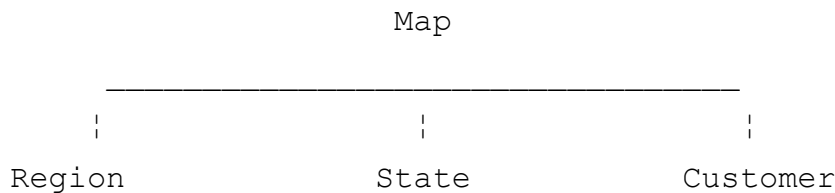
The Excel file document we're going to create will consist of a set of different row types for the region, state, customer and each customer order.

### **10. Add row groups to a Report-to-Excel Map**

Run WRKRPTXL (Work with Report-to-Excel Maps), select 8=Row groups against the report-to-Excel map you just created and press F6=Create, or use the ADDRPTXLR (Add Report-to-Excel Row Group) command to add a new row group to the map.

An Excel map does not need a root row group in the same way an XML map requires a root element. However, like XML elements, Excel row groups can be nested inside one another in a parent-child relationship. It is important to define these relationships correctly in order to obtain the right results. The Excel file is built by using the Excel map as a template to generate a tree structure from the report data and the structure of the map you define is crucial in determining the structure of the Excel files you create. Specifically, rows in the Excel file are output in the sequence of the corresponding row groups at a particular level in the hierarchy of row groups in the Excel map.

For example, if you were to define an Excel map for the Customer Orders Report that had row groups corresponding to the region, state and customer sections at the same level, like this:



the resultant Excel file would have rows for all of regions first, then rows for all states, then rows for all customers, thus:

Region = NORTHEAST

Region = SOUTH

Region = WEST

...

State = MASSACHUSETTS

State = NEW JERSEY

State = NEW YORK

...

Customer = TRULY TASTY TURNIPS  
Customer = PRAIRIE TREE AND SEED  
Customer = EVERGREEN & HARWOOD SEEDS

...

whereas defining the correct hierarchy thus:

```
Map
|
Region
|
State
|
Customer
```

would give properly nested results, thus:

Region = NORTHEAST  
State = MASSACHUSETTS  
Customer = TRULY TASTY TURNIPS

...

State = NEW JERSEY  
Customer = PRAIRIE TREE AND SEED

...

State = NEW YORK  
Customer = EVERGREEN & HARWOOD SEEDS

...

Region = SOUTH  
State = FLORIDA  
Customer = SARAH'S SAFARI FRUITS  
Customer = ABUNDANT FRUIT & FLOWERS

...

Note that unlike element names, row group name are not case-sensitive and conform to the normal conventions for system i object names, except that they can be up to 20 characters long.

The commands

Add the following row groups:

```
ADDRPTXLR
      MAPNAME(CUSTOMER_ORDERS)
      ROWGRPNAME(CUSTOMER_ORDERS)
```



**PARENT(\*NONE)**  
**SEQNBR(\*NEXT)**  
**TEXT('Report parameter row for Customer Orders Report')**  
**NEWGRPOPT(\*NEVER)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ROWGRPNAME(STATE)**  
**SEQNBR(\*NEXT)**  
**TEXT('State header row')**  
**NEWGRPOPT(\*SECTION)**  
**SECTION(STATE)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ROWGRPNAME(CUSTOMER)**  
**SEQNBR(\*NEXT)**  
**TEXT('Customer header row')**  
**NEWGRPOPT(\*SECTION)**  
**SECTION(CUSTOMER)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ROWGRPNAME(ORDER)**  
**SEQNBR(\*NEXT)**  
**TEXT('Order line')**  
**NEWGRPOPT(\*LINE)**  
**LINENAME(ORDER\_LINE)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ROWGRPNAME(ORDER)**  
**SEQNBR(\*NEXT)**  
**TEXT('Order line')**  
**NEWGRPOPT(\*LINE)**  
**LINENAME(ORDER\_LINE)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ROWGRPNAME(REGION\_TOTALS)**  
**SEQNBR(\*NEXT)**  
**TEXT('Region totals row')**  
**NEWGRPOPT(\*SECTION)**  
**SECTION(REGION\_TOTAL)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)**  
**ROWGRPNAME(STATE\_TOTALS)**  
**SEQNBR(\*NEXT)**  
**TEXT('State totals row')**  
**NEWGRPOPT(\*SECTION)**

**SECTION(STATE\_TOTAL)**

**ADDRPTXLR**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_TOTALS)  
SEQNBR(\*NEXT)  
TEXT('Customer totals row')  
NEWGRPOPT(\*SECTION)  
SECTION(CUSTOMER\_TOTAL)**

## **11. Add cells Report-to-Excel Row Groups**

**ADDRPTXLC**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_ORDERS)  
ROWNR(1)  
COLUMN(A)  
CONTENT(\*TEXT)  
CELLTEXT('CUSTOMER ORDER REPORT')  
MRGCELLS(1 F)**

**ADDRPTXLC**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_ORDERS)  
ROWNR(2)  
COLUMN(A)  
CONTENT(\*EMPTY)  
MRGCELLS(2 F)**

**ADDRPTXLC**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_ORDERS)  
ROWNR(3)  
COLUMN(A)  
CONTENT(\*TEXT)  
CELLTEXT('From date:')**

**ADDRPTXLC**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_ORDERS)  
ROWNR(3)  
COLUMN(B)  
CONTENT(\*ITEM)  
CELLITEM(FROM\_DATE)**

**ADDRPTXLC**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_ORDERS)  
ROWNR(3)  
COLUMN(C)  
CONTENT(\*TEXT)**

**CELLTEXT('To date:')**

**ADDRPTXLC**

**MAPNAME(CUSTOMER\_ORDERS)  
ROWGRPNAME(CUSTOMER\_ORDERS)  
ROWNBR(4)  
COLUMN(D)  
CONTENT(\*ITEM)  
CELLITEM(TO\_DATE)**

## **12. Use a Report-to-Excel Map to generate an Excel file**

Now you have defined your report-to-XML map, you can use it to convert a spooled file of the right type to an XML document. Run something like this:

**CVTSPLXL**

**MAPNAME(CUSTOMER\_ORDERS)  
FROMFILE(DM\_ORDRPT1)  
TOSTMF(DM\_ORDRPT1.xls)  
SPLNBR(\*LAST)  
STMFOPT(\*REPLACE)**

and the resultant XML document should look like the sample file supplied with CoolSpools and stored in the IFS as

**/ariadne/CoolSpoolsV6R1/samples/dm\_ordrpt1.xls**

## **Digital Signatures**

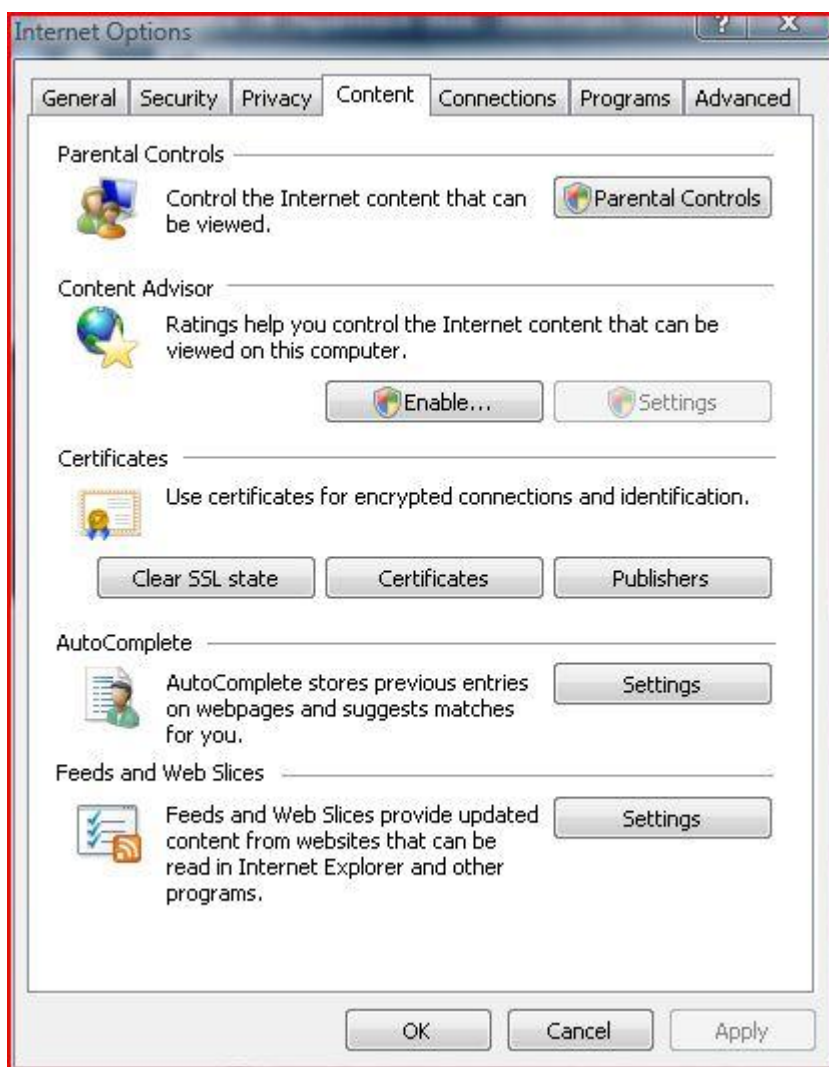
In order to add a digital signature to a PDF, you first need to obtain a suitable digital certificate. Digital certificates can be obtained from a variety of authorities, including specialist organizations such as Verisign, Thawte and Comodo.

Once you have obtained a certificate, it will need to be exported in PKCS#12 format for CoolSpools to be able to use it. If you have obtained a digital certificate online from an organization such as those mentioned above, you will probably have already installed the certificate in your PC's browser. To export the certificate for use with signing PDFs, follow these steps (this example relates to IE 8 but similar options are available in earlier versions of IE and other browsers).

From the toolbar, select **Tools -> Internet Options**

Click on the **Content** tab

Click the **Certificates** button in Certificates section



Select the certificate you want to use and click **Export**

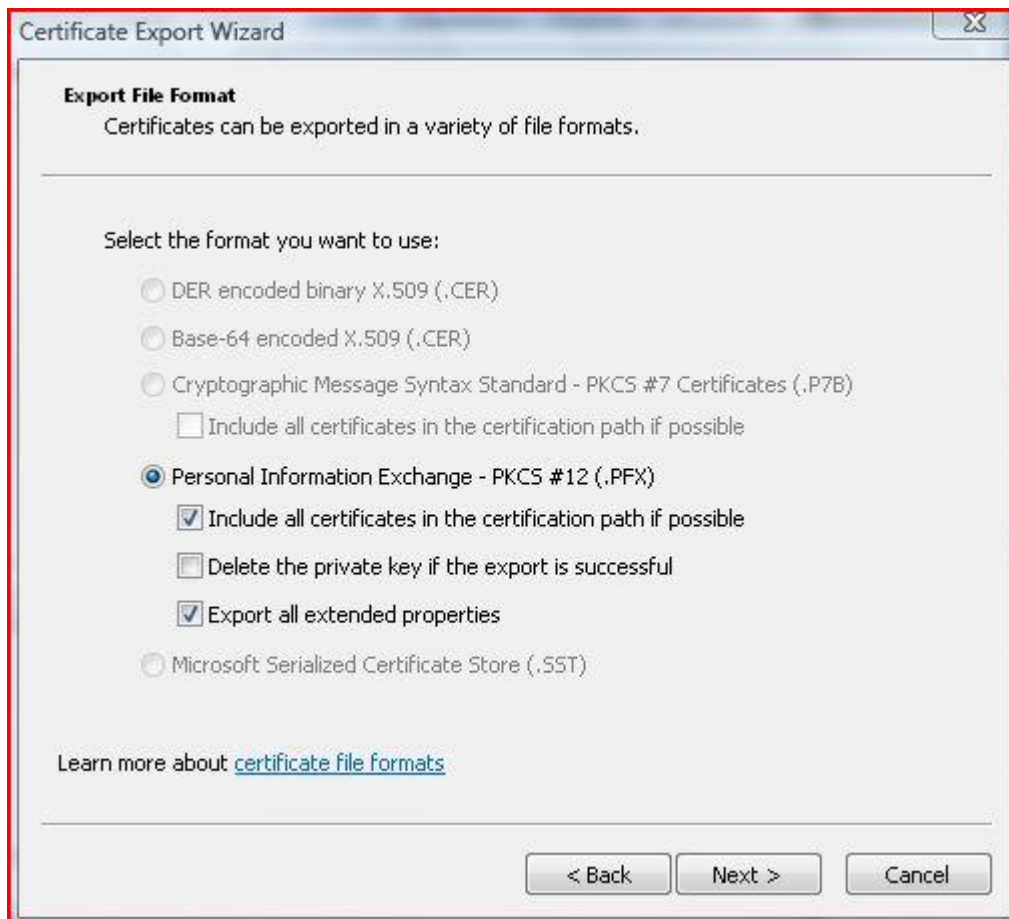
Click **Next**

Select **Yes, export the private key** and click **Next**



Select **Personal Information Exchange – PKCS #12 (.PFX)**, check **Include all certificates in the certification path if possible** and **Export all extended properties** and click **Next**

It is also possible to export certificate files in PKCS12 format from the IBM Key Management utility supplied with System i Access or from the system i's own Certificate Store using Digital Certificate Manager.



Enter a password for the file and click **Next**

Specify a path for the file and click **Next**. IE8 will add an extension of .pfx to whatever you specify.

Click **Finish**

If you did not export the file directly to the IFS of your system i, you should now transfer the file to somewhere in the IFS where you can reference it and CoolSpools can read it when you use it.

When you create a PDF with CVTSPLPDF, specify the file you just created on the **Certificate file path element** of the SIGNATURE parameter and specify the password you created for it on the **Certificate password element**.

## **Commands related to Parameter Sets**

### **CRTPRMSET – Create Parameter Set**

The CRTPRMSET (Create Parameter Set) command creates a [parameter set](#) that stores and provides a convenient way of retrieving a set of command parameters.

#### **PRMSETNAME – Parameter set name**

Specify the name you wish to give to the parameter set.

Parameter set names conform to the normal rules for OS/400 object names, except that they can be up to 50 characters long.

#### **CMDUSER – User running command**

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, only parameter sets where this attribute matches the user profile of the user running the command will be selected.

Options are:

<b>*CURRENT</b>	The current user profile is the one for which a parameter set is being created.
<b>*SYSDFT</b>	The parameter set being defined is a system default parameter set, i.e. the parameter set will potentially match all users. The system checks first for a parameter set specific to the user running the command, but, if none is found, will then check for a system default parameter set instead.
<b>user-profile</b>	Specify the user profile.

The following restrictions apply:

- In order to create a system default parameter set (i.e. one where CMDUSER(\*SYSDFT) is specified, you must either be a system administrator or be authorized to registered function **ARIADNE\_PRM\_SET\_SYS\_DFT\_CHG**.
- In order to create a parameter set for a user other than yourself, you must either be a system administrator or be authorized to registered function **ARIADNE\_PRM\_SET\_OTH\_USR\_CHG**.
- In order to create a parameter set for yourself, you must either be a system administrator or be authorized to registered function **ARIADNE\_PRM\_SET\_OWN\_USR\_CHG**.

A system administrator is any user with \*SYSADM or \*ALLOBJ special authorities or who is authorized to registered function **ARIADNE\_SYS\_ADMIN**.

#### **SPLF- Spooled file name**

Specifies the name of the spooled file to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, only parameter sets where this attribute matches the name of the spooled file being converted will be selected.

Options are:

<b><u>*ALL</u></b>	The parameter set is relevant to all spooled files and spooled file name is not a pertinent criterion when the system is searching for a parameter set to use.
<b>character-value</b>	Specify the name of the spooled file. *generic* values may be specified, i.e. you may use * as a wildcard matching one or more characters either at the start or the end of the name.

### ***SPLUSER - Spooled user profile***

Specifies the name of the spooled file user to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, only parameter sets where this attribute matches the user profile of the owner of the spooled file being converted will be selected.

Options are:

<b><u>*ALL</u></b>	The parameter set is relevant to all spooled file users and spooled file user is not a pertinent criterion when the system is searching for a parameter set to use.
<b>character-value</b>	Specify the owning user profile. *generic* values may be specified, i.e. you may use * as a wildcard matching one or more characters either at the start or the end of the name.

### ***SPLNBR - Spooled file number in job***

Specifies the number of the spooled file in the job to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, only parameter sets where this attribute matches the spooled file number of the spooled file being converted will be selected.

This option can be useful where there are several spooled files with the same name in the same job and you need to specify different parameter sets for those different spooled files.

Options are:

<b><u>*ALL</u></b>	The parameter set is relevant to all spooled file numbers and spooled file number is not a pertinent criterion when the system is searching for a parameter set to use.
--------------------	---



<b>*ONLY</b>	There is only one spooled file of the specified name in the job.
<b>*LAST</b>	The spooled file is the last of its name in the job.
<b>1-999999</b>	Specify the spooled file number.

### ***OUTQ - Spooled file output queue***

Specifies the name of the output queue to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, the system will only select parameter sets where this attribute matches the output queue on which the spooled file which is being converted is located.

Single values:

<b><u>*ALL</u></b>	The parameter set is relevant to all output queues and output queue is not a pertinent criterion when the system is searching for a parameter set to use.
--------------------	---

Qualifier 1: Spooled file output queue

The name of the output queue.

Options are:

<b>*ANY</b>	The parameter set is relevant to all output queues in the library specified below.
<b>generic-name</b>	Specify the output queue. generic* values may be specified, i.e. you may use * as a wildcard matching one or more characters at the end of the name.

Qualifier 2: Library

Specify the library in which the output queue exists.

Options are:

<b>*ANY</b>	The parameter set is relevant to all output queues of the name specified above, irrespective of the library in which the output queue is located.
<b>character-value</b>	Specify the name of the library in which the output queue exists.

### ***FORMTYPE - Spooled file form type***

Specifies the form type to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, the system will only select parameter sets where this attribute matches the form type of the spooled file which is being converted.

Options are:

<b>*ALL</b>	The parameter set is relevant to all form types and form type is not a pertinent criterion when the system is searching for a parameter set to use.
<b>*STD</b>	The standard form type.
<b>character-value</b>	Specify the form type. *generic* values may be specified, i.e. you may use * as a wildcard matching one or more characters either at the start or the end of the name.

### ***USRDTA - Spooled file user data***

Specifies the user data value to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, the system will only select parameter sets where this attribute matches the user data of the spooled file which is being converted.

Options are:

<b><u>*ALL</u></b>	The parameter set is relevant to all user data values and user data is not a pertinent criterion when the system is searching for a parameter set to use.
<b>character-value</b>	Specify the user data. *generic* values may be specified, i.e. you may use * as a wildcard matching one or more characters either at the start or the end of the name.

### ***JOB - Spooled file job name***

Specifies the name of the job to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, the system will only select parameter sets where this attribute matches the job name in which the spooled file which is being converted was created.

Options are:

<b>*ALL</b>	The parameter set is relevant to all job names and job name is not a pertinent criterion when the system is searching for a parameter set to use.
<b>generic-name</b>	Specify the name or generic name of the job in which the spooled file was created.

### ***PGM - Spooled file program name***

Specifies the name of the creating program to which this parameter set relates.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, the system will only select parameter sets where this attribute matches the name of the program that created the spooled file which is being converted.

Single values

Options are:

**\*ALL**

The parameter set is relevant to all creating programs and creating program is not a pertinent criterion when the system is searching for a parameter set to use.

Qualifier 1: Spooled file program name

Options are:

**\*ANY**

The parameter set is relevant to all programs in the library specified below.

**generic-name**

Specify the generic name of the program that created the spooled file.

Qualifier 2: Library

Options are:

**\*ANY**

The parameter set is relevant to all programs of the name specified above, irrespective of the library in which the program exists.

**generic-name**

Specify the generic name of the library in which the creating program exists.

## ***PRIORITY - Evaluation priority***

Specifies the evaluation priority of the parameter set.

When RTVPRMSET(\*SPLF) is specified on CVTSPLPDF or any of the other commands that support parameter sets, the system searches for a matching parameter set in the order of ascending priority numbers and the first matching parameter set is selected. You can therefore use this attribute to prioritize specific parameter sets over more general or default parameter sets.

Options are:

5000

The default is the "median" value 5000.

1-9999

Specify a priority between 1 and 9999, where 1 is the highest priority. Rules with the highest priority (lowest priority number) are selected first.

## ***CMD - Command string***

Specifies a command string which defines:

- The command to which this parameter set relates
- The associated parameters which are identified by means of this parameter set.

Options are:

command-string Specify the command string. When you are defining the parameter string, you should specify the following special values:

- FROMFILE(\*SLT)
- JOB(\*SLT)
- SPLNBR(\*SLT)

This indicates to the system that the parameters relate to the spooled file that is being converted at the time the parameter set is used.

- RTVPRMSET(\*NONE)

This is needed to avoid defining a nested or recursive parameter set.

### ***DFTUSEAUT – Default use authority***

The default authority to use this parameter set.

Individual user authorities to the report can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a parameter set is

#### **ARIADNE\_PRM\_SET\_nnnnnnnnnn\_USE**

where nnnnnnnnn is the internal parameter set ID, which is displayed by DSPPRMSET.

Options are:

##### **\*ALLOWED**

By default, users other than the user creating the report are permitted to use it.

##### **\*DENIED**

By default, users other than the user creating the report are not permitted to use it.

### ***DFTCHGAUT – Default change authority***

The default authority to change or delete this parameter set.

Individual user authorities to the report can be managed by means of the IBM CHGFCNUSG command or CoolSpools' WRKREGFNC. The function controlling authority to use a parameter set is

#### **ARIADNE\_PRM\_SET\_nnnnnnnnnn\_CHG**

where nnnnnnnnn is the internal parameter set ID, which is displayed by DSPPRMSET.

Options are:

##### **\*DENIED**

By default, users other than the user creating the parameter set are not permitted to change, delete or manage it.

##### **\*ALLOWED**

By default, users other than the user creating the parameter set are permitted to change, delete or manage it.

## ***TEXT – Text ‘description’***

Specify up to 50 characters of free-format descriptive text to help you identify the parameter set.

Options are:

<b><u>*BLANK</u></b>	No text is specified.
<b>character-value</b>	Specify the text description .

## ***Examples for CRTPRMSET***

### **Example 1: Simple Command Example**

```
CRTPRMSET  
PRMSETNAME(BLUE_ON_YELLOW)  
CMDUSER(*SYSDFT)  
CMD(CVTSPDPDF FROMFILE(*SLT) JOB(*SLT) SPLNBR(*SLT) COLOR(*BLUE  
*YELLOW))  
TEXT('Blue on yellow')
```

This command creates a parameter set called BLUE\_ON\_YELLOW.

The CMDUSER(\*SYSDFT) is specified, identifying the parameter set as a system default set, i.e. it will potentially be matched to any user running the CVTSPLPDF command.

The CMD parameter indicates that the command to which this parameter set relates is CVTSPLPDF and the associated parameter string specifies a text and background colour - blue on yellow.

Finally, the descriptive text 'Blue on yellow' is assigned to the parameter set. In order to use this parameter set, you would run the CVTSPLPDF command and specify RTVPRMSET(BLUE\_ON\_YELLOW). The values specified here for the PDF colours would then override the defaults on the CVTSPLPDF command.

### **Example 2: More Complex Command Example**

```
CRTPRMSET  
PRMSETNAME(INVOICES)  
CMDUSER(*SYSDFT)  
SPLF(INVOICE)  
CMD(CVTSPDPDF FROMFILE(*SLT) JOB(*SLT) SPLNBR(*SLT)  
INCLFILE((invoice_form.jpg *JPG)))  
TEXT('Default parameters for converting invoices to PDF')
```

This command creates a parameter set called INVOICES. The CMDUSER(\*SYSDFT) is specified, identifying the parameter set as a system default set, i.e. it will potentially be matched to any user running the CVTSPLPDF command.

The SPLF parameter indicates that, when RTVPRMSET(\*SPLF) parameter is specified on the CVTSPLPDF command or any other command that supports parameter sets, this parameter set should only be selected where the name of the spooled file being converted is INVOICE.

The CMD parameter indicates that the command to which this parameter set relates is CVTSPLDPF and the associated parameter string specifies a forms overlay image to be included.

If the CVTSPLDPF command were run against a spooled file called INVOICE and the default value RTVPRMSET(\*SPLF) specified, this parameter set could potentially be selected and the INCLFILE parameter specified here override the command defaults.

The following commands also operate on parameter sets. Parameters are only described where they differ significantly from those of the CRTPRMSET command described above.

### **CHGPRMSET – Change Parameter Set**

The CHGPRMSET (Change Parameter Set) command modifies an existing parameter set.

See CRTPRMSET above for a discussion of the various parameters.

### **CPYPRMSET – Copy Parameter Set**

The CPYPRMSET (Copy Parameter Set) command copies a parameter set and its associated report lines, items and sections.

#### ***FROMPRMSET – From parameter set name***

Specify the name of the parameter set you wish to copy.

#### ***TOPRMSET – To parameter set name***

Specify the name of the parameter set you wish to create, based on the parameter set being copied.

The remaining parameters allow attributes to be modified while the parameter set is being copied. See CRTPRMSET above for a discussion of these parameters.

### **DLTPRMSET – Delete Parameter Set**

The DLTPRMSET (Delete Parameter Set) command deletes a parameter set.

### **DSPPRMSET – Display Parameter Set**

The DSPPRMSET (Display Parameter Set) command displays details of a parameter set.

### **RNMPRMSET – Rename Parameter Set**

The RNMPRMSET (Rename Parameter Set) command renames a parameter set.

#### ***PRMSETNAME – Parameter set name***

Specify the name of the parameter set you wish to rename.

## ***NEWPRMSET – New parameter set name***

Specify the new name for the parameter set.

## ***WRKPRMSET – Work with Parameter Set***

The WRKPRMSET (Work with Parameter Set) command lets you work with a list of parameter sets.

## ***CFGPRMSET – Configure Parameter Set***

The CFGPRMSET (Configure Parameter Set) command runs a “wizard” which guides you through the process of creating a parameter set step by step.

## ***SAMPLEPLF – Sample spooled file***

Identifies a spooled file which will be used as the model for defining parameters for conversion of this type of spooled file.

Single values

Options are:

<b><u>*SELECT</u></b>	Select the spooled file from a list.
-----------------------	--------------------------------------

### ***Element 1: Spooled file name***

Options are:

<b>name</b>	Specify the name of the spooled file.
-------------	---------------------------------------

### ***Element 2: Spooled file job***

The job in which the spooled file was created.

Single values:

<b>*</b>	The current job.
----------	------------------

Qualifier 1: Job name

Options are:

<b>name</b>	Specify the name of the job.
-------------	------------------------------

Qualifier 2: User name

<b>name</b>	Specify the job user.
-------------	-----------------------

Qualifier 3: job number

<b>000000-999999</b>	Specify the job number.
----------------------	-------------------------

Element 3: Spooled file number

The spooled file number in the job.

Options are:

<b>*LAST</b>	The last spooled file of the specified name in the job.
<b>*ONLY</b>	The only spooled file of the specified name in the job.
<b>1-999999</b>	Specify the spooled file number.

### ***PRMSETNAME – Parameter set name***

Specify the name of the existing parameter set you wish to modify, if any.

Options are:

<b><u>*NONE</u></b>	No existing parameter set is specified. A new parameter set will be created.
<b>name</b>	Specify the name of the parameter set.